LECTURE 7: NONPARAMETRIC SMOOTHING IN REGRESSION: KERNEL REGRESSION REVISITED

REFERENCES

Shalizi Chapter 4 and Sections 1.5 and 8.3, Appendices A and B

OVERVIEW

This lecture is intended to give some motivation for nonparametric smoothing in regression, as well as point out some of the associated challenges. It's difficult, in general, to give a precise definition of "nonparametric" inference but the basic idea is to use data to infer an unknown quantity *while making as few assumptions as possible*. Usually, this means using statistical models that are *infinite-dimensional*, versus "parametric" models that can be parametrized by a finite number of parameters (such as the β parameters in linear regression).

In the context of regression, if we assume that the (unknown) regression function $r \in \mathcal{F}$ where $\mathcal{F}$ is finite-dimensional — such as the set of straight lines, $\mathcal{F}_{lin} = \{\beta_0 + \beta_1 x : \beta_0, \beta_1 \in \mathbb{R}\}$ — then we have a *parametric regression model*. If we assume that $r \in \mathcal{F}$ where $\mathcal{F}$ is not finite-dimensional — such as, the set of all functions that are not "too wiggly" as defined by the Sobolev space $\mathcal{F}_{sob} = \{r : \int (r''(x))^2 \, dx < \infty\}$ — then we have a *nonparametric regression model*.

As previously discussed, nonparametric regression models are more flexible than parametric models with a smaller bias, but there's a price we pay in variance, even if we choose the smoothing/tuning parameters in the model in an optimal way. In this lecture, we are going to revisit one nonparametric regression estimator, the kernel smoother, and discuss its bias-variance tradeoff.

QUESTIONS

1. How does one use kernel regression in practice?

2. How does one choose bandwidths and kernel functions?

3. How do we measure the performance of kernel regressors with different amounts of smoothing?

■ Definitions and Notations

1. Kernel regression
2. The Nadaraya–Watson kernel estimator
3. Big-O notation

■ Topics

1. Kernel regression
2. Bias and variance of kernel smoothers
3. The curse of dimensionality

■ What's Next?

Lecture 8: Additive Models

Review: Kernel Regression

Reading: Shalizi section 1.5.2

Recall our basic setup: We are given n pairs of observations $(X_1, Y_1), \ldots, (X_n, Y_n)$, where

$$Y_i = r(X_i) + \epsilon_i, \quad i = 1, \ldots, n,$$

and

$$r(x) = \mathbb{E}[Y|X = x] = \int y f(y|x)\, dy.$$

Our goal is to estimate the unknown regression function r with some function $\hat{r}$. Assume for now that each $X_i \in \mathbb{R}$ (i.e., the predictors are 1-dimensional).

We have discussed considering $\hat{r}$ in the class of so-called **linear smoothers**, i.e. regression estimators $\hat{r}$ which has the form $\hat{r}(x) = \sum_i \ell_i(x) Y_i$ for some choice of weights $\ell_i(x)$. Linear regression, k-nearest-neighbors regression and splines are special cases of linear smoothers.

Here we will revisit another important linear smoother, namely **kernel smoothing** a.k.a **kernel regression** or **Nadaraya–Watson regression**, that takes a weighted average of the Y_i's, giving higher weight to those points near x. The starting point is to define a "kernel" function $K : \mathbb{R} \to \mathbb{R}$. For our purposes, the word **kernel** refers to any (usually smooth) function K such that $K(x) \geq 0$ and

Notes:

$$\int K(x)\, dx = 1, \quad \int xK(x)\, dx = 0 \quad \text{and} \quad \sigma_K^2 \equiv \int x^2 K(x) dx > 0.$$

The first condition just scales the kernel function; the second ensures the kernel is somewhat symmetric around $x = 0$; the third implies that $K(x) \to 0$ as $x \to \infty$.

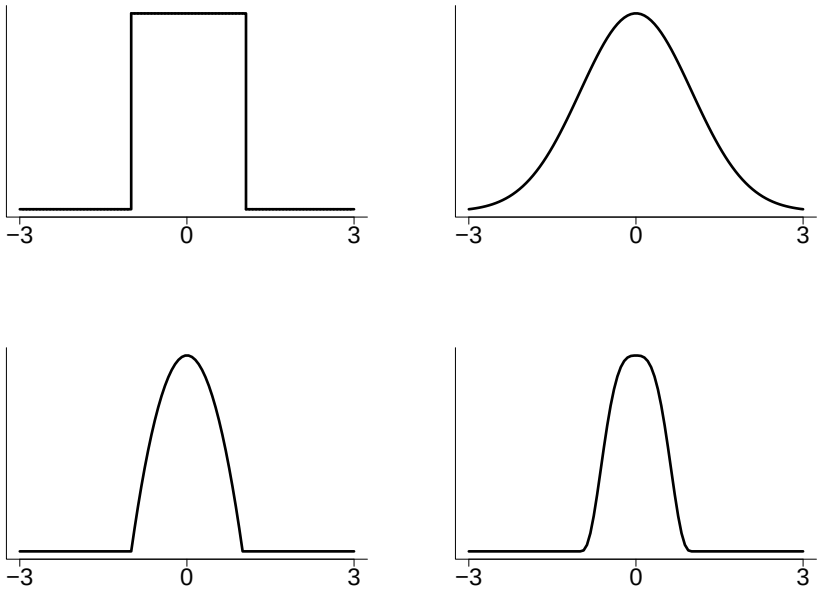

Figure 1: Examples of kernels: boxcar (top left), Gaussian (top right), Epanechnikov (bottom left), and tricube (bottom right).

Some commonly used kernels are the following:

$$\begin{aligned}
\text{the boxcar kernel:} \quad & K(x) = \frac{1}{2} I(x), \\
\text{the Gaussian kernel:} \quad & K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, \\
\text{the Epanechnikov kernel:} \quad & K(x) = \frac{3}{4}(1 - x^2) I(x) \\
\text{the tricube kernel:} \quad & K(x) = \frac{70}{81}(1 - |x|^3)^3 I(x)
\end{aligned}$$

where

$$I(x) = \begin{cases} 1 & \text{if } |x| \leq 1 \\ 0 & \text{if } |x| > 1. \end{cases}$$

These kernels are plotted in Figure 1.

Let $h > 0$ be a positive number, called the *bandwidth*. The *Nadaraya–Watson kernel*

estimator is defined by

$$\hat{r}_n(x) = \sum_{i=1}^{n} w(x, X_i) Y_i$$

where K is a kernel and the weights $w(x, X_i)$ are given by

Notes:

$$w(x, X_i) = \frac{K\left(\frac{x - X_i}{h}\right)}{\sum_{j=1}^{n} K\left(\frac{x - X_j}{h}\right)}.$$

The bandwidth controls how quickly the weight drops off as x moves away from each observed X_i.

Think: What does this achieve? What happens to data "close to" versus "far away" from the evaluation point x? What are the differences between NW regression and k-NN regression?

Notes: The estimated regression function is a weighted average of Y_i values. This makes it a linear smoother. Think about laying doing a Gaussian kernel around a specific query point x, and evaluating its height at each x_i in order to determine the weight associate with Y_i. **Draw a picture!**

Because these weights vary smoothly with x, the kernel regression estimator $\hat{r}(x)$ will vary smoothly with x. For k-nearest-neighbors regression, there is a jump every place where the set of nearest neighbors changes, but the regression function is flat otherwise.

Q: What's in the choice of kernel? Different kernels can give different results. But many of the common kernels tend to produce similar estimators; e.g., Gaussian vs. Epanechnikov, there's not a huge difference. The one big difference is what happens when you extrapolate. For x that are below the smallest x_j or above

the largest x_j, the Gaussian kernel regression will eventually predict something very close to the most extreme value in the same direction as x. The Epanechnikov kernel (and each of the other bounded kernels) will eventually be unable to predict at all because all of the weights become 0.

What does matter much more is the *choice of bandwidth* h which controls the amount of smoothing. What's the tradeoff when we vary h? Hint: as we've mentioned before, you should always keep two quantities in mind...

Bias and Variance of Kernel Smoothers

Reading: Shalizi sections 4.1–4.2, Appendix A, Appendix B

Imagine that we want to make predictions at a new predictor value X which might be random with a density f (or f_X.) Conditional on $X = x$, recall how the generalization or prediction error decomposes (**review Lecture 3**):

$$\begin{aligned} R(x) = \mathbb{E}[\text{TestErr}(\hat{r}(x))] &= \mathbb{E}\left[\left(Y - \hat{r}(x)\right)^2 \middle| X = x\right] \\ &= \sigma^2 + \text{Bias}(\hat{r}(x))^2 + \text{Var}(\hat{r}(x)). \end{aligned}$$

The overall risk would be $\int R(x) f(x)\, dx$, by the law of iterated expectations.

Questions: So what is the bias and variance of the kernel regression estimator? How do these terms depend on the smoothing parameter h_n and the sample size n? What is the optimal bandwidth h_n? How does the prediction error (for a kernel smoother with an optimal bandwidth h_n) depend on n?

Fortunately, these can roughly be worked out theoretically under some smoothness assumptions on r (and other assumptions). Using Taylor expansion (see Shalizi, Appendix B), one can show that the bias at x is

$$\mathbb{E}[\hat{r}(x) - r(x) | X_1 = x_1, \ldots, X_n = x_n] = h^2 \left[\frac{1}{2} r''(x) + \frac{r'(x) f'(x)}{f(x)}\right] \sigma_K^2 + o(h^2) \quad (1)$$

where f is the density of x, and $\sigma_K^2 = \int u^2 K(u) du$ is the variance of the probability density corresponding to the kernel. One can also work out the variance of the kernel regression estimator,

$$\text{Var}[\hat{r}(x)|X_1 = x_1, \ldots, X_n = x_n] = \frac{\sigma^2 C(K)}{nhf(x)} + o\left((nh)^{-1}\right) \tag{2}$$

where $C(K) \equiv \int K^2(u) du$. Do these terms make sense? What happens to the bias and variance as h shrinks (i.e. less smoothing)? As h grows (i.e. more smoothing)? Where does the sample size come in to the equation? What about the regression function itself?

Notes: As h gets larger, the bias term goes up and the variance term goes down. The bias makes sense because large h means that you give high weight to Y_i values whose x_i are not so close to x. When h is small, you give high weight only to Y_i whose x_i are close to x. These facts suggest that the bias should be small for small h and large for large h.

The variance claim makes sense because large h means that you are including more data into your weighted average, thereby making more of the noise terms cancel each other. Small h means that $\hat{r}(x)$ is based on very few data values, and hence will have larger variance. The nh in the denominator of the variance term suggests that small h means that we are getting less of each of the n observations contributing to the average.

The appearance of $r'(x)f'(x)$ in the bias calculation comes from how many x_j you expect to get on each side of x and whether the mean function $r(\cdot)$ is larger or smaller on each side. The σ_K^2 comes from the fact that you get sums of things like $(x - x_i)^2 K(x - x_i)$ from the second-derivative approximation.

Putting the bias together with the variance, we get an expression for the mean squared error of the kernel regression conditional on $X = x$, $R(x)$. Integrating

the MSE gives that the risk of the NW kernel estimator is

$$R(h_n) = \frac{h_n^4}{4}\sigma_K^4 \int \left(r''(x) + 2r'(x)\frac{f'(x)}{f(x)} \right)^2 f(x)dx + \frac{\sigma^2 \int K^2(x)dx}{nh_n} + o([nh_n]^{-1}) + o(h_n^4) + \sigma^2 \quad (3)$$

as $h_n \to 0$ and $nh_n \to \infty$.

That is, using big-O order symbols, we have that:

Notes:

$$R(h_n) = \sigma^2 + \mathcal{O}(h_n^4) + \mathcal{O}([nh_n]^{-1}),$$

as $n \to \infty$, $h_n \to 0$, $nh_n \to \infty$.

If you're not familiar: we say some function $f(x)$ is $\mathcal{O}(1)$ if $f(x)/c \to 1$ as $x \to \infty$, for some constant $c < \infty$. We say $f(x)$ is $\mathcal{O}(g(x))$ if $f(x)/g(x) = \mathcal{O}(1)$. You can think of this as saying that "f grows at the same rate as g", and the notation lets us drop constants and smaller-order terms.

Little-o notation is similar, except it says that $f(x)$ is $o(1)$ if $f(x)/c \to 0$ as $x \to \infty$ for *any* nonzero constant c.

If we differentiate (3) and set the result equal to 0, we find that the *optimal bandwidth* h_* is

$$h_* = \left(\frac{1}{n}\right)^{1/5} \left(\frac{\sigma^2 \int K^2(x)dx}{\sigma_K^4 \int \left(r''(x) + 2r'(x)\frac{f'(x)}{f(x)} \right)^2 f(x)dx} \right)^{1/5}. \quad (4)$$

Thus, $h_* = \mathcal{O}(n^{-1/5})$.

Plugging h_* back into (3) we see that the risk decreases at rate $\mathcal{O}(n^{-4/5})$ when we use the optimal bandwidth.

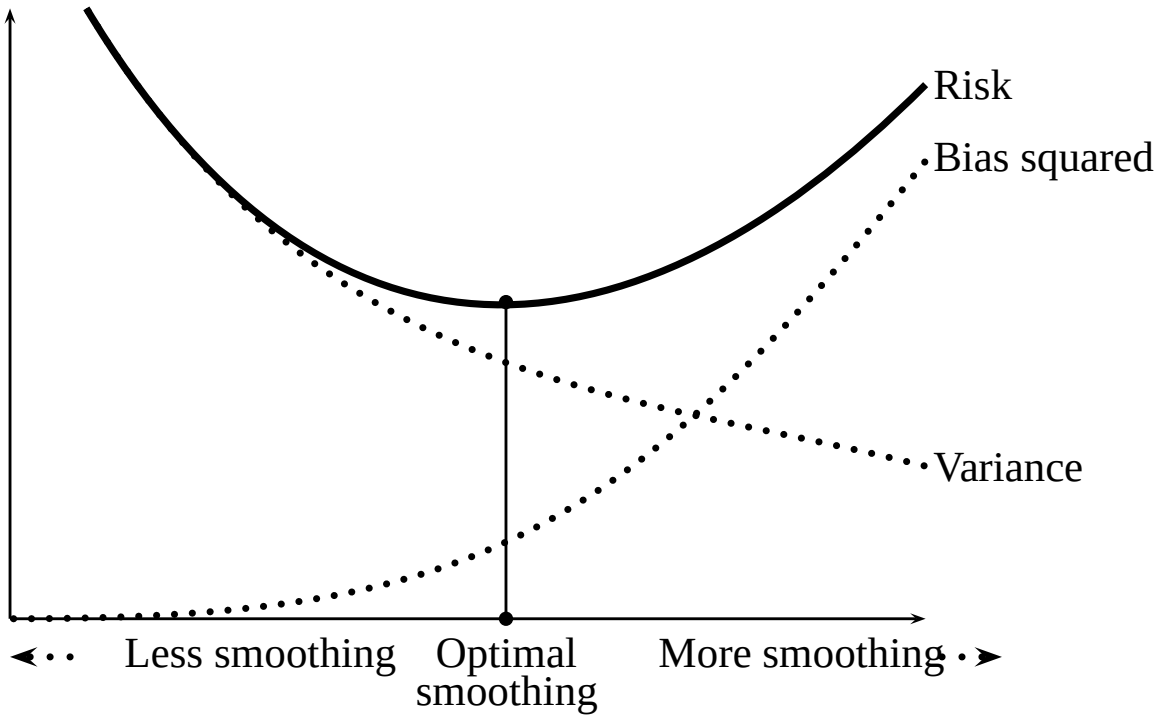

Figure 2: The bias–variance tradeoff. The bias increases and the variance decreases with the amount of smoothing. The optimal amount of smoothing, indicated by the vertical line, minimizes the risk = bias^2 + variance.

We can also derive the optimal bandwidth and the optimal rate "intuitively" by balancing the bias and variance terms as in Figure 2:

Notes: The trick is to balance the terms against each other instead of taking derivatives.

In (3) we see that the bias term increases at a rate of h^4, while the variance decreases at a rate of $1/(nh)$. We would like each of these to go to 0 as $n \to \infty$, so we should choose h as a function of n that makes them go to zero at the same rate.

These rates match when $h^4 \sim 1/(nh)$, ignoring all the constants.

With algebra, we see the rates match when $h \sim 1/n^{1/5}$, so the optimal bandwidth $h_* = \mathcal{O}(n^{-1/5})$ as we derived.

Plugging that back in, we get

$$\begin{aligned} R(h_*) &= \sigma^2 + \mathcal{O}(h_*^4) + \mathcal{O}([nh_*]^{-1}) \\ &= \sigma^2 + \mathcal{O}(n^{-4/5}) + \mathcal{O}\left(\frac{n^{1/5}}{n}\right) \\ &= \sigma^2 + \mathcal{O}(n^{-4/5}), \end{aligned}$$

as we said.

In (most) parametric models, the risk of the maximum likelihood estimator decreases to 0 at rate $1/n$. The slower rate $n^{-4/5}$ is the price of using nonparametric methods.

In practice, we cannot use the bandwidth given in (4) since h_* depends on the unknown function r. Instead, we use *cross-validation* as described in earlier lectures. To summarize (when to use parametric versus nonparametric models):

Notes: If you believe the model is correct, parameteric convergence rate is faster, meaning the risk will decrease faster as the sample size increases.

If there is any reason to suspect the assumptions of the parametric model, it might be worth the slower convergence in order to remove the bias. The fit of a bad parametric model converges rapidly to a wrong answer. (The bias doesn't go away.)

Multivariate Extension. The Curse of Dimensionality

Reading: Shalizi section 8.3.

In multiple dimensions, say, each $X_i \in \mathbb{R}^p$, we can easily use multivariate kernels: we just replace $X_i - x$ in the kernel argument by $\|X_i - x\|_2$, so that the multivariate kernel regression estimator becomes

$$\hat{r}(x) = \frac{\sum_{i=1}^n K\left(\frac{\|x_i - x\|_2}{h}\right) \cdot Y_i}{\sum_{i=1}^n K\left(\frac{\|x_i - x\|_2}{h}\right)}.$$

This is called an *isotropic* kernel because the same bandwidth is used in each dimension. Generally, one uses a different bandwidth in each dimension. (See Shalizi section 4.3.)

The same calculations as those that went into deriving the bias and variance bounds above can be done in this multivariate case. With some intuitive reasoning we can also figure out the order of the bias and variance terms:

Notes: The squared bias is $\text{Bias}(\hat{r}(x))^2 = \mathcal{O}(h^4)$.

For variance, only data points falling into a sphere with volume $\mathcal{O}(h^p)$ around x can contribute to the prediction at x. Hence

$$\text{Var}(\hat{r}(x)) = \mathcal{O}\left(\frac{1}{nh^p}\right).$$

Why is the variance so strongly affected by the dimension p? What is the optimal bandwidth h and the optimal rate of the kernel smoother in p dimensions?

Notes: In one dimension, the number of data points that lie in an interval of length h around x has mean about $nhf(x)$. In p dimensions, the analog is the hypersphere around x with radius h, and that has volume proportional to h^p. The number of points in that sphere is proportional to $nh^p f(x)$. So to get the

same number of expected points as in 1 dimension, we need the bandwidth to be $h^{1/p}$.

To find the rates, let's balance the bias and variance terms again. Ignoring constants, setting h^4 to be of the same order as $1/(nh^p)$ leads to the optimal smoothing rate $h_* = \mathcal{O}(n^{-1/(4+p)})$.

If we insert that into the risk, we obtain

$$\begin{aligned} R(h_*) &= \sigma^2 + \mathcal{O}(h_*^4) + \mathcal{O}\left(\frac{1}{nh_*^p}\right) \\ &= \sigma^2 + \mathcal{O}(n^{-4/(p+4)}). \end{aligned}$$

This is the optimal rate.

Shalizi (Sec 8.3): *"For $p = 1$, the nonparametric rate is $O(n^{-4/5})$, which is of course slower than $O(n^{-1})$, but not all that much, and the improved bias usually more than makes up for it. But as p grows, the nonparametric rate gets slower and slower, and the fully nonparametric estimate more and more imprecise, yielding the infamous* curse of dimensionality. *For $p = 100$, say, we get a rate of $O(n^{-1/26})$, which is not very good at all. [....] Said another way, to get the same precision with p inputs that n data points gives us with one input takes $n^{(4+p)/5}$ data points. For $p = 100$, this is $n^{20.8}$, which tells us that matching the error of $n = 100$ one-dimensional observations requires $O(4 \times 10^{41})$ hundred-dimensional observations."*

Notes: Let's do the example.

Suppose we want to get the a certain predictive performance δ, as measured by the MSE.

If we need N data points to get this MSE in 1 dimension ($p = 1$), then we need $N^{(4+p)/5}$ in p dimensions.

For example, suppose we needed $N = 100$ when $p = 1$. Then if $p = 100$, to get the same MSE would require $100^{104/5} = 4 \times 10^{41}$ observations.

In Lecture 8 we will see an alternative extension to higher dimensions that doesn't nearly suffer the same variance; this is called an *additive model*.