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

Poll.

Which of these regression models are nonparametric? Check all.

A. linear regression
B. regression with third-order polynomials
C. k nearest-neighbor regression with k chosen by 5-fold cross-validation
D. regression splines with 30 knots
E. smoothing splines with tuning parameter λ chosen by GCV

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 $x \to \infty$.

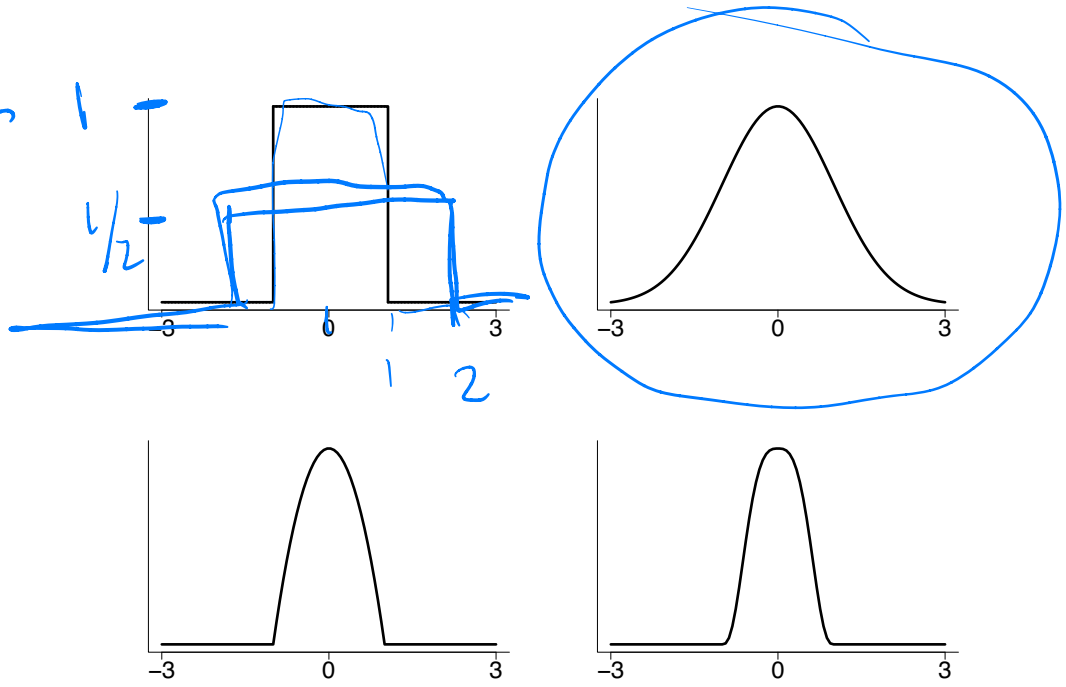


Figure 1: Examples of kernels: boxcar (top left), Gaussian (top right), Epanechnikov (bottom left), and tricube (bottom right).

Some commonly used kernels are the following:

$$\text{the boxcar kernel:} \quad K(x) = \frac{1}{2} I(x),$$

$$\text{the Gaussian kernel:} \quad K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2},$$

$$\text{the Epanechnikov kernel:} \quad K(x) = \frac{3}{4}(1 - x^2) I(x)$$

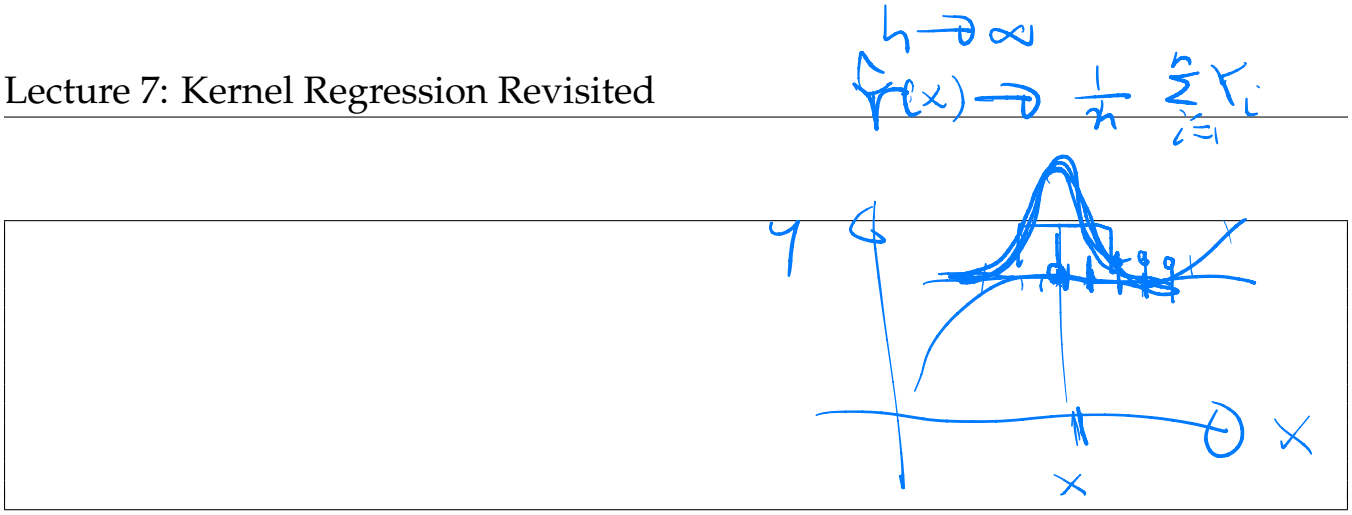
$$\text{the tricube kernel:} \quad K(x) = \frac{70}{81}(1 - |x|^3)^3 I(x)$$

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*. A rescaled kernel with bandwidth h looks like this:



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

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

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?

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...

Poll.

We fit a kernel smoother with bandwidth h to a dataset with n observations. Define the estimation bias and variance as

$$\begin{aligned} \text{estimation bias} &= \mathbb{E}\left[\mathbb{E}[\hat{r}(X) - r(X) \mid X]\right] \\ \text{estimation variance} &= \mathbb{E}\left[\text{Var}(\hat{r}(X) \mid X)\right], \end{aligned}$$

where the outer expectations average over the distribution of X.

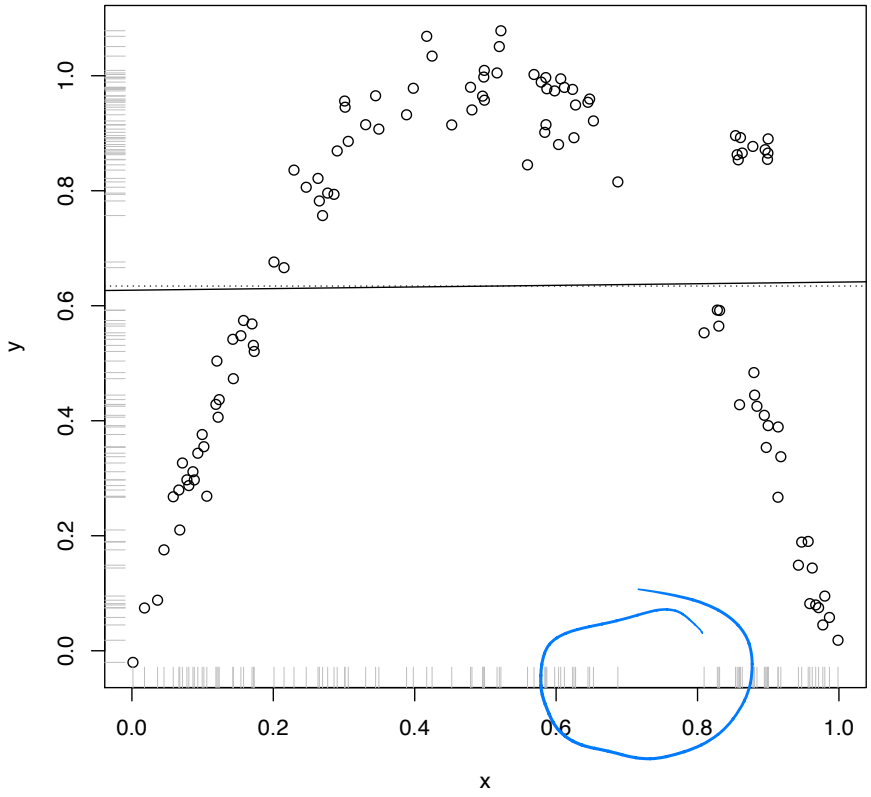
What happens to these quantities in the limit as h increases toward infinity?

A. The bias decreases, because the model is more flexible; the variance increases, because it averages over less data

B. The bias increases, because the model is less flexible; the variance decreases, because it averages over more data

C. The bias increases, because the model is less flexible; the variance increases, because it averages over more random data

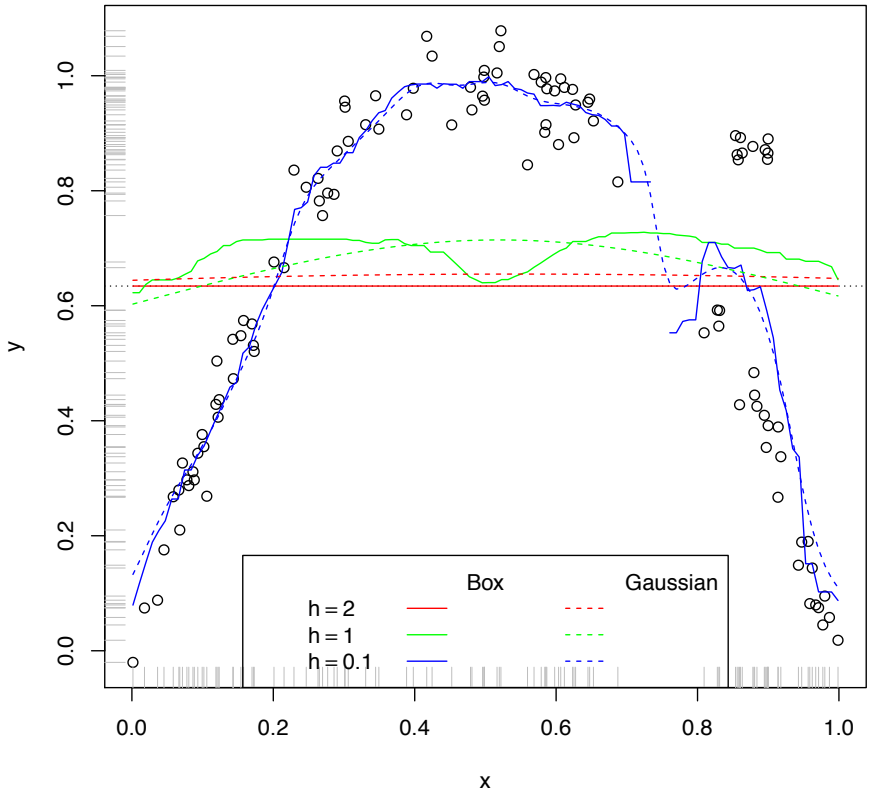
D. The bias does not change, since the model is the same; the variance decreases, because it averages over a larger sample




```
plot(all.x, all.y, xlab = "x", ylab = "y")
rug(all.x, side = 1, col = "grey")
rug(all.y, side = 2, col = "grey")
abline(h = mean(all.y), lty = "dotted")
fit.all = lm(all.y ~ all.x)
abline(fit.all)
```

Figure 1.4 Data from Figure 1.1, with a horizontal line at the mean (dotted) and the ordinary least squares regression line (solid).

As I just said, the sample mean is a special case; see Exercise 1.7. Ordinary linear regression is another special case, where $\widehat{w}(x_i, x)$ is given by Eq. 1.52. Both of these, as remarked earlier, ignore how far x_i is from x. Let us look at some linear smoothers which are not so silly.




```
lines(ksmooth(all.x, all.y, "box", bandwidth = 2), col = "red")
lines(ksmooth(all.x, all.y, "box", bandwidth = 1), col = "green")
lines(ksmooth(all.x, all.y, "box", bandwidth = 0.1), col = "blue")
lines(ksmooth(all.x, all.y, "normal", bandwidth = 2), col = "red", lty = "dashed")
lines(ksmooth(all.x, all.y, "normal", bandwidth = 1), col = "green", lty = "dashed")
lines(ksmooth(all.x, all.y, "normal", bandwidth = 0.1), col = "blue", lty = "dashed")
legend("bottom", ncol = 3, legend = c("", expression(h == 2), expression(h ==
    1), expression(h == 0.1), "Box", "", "", "", "Gaussian", "", "", ""), lty = c("blank",
    "blank", "blank", "blank", "blank", "blank", "solid", "solid", "solid", "blank",
    "dashed", "dashed", "dashed"), col = c("black", "black", "black", "black",
    "black", "red", "green", "blue", "black", "red", "green", "blue"), pch = NA)
```

Figure 1.6 Data from Figure 1.1 together with kernel regression lines, for various combinations of kernel (box/uniform or Gaussian) and bandwidth. Note the abrupt jump around $x = 0.75$ in the $h = 0.1$ box-kernel (solid blue) line — with a small bandwidth the box kernel is unable to interpolate smoothly across the break in the training data, while the Gaussian kernel (dashed blue) can.

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,

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

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?

squared error would always be σ^2, the noise variance. However, our estimate would be unbiased.

Smoothing methods try to use multiple measurements at points x_i which are *near* the point of interest x. If the regression function is smooth, as we're assuming it is, $\mu(x_i)$ will be close to $\mu(x)$. Remember that the mean-squared error is the sum of bias (squared) and variance. Averaging values at $x_i \neq x$ is going to introduce bias, but averaging independent terms together also reduces variance. If smoothing gets rid of more variance than it adds bias, we come out ahead.

Here's a little math to see it. Let's assume that we can do a first-order Taylor expansion (Figure B.1), so

$$\mu(x_i) \approx \mu(x) + (x_i - x)\mu'(x) \tag{4.3}$$

and

$$y_i \approx \mu(x) + (x_i - x)\mu'(x) + \epsilon_i \tag{4.4}$$

Now we average: to keep the notation simple, abbreviate the weight $w(x_i, x, h)$ by just w_i.

$$\widehat{\mu}(x) = \sum_{i=1}^{n} y_i w_i \tag{4.5}$$

$$= \sum_{i=1}^{n} (\mu(x) + (x_i - x)\mu'(x) + \epsilon_i) w_i \tag{4.6}$$

$$= \mu(x) + \sum_{i=1}^{n} w_i \epsilon_i + \mu'(x) \sum_{i=1}^{n} w_i (x_i - x) \tag{4.7}$$

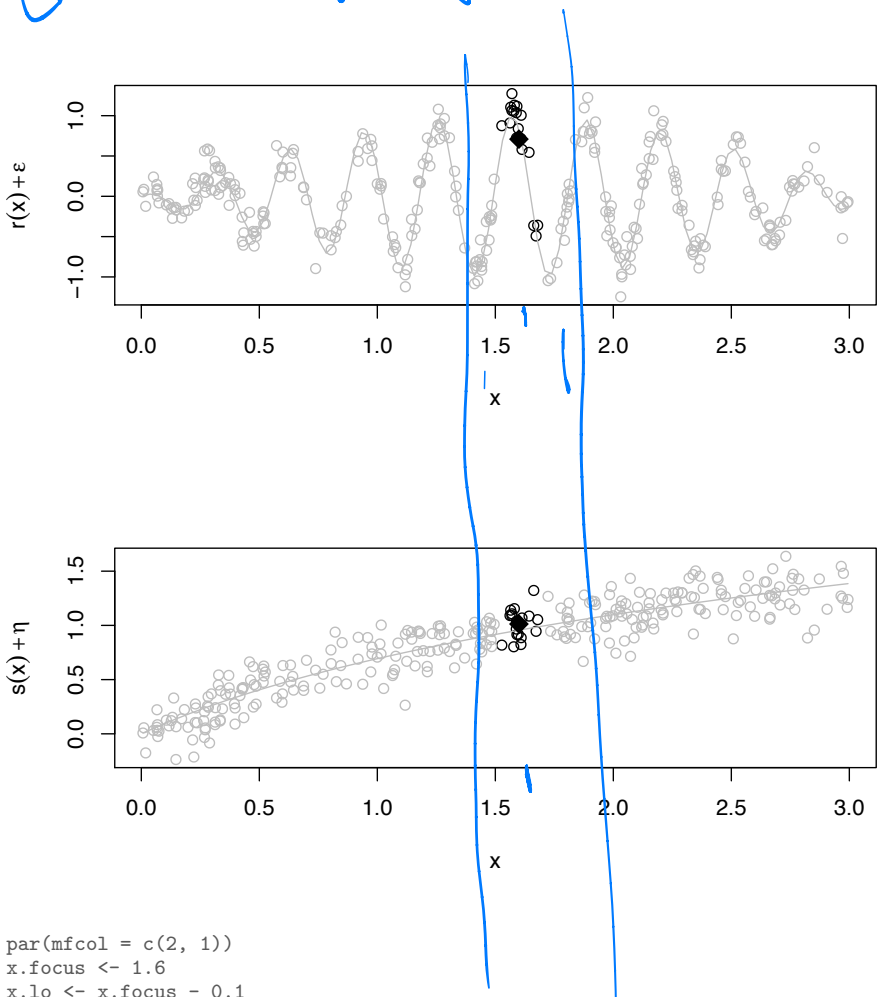
$$\widehat{\mu}(x) - \mu(x) = \sum_{i=1}^{n} w_i \epsilon_i + \mu'(x) \sum_{i=1}^{n} w_i (x_i - x) \tag{4.8}$$

$$\mathbb{E}\left[(\widehat{\mu}(x) - \mu(x))^2\right] = \sigma^2 \sum_{i=1}^{n} w_i^2 + \mathbb{E}\left[\left(\mu'(x) \sum_{i=1}^{n} w_i (x_i - x)\right)^2\right] \tag{4.9}$$

(Remember that: $\sum w_i = 1$; $\mathbb{E}[\epsilon_i] = 0$; ϵ is uncorrelated with everything; and $\mathbb{V}[\epsilon_i] = \sigma^2$.)

The first term on the final right-hand side is an estimation variance, which will tend to shrink as n grows. (If we just did a simple global mean, $w_i = 1/n$ for all i, so we'd get σ^2/n, just like in baby stats.) The second term, an expectation, is bias, which grows as x_i gets further from x, and as the magnitudes of the derivatives grow, i.e., this term's growth varies with *how* smooth or wiggly the regression function is. For smoothing to work, w_i had better shrink as $x_i - x$ and $\mu'(x)$ grow.[4] Finally, all else being equal, w_i should also shrink with n, so that the over-all size of the sum shrinks as we get more data.

[4] The higher derivatives of μ also matter, since we should really keep more than just the first term in the Taylor expansion. The details get messy, but Eq. 4.12 below gives the upshot for kernel smoothing.




```
par(mfcol = c(2, 1))
x.focus <- 1.6
x.lo <- x.focus - 0.1
x.hi <- x.focus + 0.1
colors = ifelse((x < x.hi) & (x > x.lo), "black", "grey")
plot(x, yr, xlab = "x", ylab = expression(r(x) + epsilon), col = colors)
curve(true.r(x), col = "grey", add = TRUE)
points(x.focus, mean(yr[(x < x.hi) & (x > x.lo)]), pch = 18, cex = 2)
plot(x, ys, xlab = "x", ylab = expression(s(x) + eta), col = colors)
curve(true.s(x), col = "grey", add = TRUE)
points(x.focus, mean(ys[(x < x.hi) & (x > x.lo)]), pch = 18, cex = 2)
par(mfcol = c(1, 1))
```

Figure 4.5 Relationship between smoothing and function roughness. In both panels we estimate the value of the regression function at $x = 1.6$ by averaging observations where $1.5 < x_i < 1.7$ (black points, others are "ghosted" in grey). The location of the average in shown by the large black diamond. This works poorly for the rough function r in the upper panel (the bias is large), but much better for the smoother function in the lower panel (the bias is small).

① $\text{bias}(\hat{r}(x)) = O(h^2) \to 0$ as $h \to 0$

② $\text{var}(\hat{r}(x)) = O\left(\frac{1}{nh}\right)$ — $n_{\text{eff}} \sim nh$

Makes sense as the # effective neighbors in train data scales as h

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

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

integrated MSE (prediction error)

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

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

$$R(h) = \sigma^2 + O(h^4) + O\left(\frac{1}{nh}\right)$$

irreducible error; integrated squared bias; integrated variance

$\left(\frac{1}{n^{1/5}}\right)^4 = \frac{1}{n} \cdots \frac{1}{h^{1/5}}$

Find optimal h^* that minimizes $R(h)$

$h^* = O\left(\frac{1}{n^{1/5}}\right)$ $\quad R(h^*) = O\left(n^{-4/5}\right)$

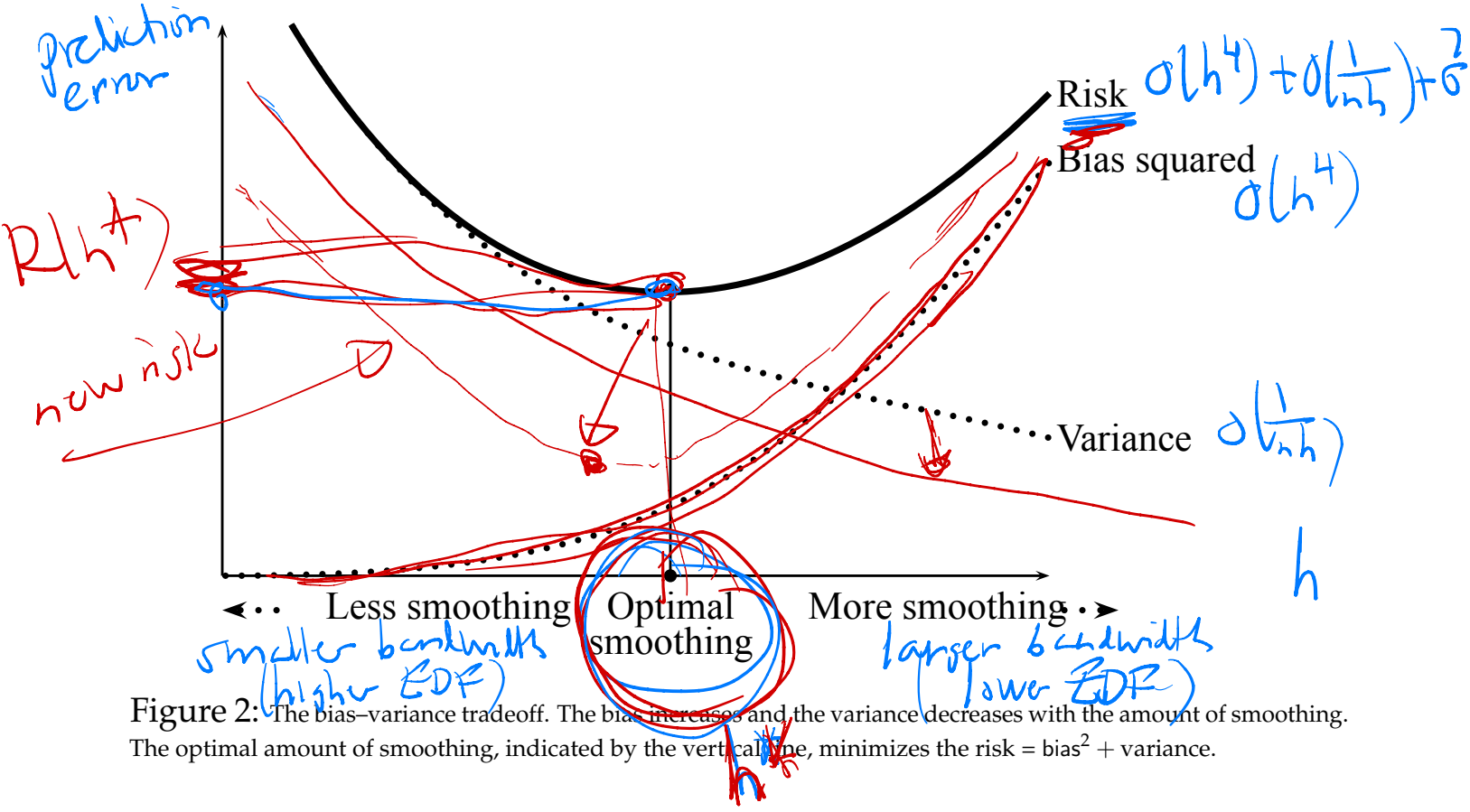


Figure 2: The bias–variance tradeoff. The bias increases and the variance decreases with the amount of smoothing. The optimal amount of smoothing, indicated by the vertical line, minimizes the risk = bias^2 + variance.

Discuss: Questions about Figure 2:

- How do these curves change as n increases?
- How does the optimal bandwidth h_* change as n increases?

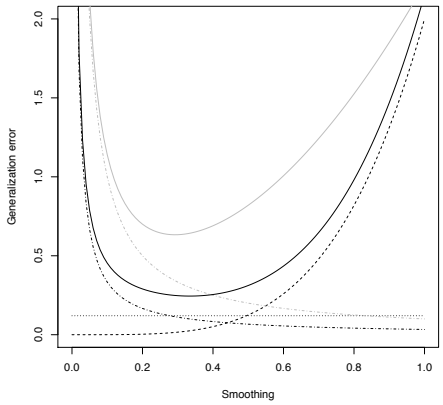
If we differentiate (3) and set the result equal to 0, we find that the *optimal bandwidth* h_* is

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

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.

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




```
curve(2 * x^4, from = 0, to = 1, lty = 2, xlab = "Smoothing", ylab = "Generalization error")
curve(0.12 + x - x, lty = 3, add = TRUE)
curve(1/(10 * x), lty = 4, add = TRUE, col = "grey")
curve(0.12 + 2 * x^2 + 1/(10 * x), add = TRUE, col = "grey")
curve(1/(30 * x), lty = 4, add = TRUE)
curve(0.12 + 2 * x^4 + 1/(30 * x), add = TRUE)
```

Figure 4.2 Consequences of adding more data to the components of error: noise (dotted) and bias (dashed) don't change, but the new variance curve (dotted and dashed, black) is to the left of the old (greyed), so the new over-all error curve (solid black) is lower, and has its minimum at a smaller amount of smoothing than the old (solid grey).

instead of knowing the derivatives at x_0 we have the values of the functions at a sequence of points $x_1, x_2, \ldots x_n$, we could use interpolation to fill out the rest of the curve. Quantitatively, however, r is less smooth than s — it changes much more rapidly, with many reversals of direction. For the same degree of accuracy in the interpolation r needs more, and more closely spaced, training points x_i than does s.

Now suppose that we don't get to actually get to see r and s, but rather just $r(x)+\epsilon$ and $s(x)+\eta$, for various x, where ϵ and η are noise. (To keep things simple I'll assume they're constant-variance, IID Gaussian noises, say with $\sigma = 0.15$.) The data now look something like Figure 4.4. Can we recover the curves?

As remarked in Chapter 1, if we had many measurements at the same x, then we could find the expectation value by averaging: the regression function $\mu(x) = \mathbb{E}[Y|X = x]$, so with multiple observations $x_i = x$, the mean of the corresponding y_i would (by the law of large numbers) converge on $\mu(x)$. Generally, however, we have at most one measurement per value of x, so simple averaging won't work. Even if we just confine ourselves to the x_i where we have observations, the mean-

Nice trick: Balance bias-variance w/o taking derivatives.

$$h^4 \sim \frac{1}{nh} \Rightarrow h^5 \sim \frac{1}{n} \Rightarrow h \sim \frac{1}{n^{1/5}}$$

bias variance

That is $h_* \asymp O(n^{-1/5})$ optimal smoothing

Plug into (*): $R(h_*) = \sigma^2 + O(n^{-4/5}) + O\left(\frac{n^{1/5}}{n}\right) \asymp O(n^{-4/5})$

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. parametric rate $O(1/n)$

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):

Pros/cons of nonparametric smoothing

If the parametric model is "correct" it will converge faster to the truth with rate $1/n$

However, if model is misspecified it will converge rapidly to the wrong answer

Nonparametric models are more flexible but with slower rates e.g. $n^{-4/5}$

Poll.

We fit a kernel smoother with a certain bandwidth h to a data set with n observations.

Suppose you increase your sample size with a factor of 10 but keep your bandwidth to be the same as before. How does the bias of your regression model change for these new larger data sets?

A. The bias decreases.

B. The bias increases.

C. The bias stays the same.

Poll.

We fit a kernel smoother with a certain bandwidth h to a data set with n observations.

Suppose you increase your sample size with a factor of 10 and now refit your regression model with cross-validation to minimize the prediction risk. How does the bias of your regression model change for these new larger data set?

A. The bias decreases.

B. The bias increases.

C. The bias stays the same.

Multivariate Extension. The Curse of Dimensionality

Reading: Shalizi section 8.3.

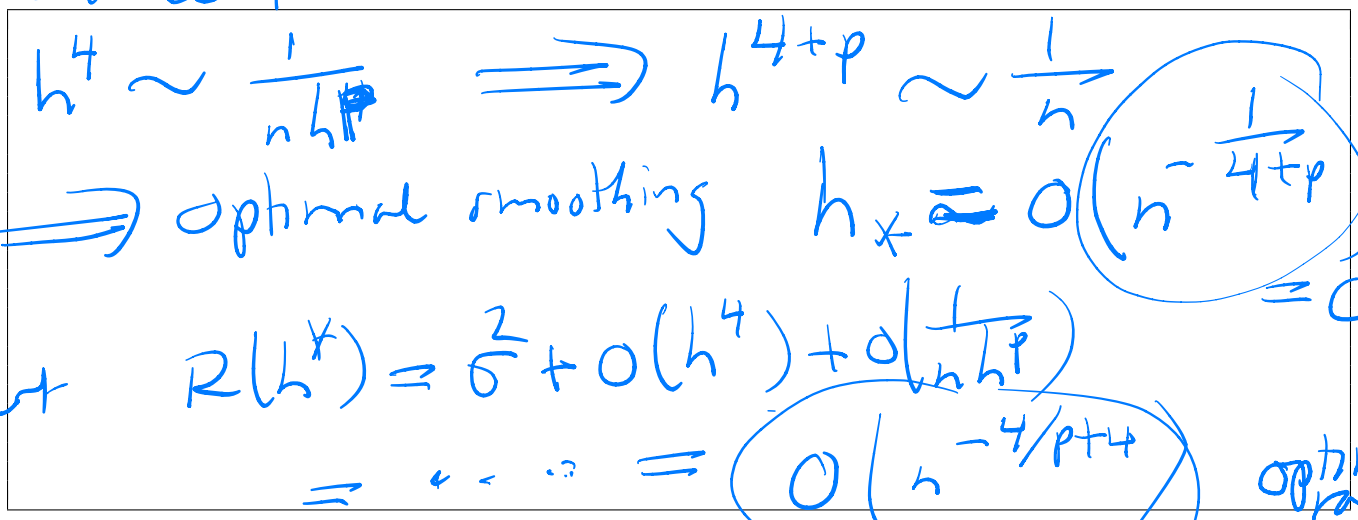
In multiple dimensions, say, each $X_i \in \mathbb{R}^p$, we can easily use multivariate kernels: we just replace $X_i - x$ in the kernel argument by $\|X_i - x\|_2$, so that the multivariate kernel regression estimator becomes

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

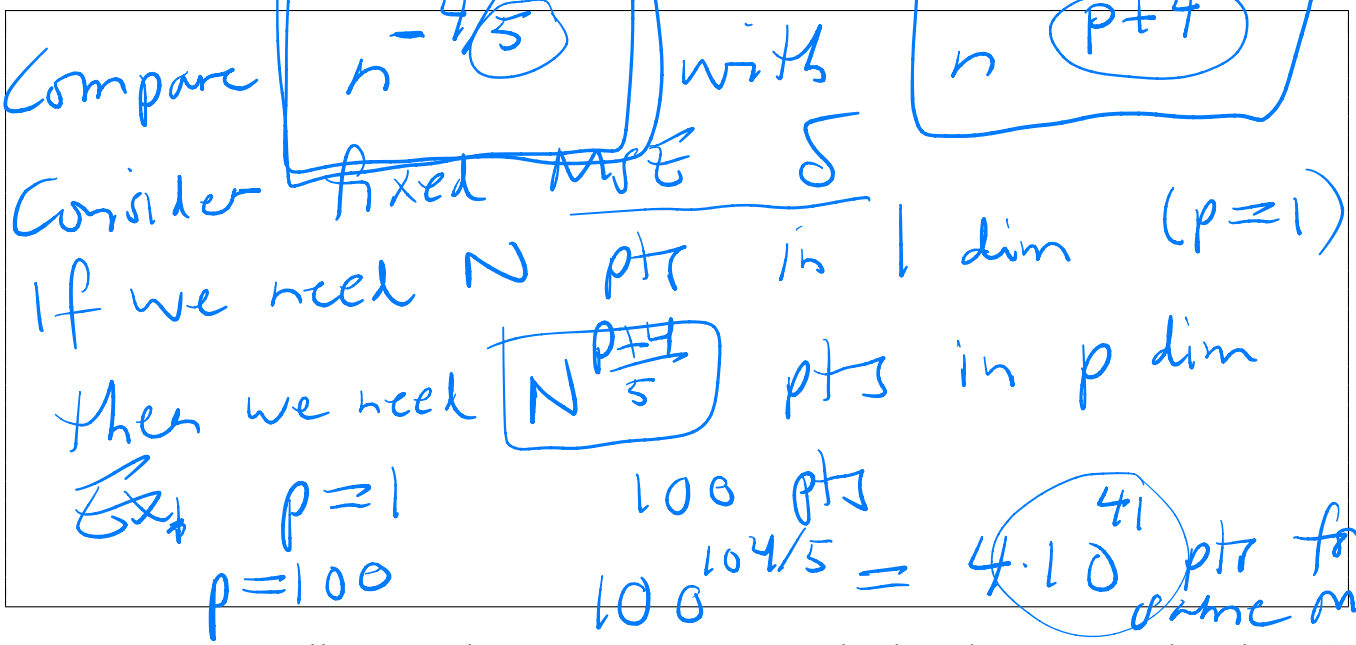
This is called an *isotropic* kernel because the same bandwidth is used in each dimension. Generally, one uses a different bandwidth in each dimension. (See Shalizi section 4.3.)

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:

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?



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."*

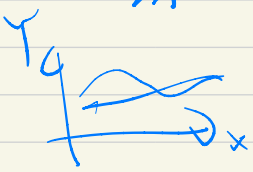


In Lecture 8 we will see an alternative extension to higher dimensions that doesn't nearly suffer the same variance; this is called an *additive model*.

$$Y = r(X_1, \cdots X_p) + \varepsilon \qquad R_{\text{苦}} = O\left(n^{-\frac{4}{p+4}}\right)$$

cf parametric rate $1/n$

Additive models



Assume

$$Y = \beta_0 + r_1(X_1) + \cdots + r_p(X_p) + \varepsilon$$

univariate reg.
e.g. 1d smoothing splines

36-402 DA Exam 1

YOUR NAME HERE (your andrewid)

3/19/2021

INSTRUCTIONS – REMOVE BEFORE SUBMITTING

This is a template for your data analysis report. It should work for anyone who is using R Markdown to generate PDFs through LaTeX. If you make the PDFs in some other way, as long as you can get 12 point fonts and reasonable margins, it will be fine.

Marking your answers

Each section below contains numbered questions. You are **required** to mark the sentence in your report that most closely answers each question.

For example, for question 2 in the EDA section, mark the sentence with **(2)**. For question 3 in Modeling & Diagnostics, mark the sentence with **(3)**.

You are writing a report, not simply writing answers to the questions, so you should **not** leave the questions in your report. Nor should your report consist only of bullet points or a numbered list of answers.

Figure captioning/sizing tips

Here is an example of creating a plot, setting its size (in inches), and giving it a caption. Figures with captions "float" on the page, usually appearing at the top or bottom; don't try to fight LaTeX and force them to appear in a certain place, because you will lose.

Notice that the figure appears in the PDF, but the code does not.

You can only make one figure per code chunk.

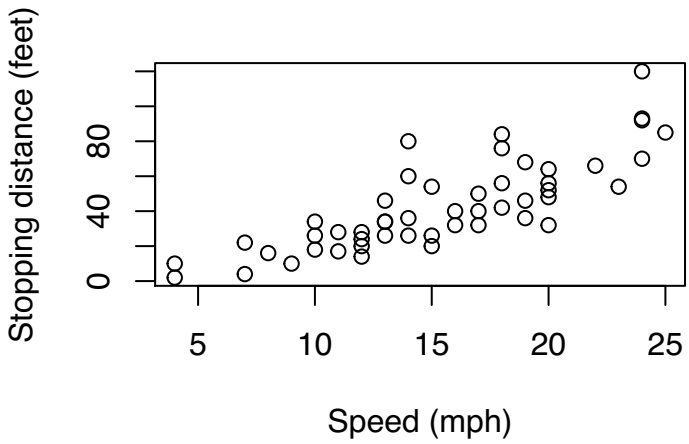


Figure 1: Distance it took cars at each speed to stop.

Introduction

1. Clearly state the research questions and objectives of your study.
2. State what data you are using and what information it contains.
3. Briefly mention your final findings. State them in terms that Beff Jezos can understand, rather than using technical jargon.

Exploratory Data Analysis

1. Create a new variable that divides `Metabolic.rate` by `Body.mass.g`. Call this `Metabolic.by.mass`. This is the amount of energy used per unit of body mass, and will allow comparisons between animals of different sizes. Use this variable instead of `Metabolic.rate` in the rest of your report.
2. Explore the key variables you need to answer the research questions. Describe them with any necessary univariate EDA, such as plots or summary statistics.
3. Identify and specify your response variable and its distribution.
4. Do multivariate EDA to explore the relationship between predictors and the response. Based on this and your univariate EDA, are there transformations you should use before modeling your data? After any transformations, do the relationships appear to be linear? If you decide to use transformations, use them for the rest of your analysis.
5. If there are plots that would help answer the research questions, by showing key relationships, show those plots.
6. Describe any trends or interesting features you see that suggest what you will find in

the analysis.

Modeling & Diagnostics

1. Construct a linear model to predict `Maximum.longevity.yrs` using `Metabolic.by.mass`. (If you decided to use transformations of either variable, use those when fitting your model.)
2. Because you're working for a billionaire, you'd like to show off a more sophisticated and flexible model. Use a smoothing spline to fit to the same data. Fit five models, setting the `df` (effective degrees of freedom) to be 3, 4, . . . , 7.
3. Use cross-validation to determine which of the models fit best to the data, in terms of prediction error. Your comparison should include the linear model and the spline models. (You can use any type of cross-validation you think is suitable.) State the error you estimated for each model and state which model you decided is best.
4. Present model diagnostic plots for your selected model. Discuss whether the model appears to fit well, and describe any possible improvements to your model to address any violations of the model assumptions.
5. Comment on whether the difference between the models appears significant, based on the uncertainty in your estimates of the prediction error; a formal test is not required here.
6. For your chosen model, examine the residual diagnostics to determine what type of bootstrap would be appropriate for this data.

Results

1. Using the selected model, determine whether animals with lower metabolic rates have longer lifespans, as requested by Beff Jezos. You can use model coefficients, tests, or plots to answer this question.
2. Look up the crab-eating raccoon in the data and obtain its features. Calculate what metabolic rate it would have it its rate were 50% smaller. Use your best model to estimate the mean lifespan of an animal with those characteristics.
3. Use bootstrapping to make a 95% confidence interval for that quantity. Use the pivotal confidence interval and 1000 bootstrap iterations. Report the confidence interval.

Conclusions

1. Summarize your main findings about the relationship between metabolic rate and lifespan.
2. Explain whether Beff Jezos can conclude that reducing the crab-eating raccoon's metabolic rate by 50% would cause its lifespan to change, and if so, by how much.
3. Discuss any limitations to your analysis that might affect the conclusions, such as violations of assumptions or limitations in the data.

THINK

- What are the standard model assumptions in regression (with linear smoothers)?

- Why do we care?

- How do we check these assumptions?

- What do we do if they don't seem to hold?

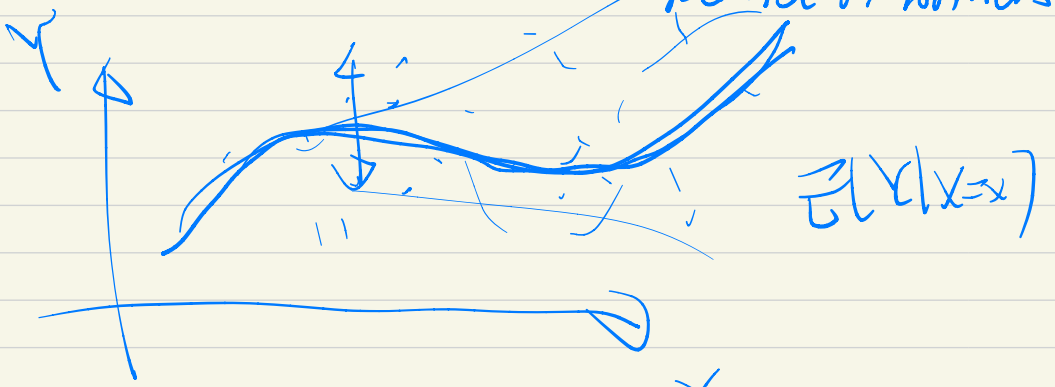
First some hints...

Review 401

mean resp

$$Y_i = r(X_i) + \varepsilon_i$$

(linear reg, smoothing splines, kernel smoothers)



Built-in assumptions (in R)

1) $E[\varepsilon | X=x] = 0$ no matter what X is
2) $\text{Var}(\varepsilon | X=x) = \sigma^2$ — " —
   $\varepsilon \sim N(0, \sigma^2)$
3)
4) ε_i's are independent across obsn. i

check assumptions

DRAFT of Homework 8

Advanced Methods for Data Analysis

March 31, 2021

/

1. **Abalone sizes.** Abalone are a type of sea snail often harvested and sold for food. The data file `abalonemt.csv` contains nine variables measured on 4173 abalones. Here is a description of the nine variables:

Name	Data Type	Units	Description
Sex	nominal	M, F, I (infant)	
Length	continuous	mm	
Diameter	continuous	mm	
Height	continuous	mm	
Whole.weight	continuous	grams	Entire abalone
Shucked.weight	continuous	grams	Weight of meat
Viscera.weight	continuous	grams	gut weight (after bleeding)
Shell.weight	continuous	grams	after being dried
Rings	integer		+1.5 gives the age in years

   A fisherperson who wishes to sell the abalone for food is interested in the edible part. Prior to cutting one open for cooking, one can predict the weight of the edible part (`Shucked.weight`) through a regression model. A natural predictor that would be available, if a scale were available, is `Whole.weight`.

   Assume that the only predictors available, when the prediction of `Shucked.weight` is needed, are `Diameter`, `Length`, and `Height`.

   Here is one hint: Objects of uniform density have weights proportional to their volume. What functions of the predictors might be good predictors of the volume of an abalone?

   As usual, start with an exploratory data analysis (EDA) of the response, `Shucked.weight`, and the three predictors: `Diameter`, `Length`, and `Height`.

   (a) We will explore linear and kernel regression models for predicting the response from some or all of the predictors. Whenever you need a bandwidth for a variable in a kernel regression, use the sample standard deviation of the variable divided by $n^{1/5}$, where n is the size of the sample.[1] This applies to the full data or to a cross-validation calculation.

   You should fit the following models:

[1] This is a simple heuristic way to set the bandwidth, instead of using cross-validation. Note that this bandwidth is proportional to n in the same way as the optimal bandwidth, though the constant factors need not match the optimal bandwidth.

Model 1: A linear regression of log(`Shucked.weight`), on the logarithms of all three predictors.

Model 2: A kernel regression of `Shucked.weight`, on all three predictors: `Diameter`, `Length`, and `Height`.

Note: When making predictions with Model 1, the `predict` function will use the predictors correctly, but it will predict the logarithm of the response. You will need to take `exp` of the predictions before comparing the predictions to `Shucked.weight` in the calculation of cross-validation error.

Once again, you will need to load the `np` package with `library(np)`. You need a different bandwidth for each predictor; see HW 4 problem 3(d) for a similar problem.

Draw a scatter plot of the fitted values for both models against each other, and comment on how similar or different the predictions seem to be.

Examine residuals from each model and say what you learn about the possible distributions of the noise terms, and comment on how well the usual regression assumptions seem to hold.

(b) Calculate the **training** error for both models. Which model fits better to the training data?

(c) Perform five-fold cross-validation to estimate the mean squared error, and compute a rough estimate of the standard errors. Based on this result and your residual analysis, which model appears to perform best?

(d) Calculate the effective degrees of freedom for each model. For Model 1, this should be easy; for Model 2, use a calculation similar to the one you used in Homework 7. The calculation may seem more complicated because you have three variables, each with their own bandwidth, rather than one variable. But in this case, the kernel is simply the product of three kernels, where each kernel is in this case a standard normal density function (each recaled by its own bandwidth h), since we're using the default Gaussian kernel. See Shalizi section 4.3 and equations 4.22–4.23.

According to Lecture 6,

$$R_{\text{in}} = \mathbb{E}[R_{\text{training}}] + 2\sigma^2 \text{edf}/n.$$

Use the model with the smallest cross-validation error (that is, the smallest prediction risk according to CV) to estimate σ^2 by dividing the training error (or RSS) for that model by n. If your best model is Model 1, remember that it is on the log scale and you must exponentiate.

Then use the training error, edf, and $\hat{\sigma}^2$ to estimate R_{in} for each model. If you use this method to pick the best model (with the smallest estimate of R_{in}), would your result match the model picked by minimizing the cross-validation error?

(e) Under the assumption that $\hat{\beta}$ from the linear model is multivariate normal, compute a 95% confidence interval for $\beta_{\text{log-Diameter}}$. Does it include 0? Confirm your calculations with the `confint` command. Based on the EDA and residual analysis, comment on how reliable you think this confidence interval is.

Hint: You can retrieve an estimate of the variance-covariance matrix using the `vcov` command in `R`. For the question, think about what assumptions went into the

computation of this confidence interval, and whether it is reasonable to assume that $\hat{\beta}$ is multivariate normal.

(f) Now let's get a confidence interval for

$$\mathbb{E}\left[\log(Y) \mid \text{log-Diameter} = 5.93, \text{log-Length} = 6.10, \text{log-Height} = 6.24\right].$$

Using again the variance-covariance matrix for $\hat{\beta}$, provide a 95% confidence interval under the assumption that $\hat{\beta}$ is distributed as a multivariate normal. Confirm your calculations with the `predict` function and `interval = "confidence"` argument.

x^* evaluation pt

Ref: see 401 Part 7 p.33

$\tilde{\mu}^* + t_{\alpha/2, n-q} \sqrt{x^{*T} \tilde{\sigma}^2 (X^T X)^{-1} x^*}$

check!

$\tilde{\sigma}^2 (X^T X)^{-1}$ = cov($\hat{\beta}$) in R: vcov(.)

computation of this confidence interval, and whether it is reasonable to assume that $\hat{\beta}$ is multivariate normal.

(f) Now let's get a confidence interval for

$$\mathbb{E}\left[\log(Y) \mid \text{log-Diameter} = 5.93, \text{log-Length} = 6.10, \text{log-Height} = 6.24\right].$$

Using again the variance-covariance matrix for $\hat{\beta}$, provide a 95% confidence interval under the assumption that $\hat{\beta}$ is distributed as a multivariate normal. Confirm your calculations with the `predict` function and `interval = "confidence"` argument.