S&DS 365 / 665

Intermediate Machine Learning

Smoothing and Density Estimation

September 12

Yale

Topics for today

- Recap: Smoothing kernels
- Kernel density estimation
- Bias-variance decomposition
- Intro to Mercer kernels

Some reminders

- Quiz 1: Great job!
- Assn 1 posted on Wednesday
- Topics: Lasso, smoothing, Mercer kernels, some neural nets
- Questions?

Notes

- Notes posted to course page
  `http://interml.ydata123.org`
- Readings from “Probabilistic Machine Learning: An Introduction”
- `https://probml.github.io/pml-book/book1.html`

Nonparametric Regression

Given $(X_1, Y_1), \ldots, (X_n, Y_n)$ predict Y from X.

Assume only that $Y_i = m(X_i) + \epsilon_i$ where where $m(x)$ is a smooth function of x.

The most popular methods are *kernel methods*. However, there are two types of kernels:

1. Smoothing kernels
2. Mercer kernels

Smoothing kernels involve local averaging.
Mercer kernels involve regularization.

Smoothing Kernels

- Smoothing kernel estimator:

$$\widehat{m}_h(x) = \frac{\sum_{i=1}^n Y_i\, K_h(X_i, x)}{\sum_{i=1}^n K_h(X_i, x)}$$

where $K_h(x, z)$ is a *kernel* such as

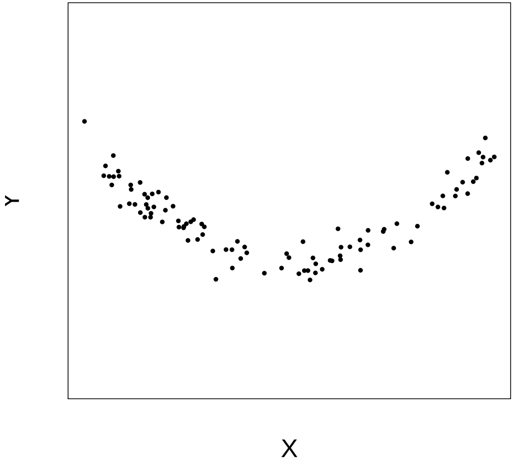
$$K_h(x, z) = \exp\left(-\frac{\|x - z\|^2}{2h^2}\right)$$

and $h > 0$ is called the *bandwidth*.

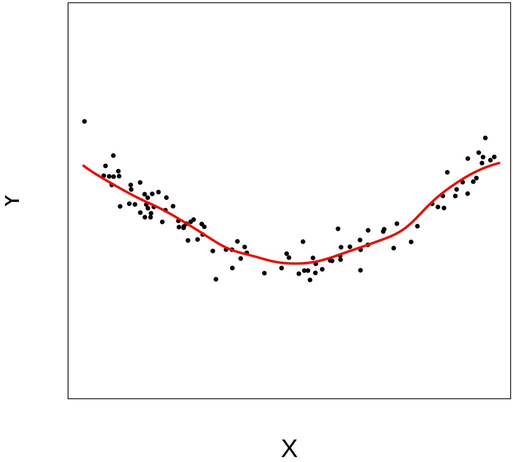
- $\widehat{m}_h(x)$ is just a local average of the Y_i's near x.

- The bandwidth h controls the bias-variance tradeoff: *Small h = large variance* while *large h = large bias*.

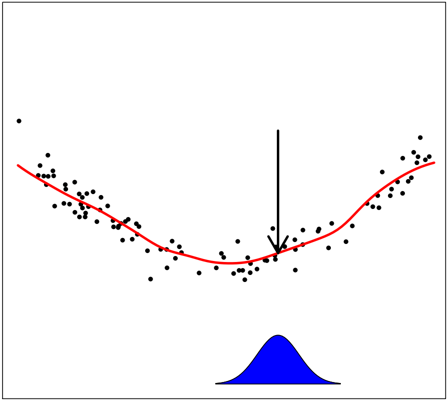
Example: Some Data – Plot of Y_i versus X_i



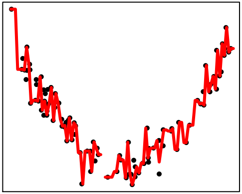
Example: $\widehat{m}(x)$



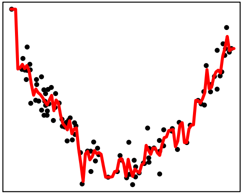
$\widehat{m}(x)$ is a local average



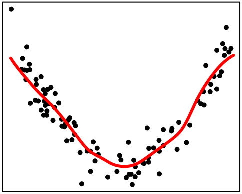
Effect of the bandwidth h



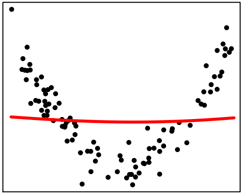
very small bandwidth



small bandwidth



medium bandwidth



large bandwidth

Let’s revisit the notebook

Smoothing Kernels

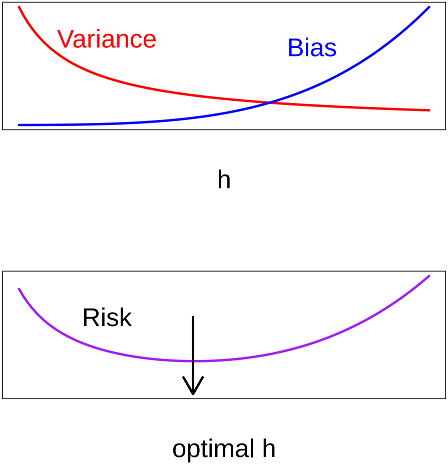
Risk $= \mathbb{E}(Y - \widehat{m}_h(X))^2 = \text{bias}^2 + \text{variance} + \sigma^2$.

$\sigma^2 = \mathbb{E}(Y - m(X))^2$ is the unavoidable prediction error.

small h: low bias, high variance (undersmoothing)

large h: high bias, low variance (oversmoothing)

Risk Versus Bandwidth



Estimating the Risk: Cross-Validation

To choose h we need to estimate the risk $R(h)$. We can estimate the risk by using *cross-validation*.

1. Omit (X_i, Y_i) to get $\widehat{m}_{h,(i)}$, then predict: $\widehat{Y}_{(i)} = \widehat{m}_{h,(i)}(X_i)$.
2. Repeat this for all observations.
3. The cross-validation estimate of risk is:

$$\widehat{R}(h) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_{(i)})^2.$$

Shortcut formula: Whenever $\widehat{Y} = LY$ we can use the shortcut

$$\widehat{R}(h) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \widehat{Y}_i}{1 - L_{ii}}\right)^2.$$

In this case $L_{ii} = K_h(X_i, X_i)/\sum_t K_h(X_i, X_t)$.

Shortcut formula

Let's prove the shortcut formula. Let $K_{ij} = K_h(X_i, X_j)$. We have

$$
\begin{aligned}
\widehat{Y}_{(i)} &= \frac{\sum_{j \neq i} K_{ij} Y_j}{\sum_{j \neq i} K_{ij}} \\
&= \frac{\sum_j K_{ij} Y_j - K_{ii} Y_i}{\sum_j K_{ij} - K_{ii}} \\
&= \frac{\sum_j L_{ij} Y_j - L_{ii} Y_i}{1 - L_{ii}} \\
&= \frac{\widehat{Y}_i - L_{ii} Y_i}{1 - L_{ii}}
\end{aligned}
$$

To show this for OLS regression we can use the formula for the inverse of a matrix plus a rank-1 matrix.

Shortcut formula

It follows that

$$\left(Y_i - \widehat{Y}_{(i)}\right)^2 = \left(Y_i - \frac{\widehat{Y}_i - L_{ii} Y_i}{1 - L_{ii}}\right)^2$$

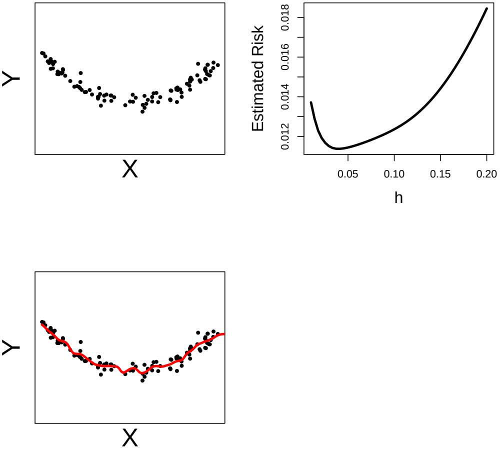
$$= \left(\frac{Y_i - \widehat{Y}_i}{1 - L_{ii}}\right)^2$$

To show this for OLS regression we can use the formula for the inverse of a matrix plus a rank-1 matrix.

Summary so far

1. Compute $\widehat{m}_h$ for each h
2. Estimate the risk $\widehat{R}(h)$ using LOOCV
3. Choose bandwidth $\widehat{h}$ to minimize $\widehat{R}(h)$
4. Let $\widehat{m}(x) = \widehat{m}_{\widehat{h}}(x)$

Example



Kernel density estimation

To estimate a density, use the same idea behind kernel smoothing:

$$\begin{aligned}\widehat{f}(x) &= \frac{1}{n}\sum_{i=1}^{n} K_h(X_i, x) \\ &= \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K\left(\frac{X_i - x}{h}\right)\end{aligned}$$

We require that $\int K(u)\,du = 1$ and $K \geq 0$ is symmetric around zero (an even function).

This places a “bump function” around each data point, and averages them (a mixture model)

Kernel density estimation

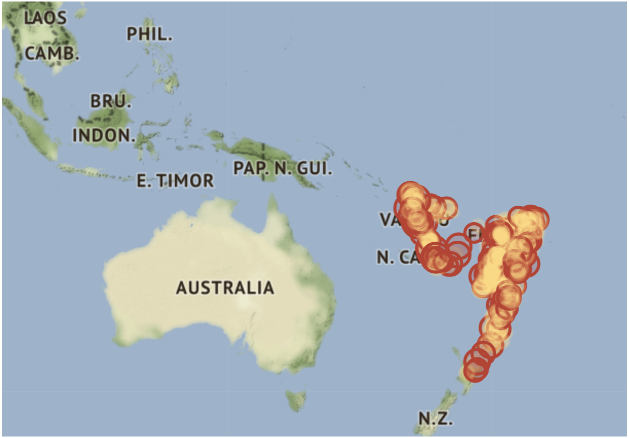
In p dimensions:

$$\widehat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(X_i, x)$$
$$= \frac{1}{n\,h^p}\sum_{i=1}^{n} K\left(\frac{X_i - x}{h}\right)$$

We require that $\int K(u)\,du = 1$ and K is symmetric around zero.

This places a “bump function” around each data point, and averages them (a mixture model)

KDE demo: Fiji earthquakes



Kernel density estimation

The bias-variance tradeoff:

$$\begin{aligned} \text{bias}^2(x) &\approx h^4 \\ \text{var}(x) &\approx \frac{1}{n\,h^p} \end{aligned}$$

Note that the variance scales according to the expected number of data points in a cube of side length h in p-dimensions.

We'll go through the calculation of this on the board. Notes are posted to `http://interml.ydata123.org`

Back to regression

Using a kernel density estimator, the "plug-in" regression estimate gives us back the kernel smoother:

$$\widehat{m}(x) = \int y\,\widehat{f}(y\,|\,x)\,dy$$

$$= \frac{\int y\,\widehat{f}(x,y)\,dy}{\widehat{f}(x)}$$

$$= \frac{\sum_i Y_i K_h(X_i,x)}{\sum_i K_h(X_i,x)}$$

Generative models

- A density estimate is a *generative model*
- We can sample from the density to “generate” a new data point
- What is an algorithm for sampling from the estimated distribution?

Generative models

1. Sample an index i uniformly from 1 to n
2. Sample a point x from a Gaussian with mean X_i and variance h^2

Generative models

Some recent generative models for images:

Generative models

Some recent generative models for images:

API RESEARCH BLOG ABOUT

DALL·E Now Available in Beta

We'll invite 1 million people from our waitlist over the coming weeks. Users can create with DALL·E using free credits that refill every month, and buy additional credits in 115-generation increments for $15.

JOIN DALL·E 2 WAITLIST ↗

July 20, 2022
3 minute read

Summary

- Smoothing methods compute local averages, weighting points by a kernel
- Shape of the kernel doesn't matter (much)
- KDE places a density around each data point, and averages
- The COD limits use of both approaches to low dimensions