Kernel Smoothing Methods

Peng Chen

School of Management
University of Science and Technology of China

October 29, 2020

Table of Contents

Table of Contents

Empirical distribution function

- Let $X_1, ..., X_n$ be independent identically distributed (i.i.d.) random variables with probability density funtion f and probability distribution function F respectively.
- The empirical distribution function of this data set is obtained by putting mass $\frac{1}{n}$ at each datum point.

$$\widehat{F_n}(x) = \frac{1}{n} \sum_{i=1}^{n} I(X_i \leq x).$$

- By the strong law of large numbers, we have

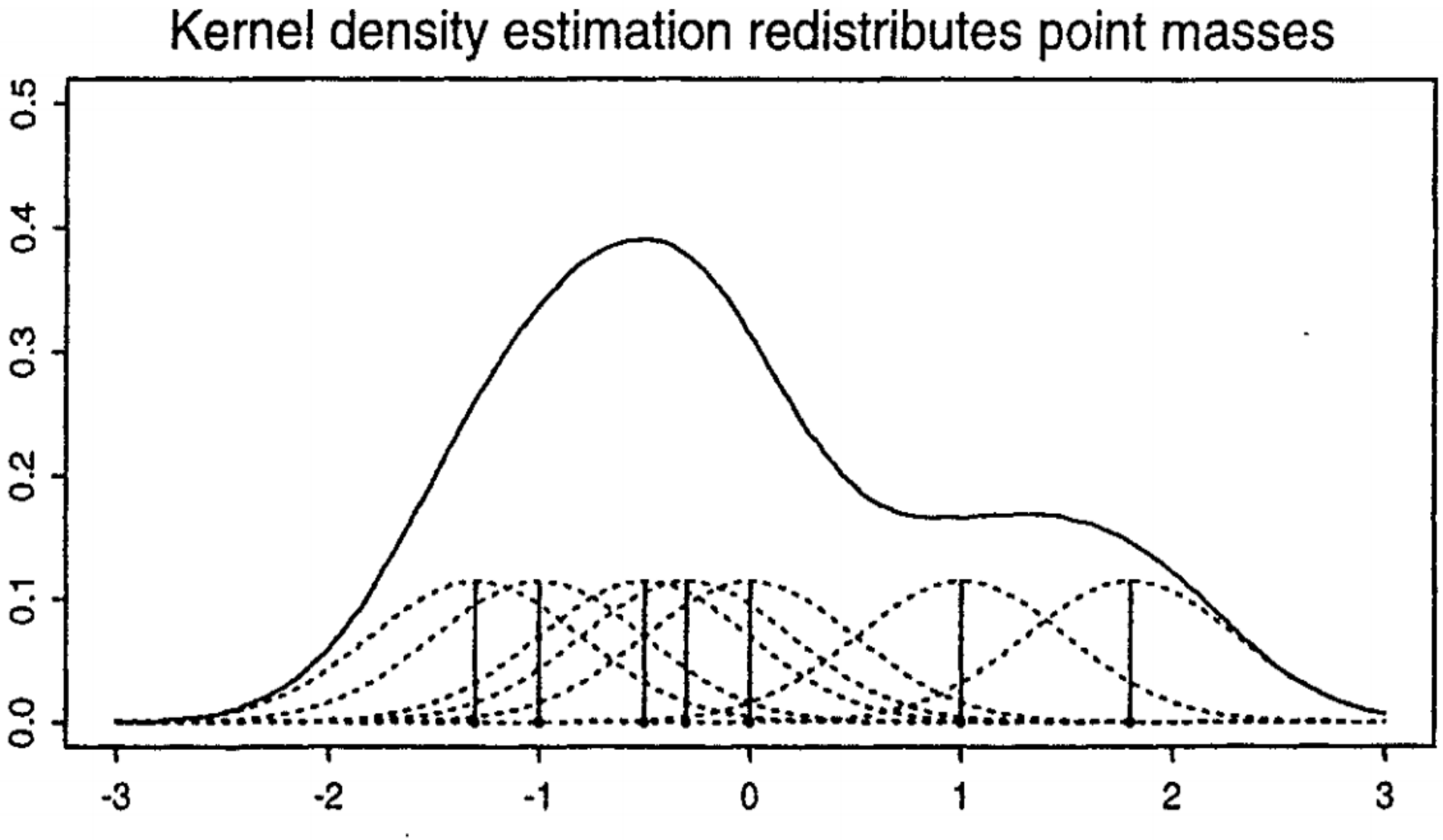
$$\widehat{F_n}(x) \to F(x), \forall x \in \mathbb{R},$$

almost surely as $n \to \infty$.

Kernel density estimation

- However, the data structure can hardly be examined via the plot of the function $\widehat{F_n}(x)$.
- A better visualization device is to attempt to plot its density function $\widehat{f_n}(x)$, but this function may not exist.
- An improved idea over the empirical distribution function is to smoothly redistribute the mass $\frac{1}{n}$ at each datum point to its vicinity.

Kernel density estimation



Kernel density estimation

- Because $f(x) = F'(x) = \lim_{h\to 0} \frac{F(x+h)-F(x-h)}{2h}$, So the plug-in estimation is

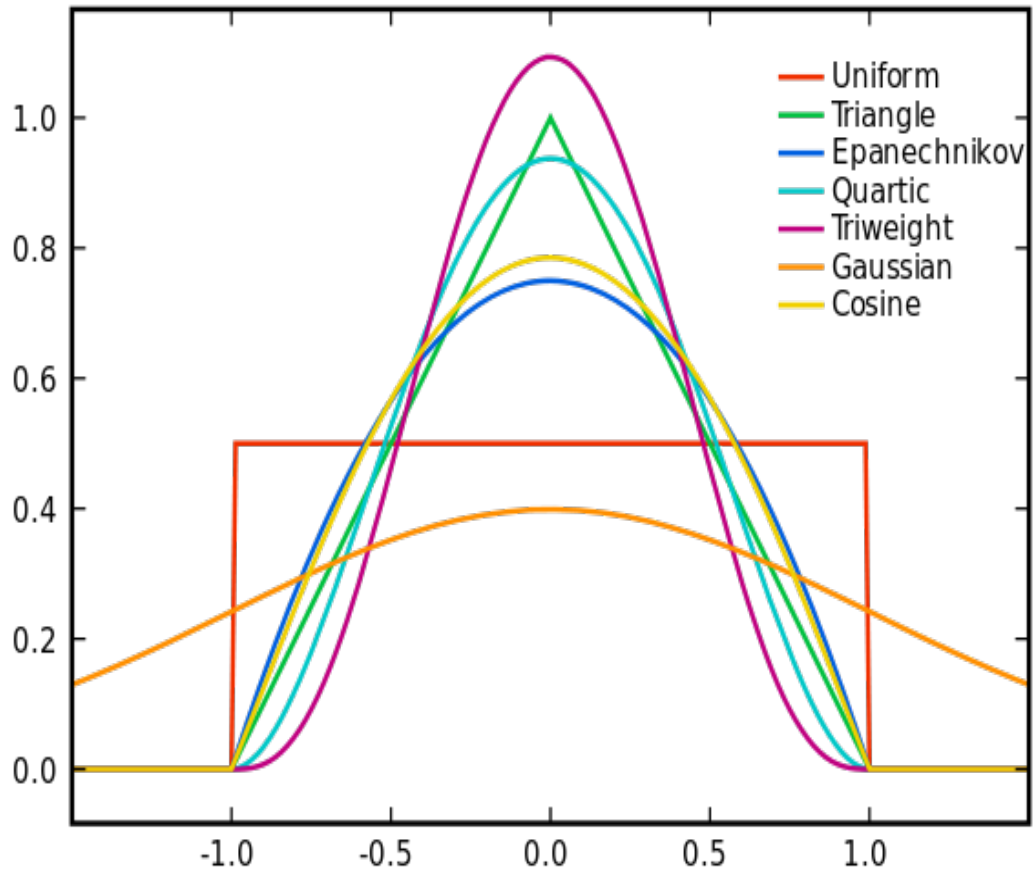
$$f_n(x) = \frac{F_n(x+h) - F_n(x-h)}{2h} = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2h} I(x-h < X_i \le x+h)$$

$$= \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h}\frac{1}{2} I(-1 < \frac{X_i - x}{h} \le 1) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K(\frac{X_i - x}{h})$$

  where h is a small number.
- Here $K(u) = \frac{1}{2} I(|u| \le 1)$ is a (equal) weight function, and is referred to as a (uniform) kernel function.

Kernel density estimation

- The uniform kernel (Equal weight) is not reasonable, here are some common kernels.



Kernel density estimation

Theorem

Suppose that X with density f and Y with distribution function G are independent. Then $X+Y$ has density $h(x)=\int f(x-y)dG(y)$.

- Letting ϕ_h denote the Gaussian density with mean zero and standard-deviation h,so the kernel density estimation has the form

$$\hat{f}_n(x)=\frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{X_i-x}{h})=\int \phi_h(x-y)d\hat{F}(y)=(\hat{F}\star\phi_h)(x),$$

the convolution of the sample empirical distribution with ϕ_h.

Kernel density estimation

- Generally, a kernel function $K(u)$ should satisfy the following conditions:

$$(1) K(u) \geq 0, K(u) = K(-u);$$

$$(2) \int K(u)du = 1;$$

$$(3) \int uK(u)du = 0;$$

$$(4) \int u^2 K(u)du = \kappa_{21} < \infty;$$

$$(5) \int K^2(u)du < \infty.$$

Properties of $\hat{f}_n(x)$

- $\hat{f}_n(x)$ itself is a density function.

$$\begin{aligned}\int \hat{f}_n(x)dx &= \frac{1}{nh}\sum_{i=1}^{n}\int_{-\infty}^{+\infty} K(\frac{X_i - x}{h})dx \\ &= \frac{1}{nh}\sum_{i=1}^{n}[-h\int_{+\infty}^{-\infty} K(u)du] = 1.\end{aligned}$$

- $\int x\hat{f}_n(x)dx = \frac{1}{n}\sum_{i=1}^{n} X_i.$
- $\int x^2\hat{f}_n(x)dx \to \frac{1}{n}\sum_{i=1}^{n} X_i^2$, as $h \to 0$.

Properties of $\hat{f}_n(x)$

$$
\begin{aligned}
E[\hat{f}_n(x)] &= E[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{X_i-x}{h})] = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}E[K(\frac{X_i-x}{h})] \\
&= \frac{1}{h}E[K(\frac{X_1-x}{h})] = \frac{1}{h}\int_{-\infty}^{+\infty}K(\frac{z-x}{h})f(z)dz \\
&= \int_{-\infty}^{+\infty}K(u)f(x+uh)du \\
&= \int_{-\infty}^{+\infty}K(u)[f(x)+f'(x)uh+\frac{1}{2}f^{(2)}(x)u^2h^2+o(h^2)]du \\
&= f(x)+\frac{1}{2}f^{(2)}(x)\kappa_{21}h^2+o(h^2).
\end{aligned}
$$

Properties of $\hat{f}_n(x)$

$$
\begin{aligned}
\mathsf{Var}[\hat{f}_n(x)] &= \mathsf{Var}[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{X_i-x}{h})] = \frac{1}{n^2}\sum_{i=1}^{n}\mathsf{Var}[\frac{1}{h}K(\frac{X_i-x}{h})] \\
&= \frac{1}{n}E[\frac{1}{h}K(\frac{X_1-x}{h})]^2 - \frac{1}{n}[E\{\frac{1}{h}K(\frac{X_1-x}{h})\}]^2 \\
&= \frac{1}{n}\int_{-\infty}^{+\infty}\frac{1}{h^2}K^2(\frac{z-x}{h})f(z)dz - \frac{1}{n}[\frac{1}{h}\int_{-\infty}^{+\infty}K(\frac{z-x}{h})f(z)dz]^2 \\
&= \frac{1}{n}\int_{-\infty}^{+\infty}\frac{1}{h}K^2(u)f(x+hu)du \\
&\quad - \frac{1}{n}[f(x)+\frac{1}{2}f^{(2)}(x)\kappa_{21}h^2+o(h^2)]^2 \\
&= \frac{f(x)}{nh}\kappa_{02}+o(\frac{1}{nh}).
\end{aligned}
$$

Bias-Varianve tradeoff

- Above all, we have

$$\text{Bias}(\hat{f}_n(x)) = \frac{1}{2} f^{(2)}(x) \kappa_{21} h^2 + o(h^2),$$
$$\text{Var}[\hat{f}_n(x)] = \frac{f(x)}{nh} \kappa_{02} + o(\frac{1}{nh}).$$

- So, the MSE of the kernel density estimate can be expressed as

$$\begin{aligned}\text{MSE}[\hat{f}_n(x)] &= \text{Var}[\hat{f}_n(x)] + [\text{Bias}(\hat{f}_n(x))]^2 \\ &= \frac{f(x)}{nh} \kappa_{02} + \frac{1}{4} [f^{(2)}(x)]^2 \kappa_{21}^2 h^4 + o(\frac{1}{nh}) + o(h^4).\end{aligned}$$

Bias-Varianve tradeoff

- An important aspect of the kernel density estimator is the selection of the bandwidth h. A too large bandwidth results in an oversmoothed estimate which obscures the fine structure of the data. A too small bandwidth makes an undersmoothed estimate, producing a wiggly curve and artificial modes.

Table of Contents

Nadaraya–Watson estimator

- Consider the model
$$Y = f(X) + \epsilon.$$
- Suppose that we have a sample $(X_1, Y_1), \cdots, (X_n, Y_n)$ of n i.i.d. pairs of random variables having the same density function as $f(x, y)$.
- If $f(x) > 0$, then we have
$$f(x) = E(Y|X = x) = \int yf(y|x)dy = \frac{\int yf(x, y)dy}{f(x)}.$$

Nadaraya–Watson estimator

- A kernel estimator of $f(x, y)$ is then given by the formula

$$\hat{f}_n(x, y) = \frac{1}{nh^2} \sum_{i=1}^{n} K(\frac{X_i - x}{h}) K(\frac{Y_i - y}{h}).$$

- If we replace $f(x, y)$ by the estimator $\hat{f}_n(x, y)$ and use the kernel estimator $\hat{f}_n(x)$ instead of $f(x)$, we obtain the N-W estimator

$$f_n^{NW}(x) = \frac{\int y \hat{f}_n(x, y) dy}{\hat{f}_n(x)} = \frac{\sum_{i=1}^{n} Y_i K(\frac{X_i - x}{h})}{\sum_{i=1}^{n} K(\frac{X_i - x}{h})}.$$

Nadaraya–Watson estimator

$$f_n^{NW}(x) = \frac{\int y\hat{f}_n(x,y)dy}{\hat{f}_n(x)} = \frac{\sum_{i=1}^{n} Y_i K(\frac{X_i - x}{h})}{\sum_{i=1}^{n} K(\frac{X_i - x}{h})}.$$

Proof.

$$\int y\hat{f}_n(x,y)dy = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{X_i - x}{h})\frac{1}{h}\int yK(\frac{Y_i - y}{h})dy.$$

$$\begin{aligned}\frac{1}{h}\int yK(\frac{Y_i - y}{h})dy &= \int \frac{y - Y_i}{h}K(\frac{Y_i - y}{h})dy + \frac{Y_i}{h}\int K(\frac{Y_i - y}{h})dy \\ &= -h\int uK(u)du + Y_i\int K(u)du = Y_i.\end{aligned}$$

k–Nearest-Neighbor

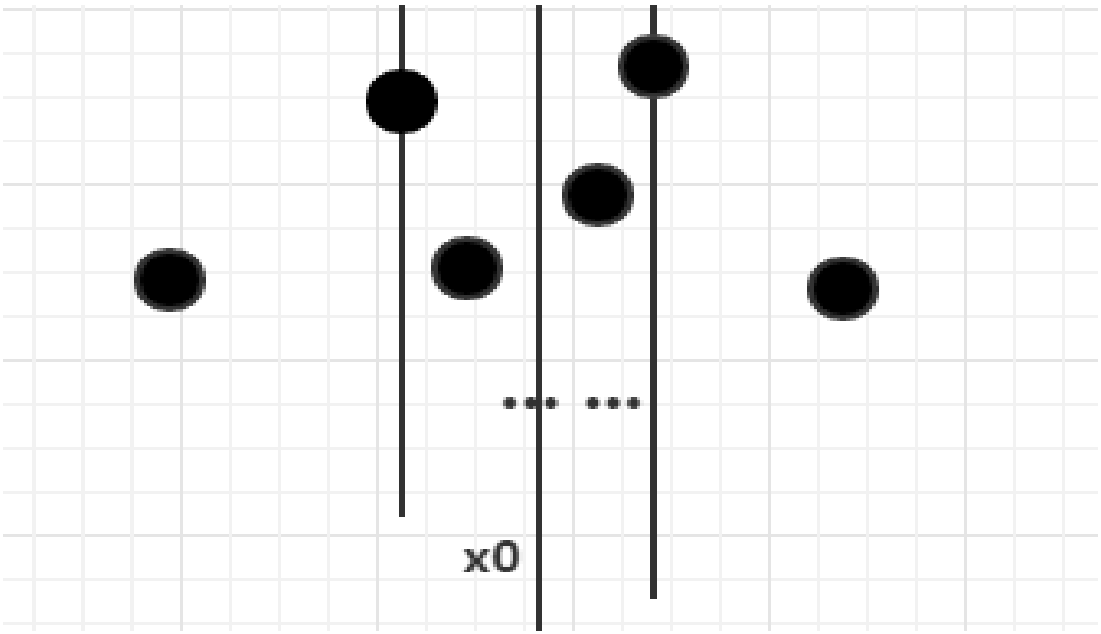
- Consider KNN, we use the k-nearest-neighbor average

$$\hat{f}(x) = \text{Ave}(y_i | x_i \in N_k(x))$$

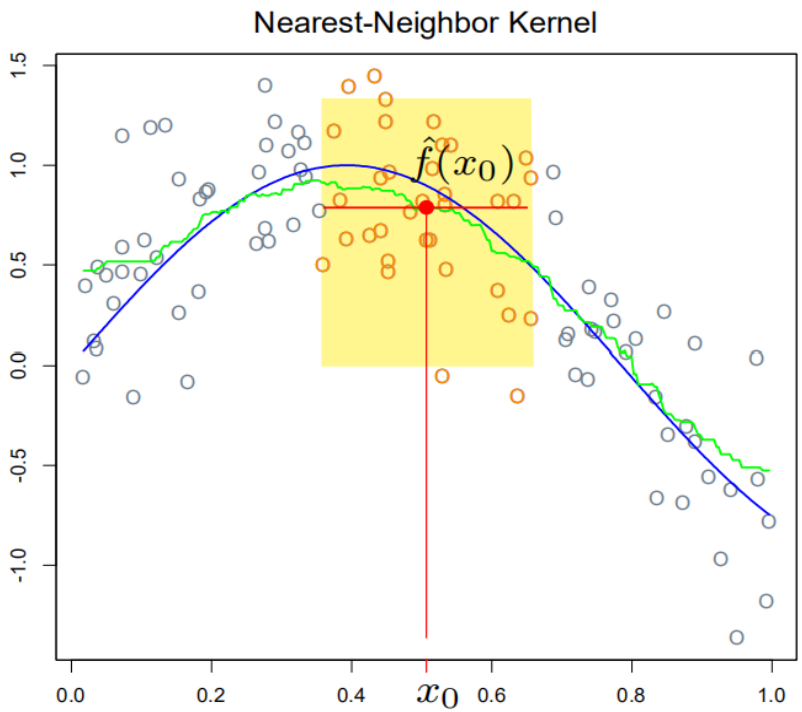
  as an estimate of the regression function $E(Y|X = x)$.
- Here $N_k(x)$ is the set of k points nearest to x in squared distanve, and Ave denotes the average (mean).

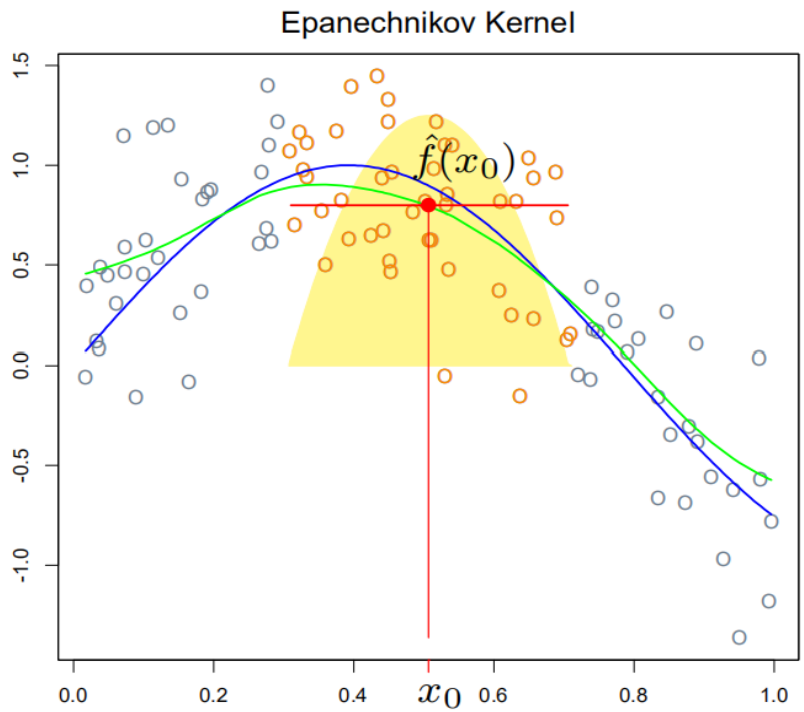
k–Nearest-Neighbor

- As we move x_0 from left to right, the k-nearest neighborhood remains constant, until a point x_i to the right of x_0 become closer than the furthest ponit $x_{i'}$ in the neighborhood to the left of x_0, at which time x_i replaces $x_{i'}$.
- The average changes in a discrete way, thus leading to a discontinuous $\hat{f}(x)$.



k–Nearest-Neighbor





k–Nearest-Neighbor

- Rather than give all the points in the neighborhood equal weight, we can assign weights that die off smoothly with distance from the target point x_0.
- For example, we can use Nadaraya-Watson kernel-weighted average

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{N} K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^{N} K_\lambda(x_0, x_i)},$$

with the Epanechnikov quadratic kernel

$$K_\lambda(x_0, x_i) = D\left(\frac{|x - x_0|}{\lambda}\right), \quad D(t) = \begin{cases} \frac{3}{4}(1 - t^2) & \text{if} |t| \leq 1; \\ 0 & \text{otherwise.} \end{cases}$$

k–Nearest-Neighbor

- More generally, we have

$$K_\lambda = D(\frac{|x - x_0|}{h_\lambda(x_0)}),$$

  where $h_\lambda(x_0)$ is a width function (indexed by λ) that determines the width of the neighborhood at x_0.
- For k-nearest neighborhoods, the neighborhood size k replaces λ, and we have $h_k(x_0) = |x_0 - x_{[k]}|$ where $x_{[k]}$ is the kth closest x_i to x_0.

Some details in practice

- The smoothing parameter λ, which determine width of the local neighborhood, has to be determined. Large λ implies lower variance (averages over more observations) but higher bias.
- Metric window widths (constant $h_\lambda(x)$) tend to keep the bias of the estimate constant, but the variance is inversely proportional to the local density.

  Nearest-neighbor window widths exhibit the opposite behavior; the variance stays constant and the absolute bias varies inversely with local density.

Some details in practice

- When there are ties in the x_i, one can reduce the data set by averaging the y_i at tied values of X, and assign an additional weight ω_i to these new observations.
- How to determine observation weights?
  With nearest neighborhoods, it is natural to insist on neighborhoods with a total weight content k ($\sum \omega_i$). In the event of overflow (the last observation needed in a neighborhood has a weight ω_j which causes the sum to exceed k), then fractional parts can be used.

Some details in practice

- Boundary issues arise. The metric neighborhoods tend to contain less points on the boundaries, while the nearest-neighborhoods get wider.
- The Epanechnikov kernel has compact support. Another popular compact kernel is based on the tri-cube function

$$D(t) = \begin{cases} (1 - |t|^3)^3 & \text{if} |t| \leq 1; \\ 0 & \text{otherwise.} \end{cases}$$

This is flatter on the top (like the nearest-neighborhood box) and is differentiable at the boundary of its support. The Gaussian density function $\phi(t)$ is a popular noncompact kernel, with standard deviation playing the role of the window size.

Some details in practice

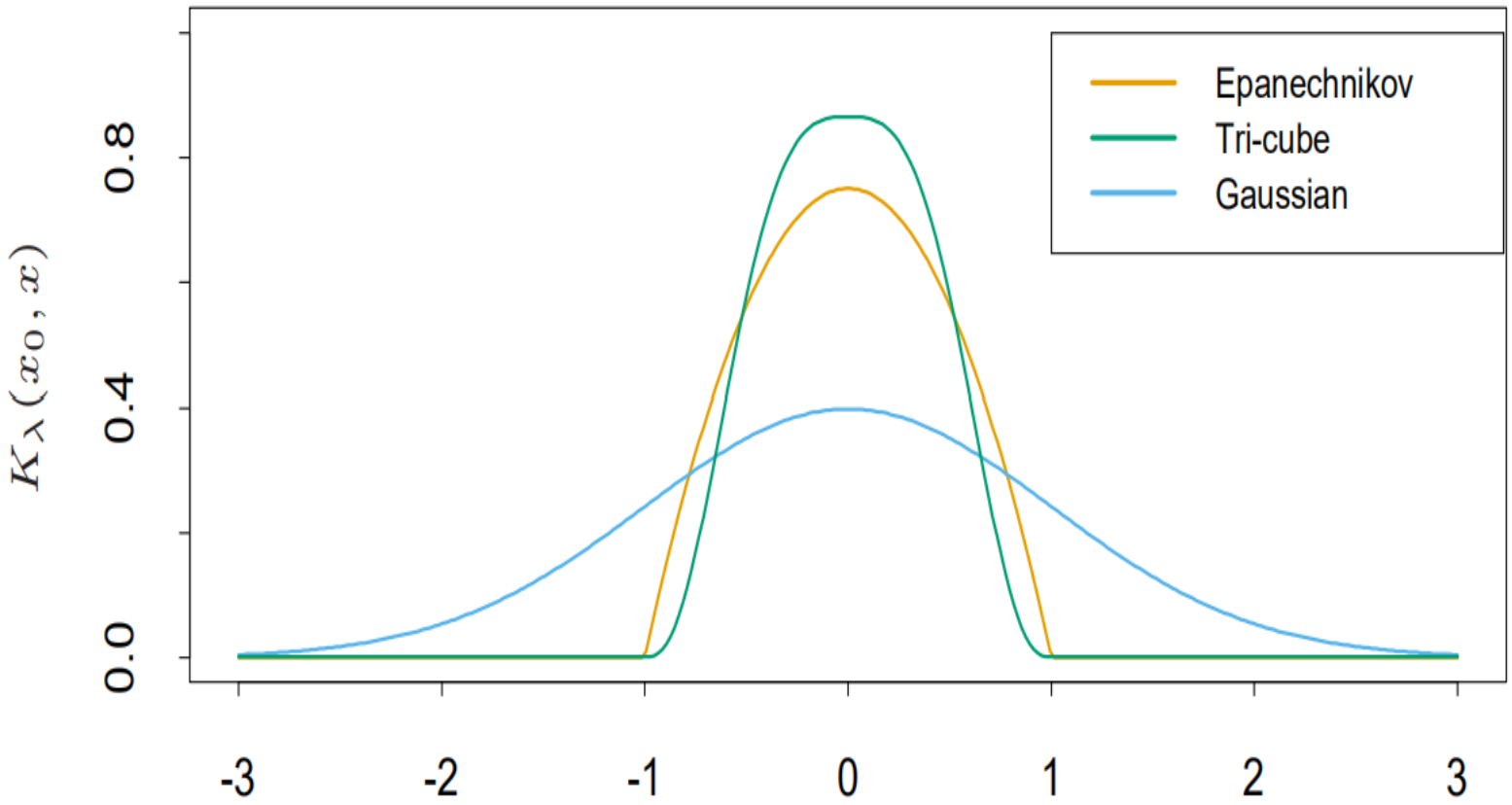
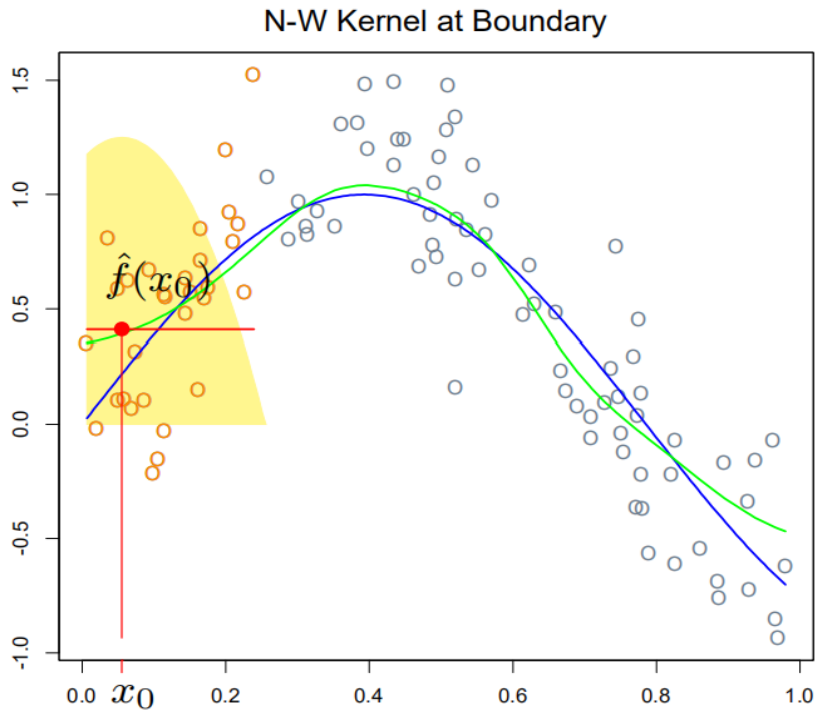
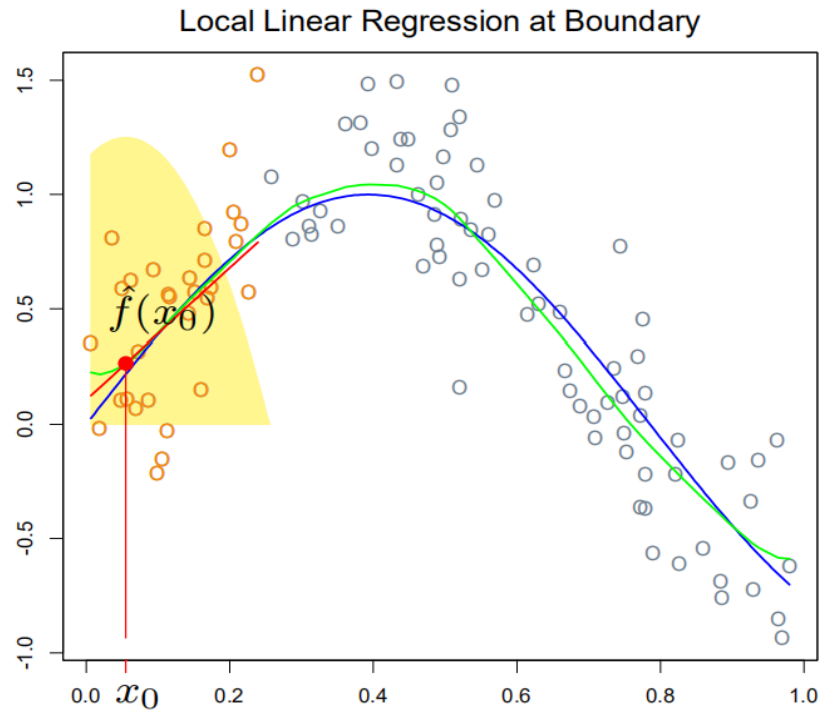


Table of Contents

Local Linear Regression

- Locally-weighted averages can be badly biased on the boundaries because of the asymmetry of the kernel in that region.
- By fitting straight lines rather than constants locally, we can remove this bias exactly to first order.





Local Linear Regression

- Locally weighted regression solves a weighted least squares problem at each target point x_0:

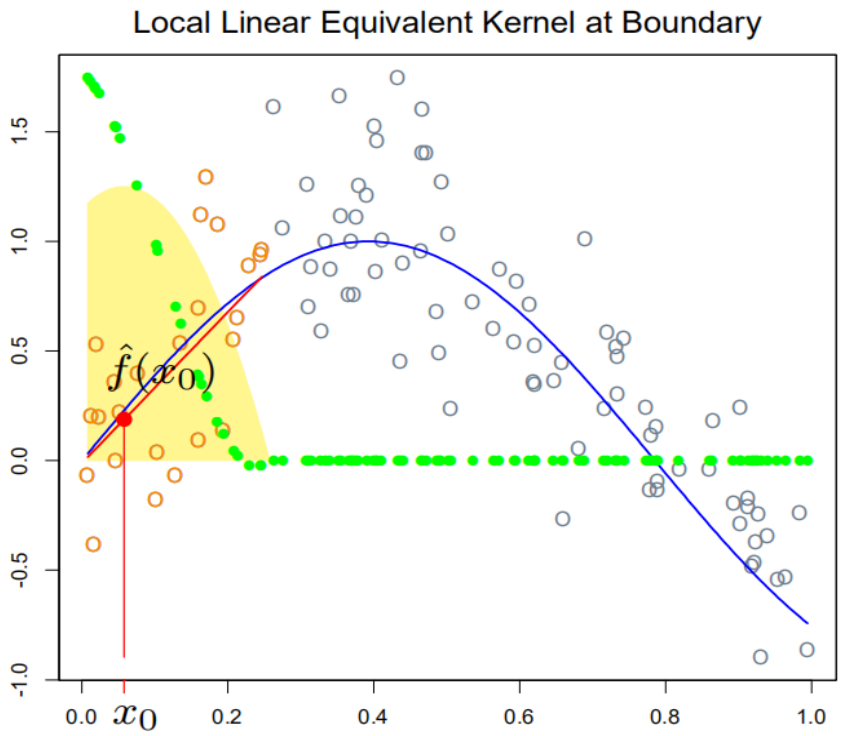
$$\min_{\alpha(x_0),\beta(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i)[y_i - \alpha(x_0) - \beta(x_0)x_i]^2.$$

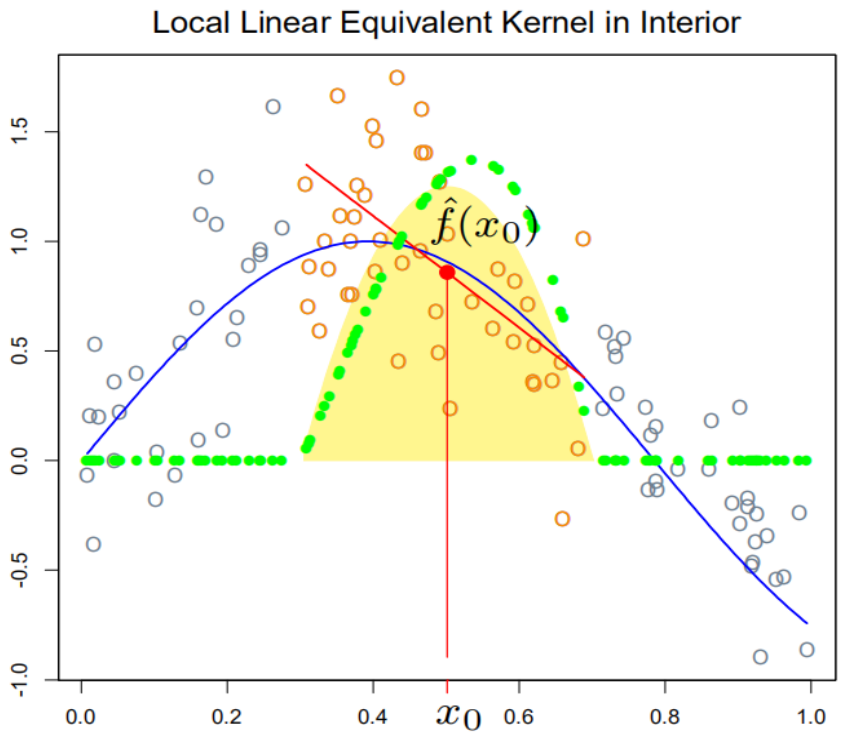
- Define $b(x)^T = (1, x)$. Let $\mathbf{B}$ be the $N \times 2$ regression matrix with ith row $b(x_i)^T$, and $\mathbf{W}(x_0)$ the $N \times N$ diagonal matrix with ith diagonal element $K_\lambda(x_0, x_i)$. Then

$$\begin{aligned}\hat{f}(x_0) &= b(x_0)^T(\mathbf{B}^T\mathbf{W}(x_0)\mathbf{B})^{-1}\mathbf{B}^T\mathbf{W}(x_0)\mathbf{y} \\ &= \sum_{i=1}^{N} l_i(x_0)y_i.\end{aligned}$$

Local Linear Regression

- These weights $l_i(x_0)$ combine the weighting kernel $K_\lambda(x_0, x_i)$ and the least squares operations, and are some times referred as the equivalent kernel.





Local Linear Regression

- Local linear regression automatically modifies the kernel to correct the bias exactly to first order.
- Consider the expansion for $E\hat{f}(x_0)$,

$$\begin{aligned} E\hat{f}(x_0) &= \sum_{i=1}^{N} l_i(x_0) f(x_i) \\ &= f(x_0) \sum_{i=1}^{N} l_i(x_0) + f'(x_0) \sum_{i=1}^{N} (x_i - x_0) l_i(x_0) \\ &\quad + \frac{f''(x_0)}{2} \sum_{i=1}^{N} (x_i - x_0)^2 l_i(x_0) + R. \end{aligned}$$

Local Linear Regression

- $\sum_{i=1}^{N} l_i(x_0) = 1, \quad \sum_{i=1}^{N} (x_i - x_0) l_i(x_0) = 0.$

Proof.

Let $l^T = (l_1(x_0), \cdots, l_n(x_0))$, from the soulution of the weighted least squares problem we have

$$l^T = b(x_0)^T (\mathbf{B}^T \mathbf{W}(x_0) \mathbf{B})^{-1} \mathbf{B}^T \mathbf{W}(x_0).$$

Thus, $(\sum_{i=1}^{n} l_i(x_0), \sum_{i=1}^{n} l_i(x_0) x_i) = l^T \mathbf{B} = b(x_0)^T = (1, x_0).$ □

- Hence the bias $E\hat{f}(x_0) - f(x_0)$ depends only on quadratic and higer-order terms in the expansion of f.

Table of Contents

Local Polynomial Regression

- We can fit local polynomial fits of any degree d,

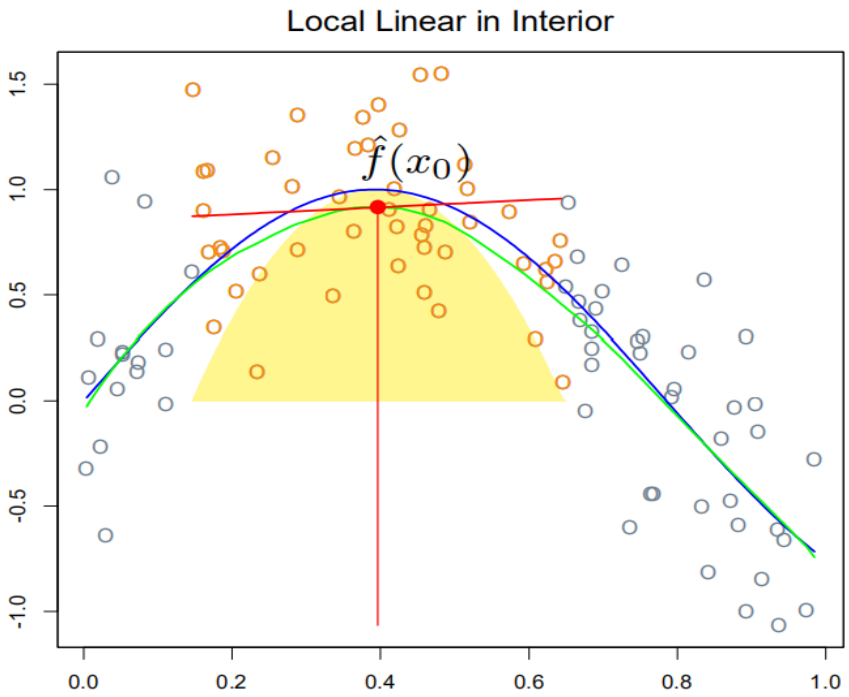
$$\min_{\alpha(x_0),\beta_j(x_0),j=1,\cdots,d} \sum_{i=1}^{N} K_\lambda(x_0,x_i)[y_i-\alpha(x_0)-\sum_{j=1}^{d}\beta_j(x_0)x_i^j]^2,$$

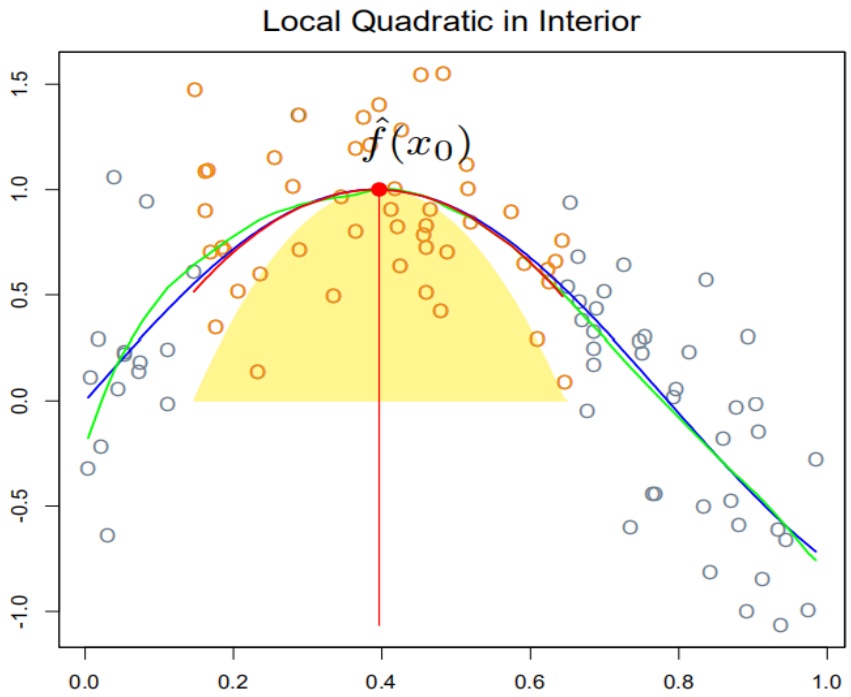
  with solution $\hat{f}(x_0)=\hat{\alpha}(x_0)+\sum_{j=1}^{d}\hat{\beta}_j(x_0)x_0^j$.

- The bias will only have components of degree $d+1$ and higer.

Local Polynomial Regression

- Local linear fits tend to be biased in regions of curvature of the true function, a phenomenon referred to as trimming the hills and filling the valleys.





Local Polynomial Regression

- A price to be paid for the bias reduction is increased variance.
- Assuming the model $y_i = f(x_i) + \epsilon_i$, with ϵ_i iid with mean zero and variance σ^2, $\text{Var}\hat{f}(x_0) = \sigma^2 ||l(x_0)||^2$, where $l(x_0)$ is the vector of equivalent kernel weights at x_0.
- It can be shown that $||l(x_0)||$ increases with d, thus there is a bias-variance tradeoff in selecting the polynomial degree.

Summary

- Local linear fits can help bias dramatically at the boundaries at a modest cost in variance. Local quadratic fits do little at the boundaries for bias, but increase the variance a lot.
- Local quadratic fits tend to be most helpful in reducing bias due to curvature in the interior of the domain.
- Asymptotic analysis suggest that local polynomials of odd degree dominate those of even degree. This is largely due to the fact that asymptotically the MSE is dominated by boundary effects.

Window Width λ

- In each kernel K_λ, λ is a parameter that controls it width.
- For the Epanechnikov or tri-cube kernel with metric width, λ is the radius of the support region.
- For the Gaussian kernel, λ is the standard deviation.
- For the KNN, λ is the number k of nearest neighbors.

Bias-variance Tradeoff for Local Average

- If the window is narrow, $\hat{f}(x_0)$ ia an average of a small number of y_i close to x_0, and its variance is close to an individual y_i. The bias will be small, because each of the $E(y_i) = f(x_i)$ should be close to $f(x_0)$.
- If the window is wide, the variance will be small. The bias will be higher, because because we use observations x_i further from x_0.

Bias-variance Tradeoff for Local Linear

- As width λ goes to zero (assign weights only to x_0 itself), the estimates approach a piecewise-linear function interpolating the training data.
- As width λ goes infinitely large(assign equal weights to each x_i), the fit approaches the local linear least-squares fit to the data.
- Leave-one-out cross-validation is particularly simple, as is generalized cross-validation, C_p, and k-fold cross-validation.

Table of Contents

Generalization

- For example, we get $b(X) = (1, X_1, X_2, X_1^2, X_2^2, X_1X_2)$ with $d = 2$ and $p = 2$.
- At each point $x_0 \in \mathbb{R}^p$ solve

$$\min_{\beta(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i)(y_i - b(x_i)^T\beta(x_0))$$

to produce the fit $\hat{f}(x_0) = b(x_0)^T\hat{\beta}(x_0)$.

- The kernel will be a radial function, such as

$$K_\lambda(x_0, x) = D(\frac{||x - x_0||}{\lambda}).$$

So, it make most sense to standardize each predictor.

The Curse of Dimensionality

- The fraction of points close to the boundary increases to one as the dimension grow.
- Define the interior of a hypercube as the hypersphere inscribed in it, then we can calculate the proportion of the edge:

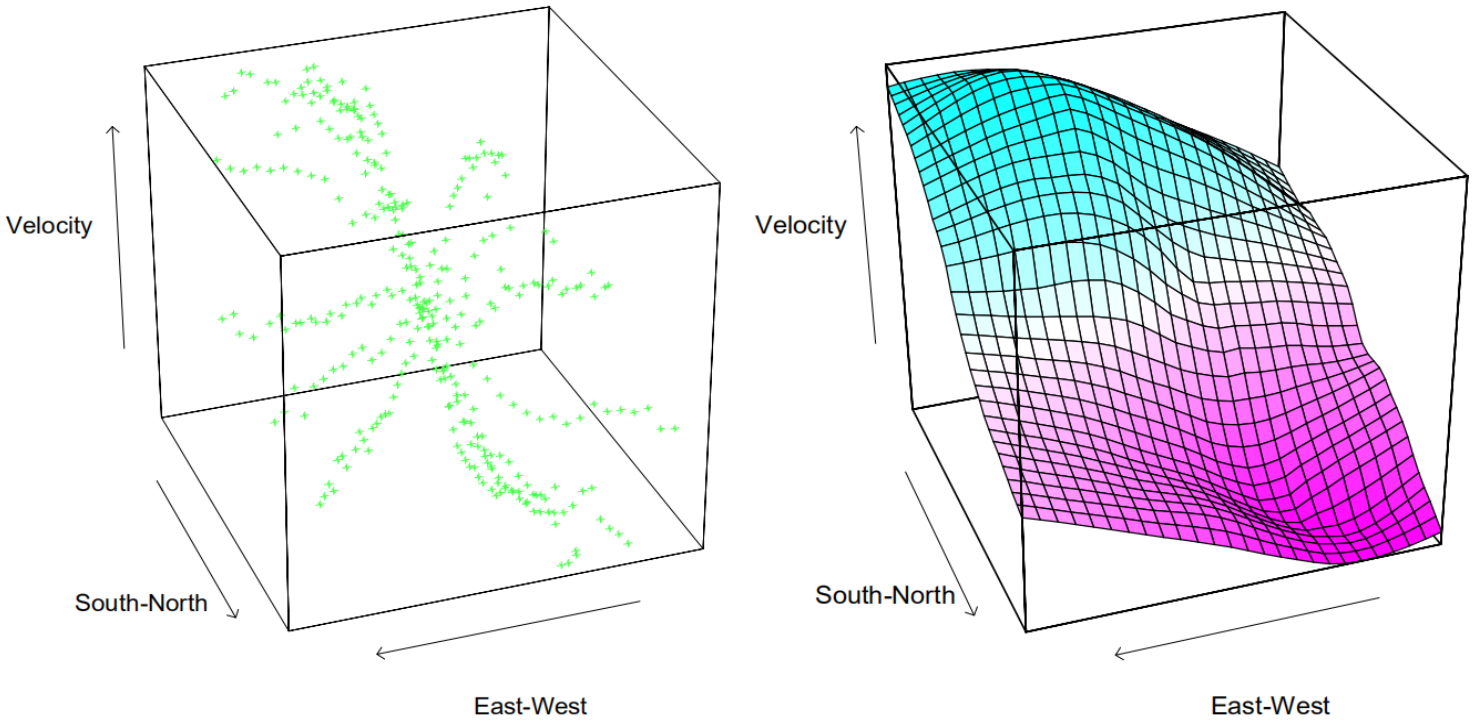
$$d = 1, P_{edge} = 1 - \frac{\pi}{4}$$

$$d = 2, P_{edge} = 1 - \frac{\pi}{6}$$

$$P_{edge} = 1 - \frac{\pi^{\frac{d}{2}}}{2^d \Gamma(1 + \frac{d}{2})} \to 1 \quad \text{as} \quad d \to \infty.$$

The Curse of Dimensionality

- It is impossible to simultaneously maintain localness (low bias) and a sizable sample in the neighborhood (low variance) as the dimension increases, without the sample size increasing exponentially in p.
- Visualization also beomes difficult which is often one of the primary goal of smoothing.



The Curse of Dimensionality

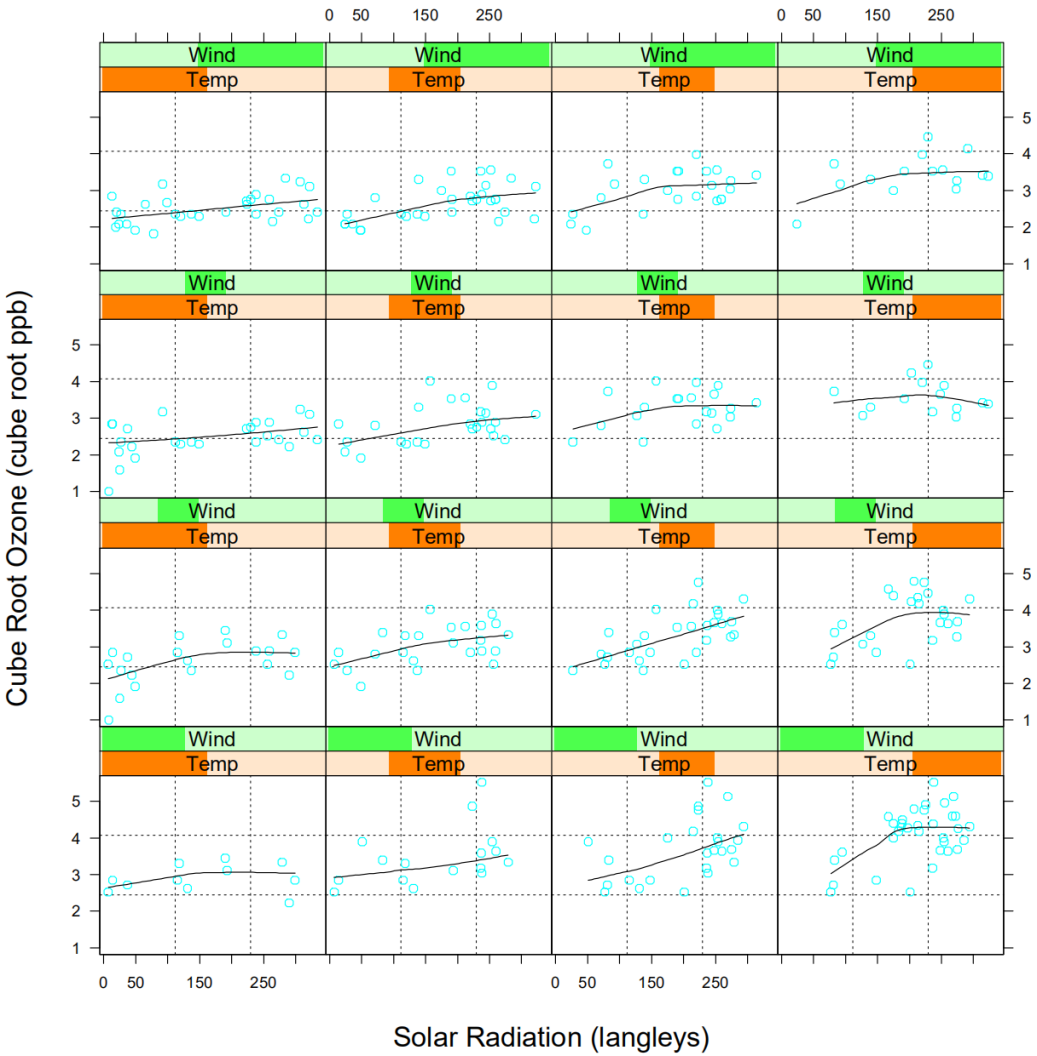


Table of Contents

Local Likelihood

- We can model $\theta(X)$ more flexibly by using the likelihood local to x_0 for inference of $\theta(X_0) = x_0^T\beta(x_0)$:

$$l(\beta(x_0)) = \sum_{i=1}^{N} K_\lambda(x_0, x_i) l(y_i, x_i^T\beta(x_0)),$$

  which allows a relaxation from a globally linear model to one that is locally linear.

Local Likelihood

- Consider the local version of the multiclass linear logistic regression model. The data consist of features x_i and an associated categorical response $g_i \in \{1, 2, ..., J\}$, and the linear model has the form

$$\Pr(G = j|X = x) = \frac{e^{\beta_{j0}+\beta_j^T x}}{1 + \sum_{k=1}^{J-1} e^{\beta_{k0}+\beta_k^T x}}.$$

- The local log-likelihood for this J class model can be written

$$\sum_{i=1}^{n} K_\lambda(x_0, x_i)\{\beta_{g_i 0}(x_0) + \beta_{g_i 0}^T(x_i - x_0) - log[1 + \sum_{k=1}^{J-1} e^{\beta_{k0}(x_0)+\beta_k(x_0)^T(x_i - x_0)}]\}.$$

Local Likelihood

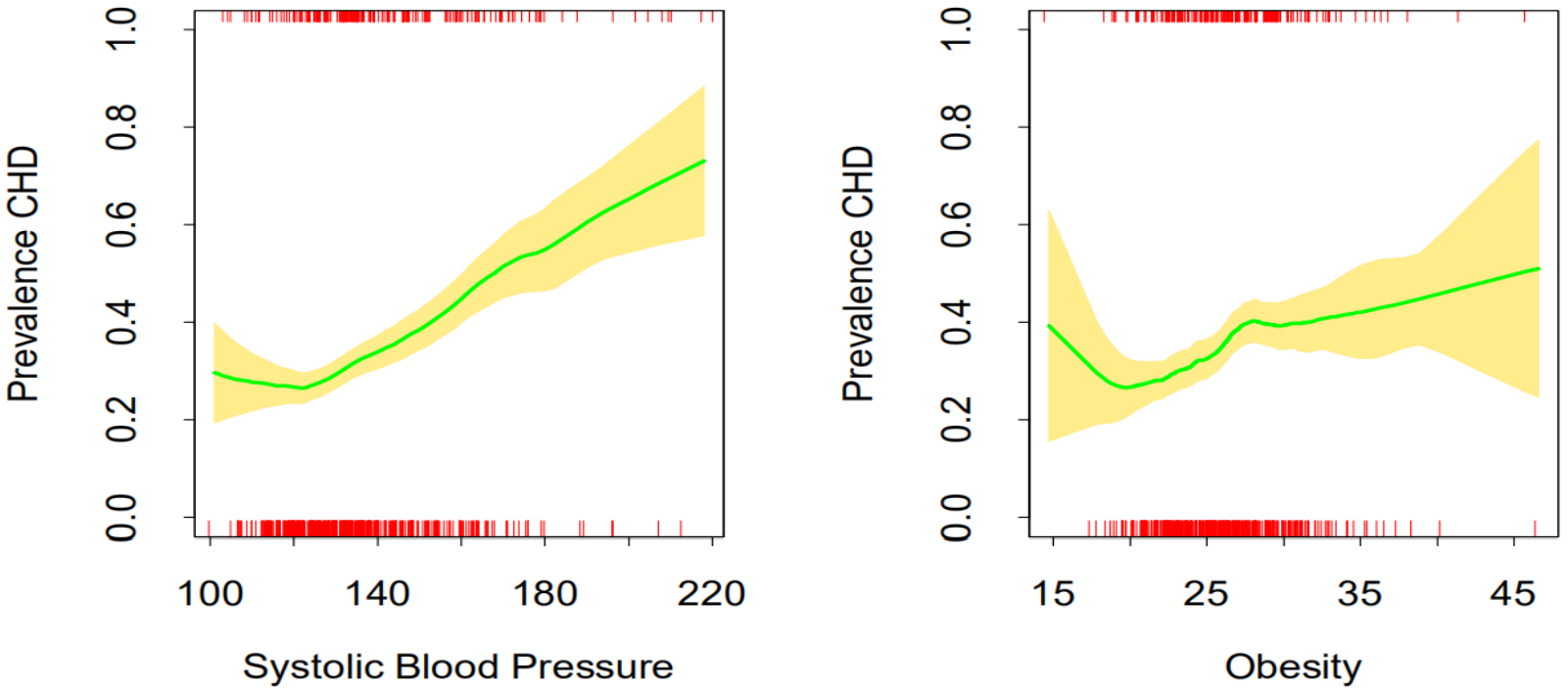


FIGURE 6.12. *Each plot shows the binary response CHD (coronary heart disease) as a function of a risk factor for the South African heart disease data. For each plot we have computed the fitted prevalence of CHD using a local linear logistic regression model. The unexpected increase in the prevalence of CHD at the lower ends of the ranges is because these are retrospective data, and some of the subjects had already undergone treatment to reduce their blood pressure and weight. The shaded region in the plot indicates an estimated pointwise standard error band.*

Table of Contents

Computional Considerations

- The computational cost to fit at a single observation x_0 is $O(N)$ flops, except in oversimplied cases. By comparison, an expansion in M basis functions cost $O(M)$ for one evaluation, and typically $M \sim O(logN)$. Basis function methods have an initial cost of at least $O(NM^2 + M^3)$.
- The smoothing parameter(s) λ are typically determined off-line, for example using cross-validation, at a cost of $O(N^2)$ flops.

Table of Contents

Structured Kernels

- When the dimension to sample-size ratio is unfavorable, we can make some structural assumptions about the model.
- The default spherical kernel

$$K_\lambda(x_0, x) = D(\frac{||x - x_0||^2}{\lambda}) = D(\frac{(x - x_0)^T I(x - x_0)}{\lambda})$$

  gives equal weight to each coordinate.
- A more general approach is to use a positive semidefinite matrix $\mathbf{A}$ to weight the different coordinates:

$$K_{\lambda,A}(x_0, x) = D(\frac{(x - x_0)^T A(x - x_0)}{\lambda}).$$

Structured Kernels

- We then can impose appropriate restrictions on $\mathbf{A}$ to downgrade or omit entire coordinates or directions.
- If $\mathbf{A}$ is diagonal, we can increase the influence of X_j by increasing A_{jj}.
- $\mathbf{A}$ can also include the information of the correlation between predictors.

Structured Regression Functions

- Consider analysis-of-variance (ANOVA) decompositions of the regression function $f(X_1, X_2, \cdots, X_p) = E(Y|X)$:

$$f(X_1, X_2, \cdots, X_p) = \alpha + \sum_j g_j(X_j) + \sum_{k<l} g_{kl}(X_k, X_l) + \cdots$$

- Additive models assume only main effect terms, and second-order models will have terms with interactions of order at most two.
- Iterative backfitting algorithms can be used to fit such low-order interaction models.
- In the additive model, if all but the kth term is assumed known, then we can estimate g_k by local regression of $Y - \sum_{j \neq k} g_j(X_j)$ on X_k. This is down for each function in turn, repeatedly, until convergence.

Structured Regression Functions

- Consider varying coefficient model:

$$f(X) = \alpha(Z) + \beta_1(Z)X_1 + \cdots + \beta_q(Z)X_q,$$

  where we divide p predictors in X in to $(X_1, X_2, \cdots, X_q)$ with $q < p$, and the remainder of the variables we collect in Z.

- For given Z, we can fit a such model by locally weighted least squares:

$$\min_{\alpha(z_0),\beta(z_0)} \sum_{i=1}^{N} K_\lambda(z_0, z_i)(y_i - \alpha(x_0) - x_{1i}\beta_1(z_0) - \cdots - x_{qi}\beta_q(z_0))^2.$$

Structured Regression Functions