

---

# Kernel Methods

## *Chapter 6: Hastie et al. (2001)*

Madhusudana Shashanka

Department of Cognitive and Neural Systems  
Boston University

# Introduction

---

- Regression techniques - estimate the regression function  $f(X)$  over the domain  $\mathbb{R}^p$ .
- Fit a different model separately at each query point  $x_0$ .
- $\hat{f}(X)$  is smooth in  $\mathbb{R}^p$ .
- Kernel  $K_\lambda(x_0, x_i)$  assigns a weight to  $x_i$  based on its distance from  $x_0$ .
- Memory based - little/no training, but model is the entire training data set.

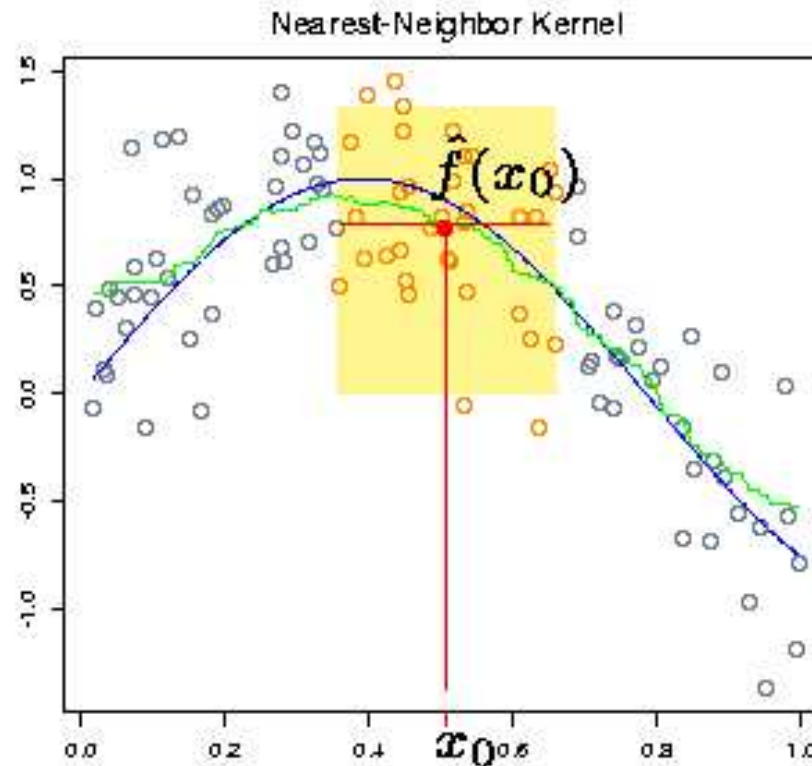
# Outline

---

- One-dimensional kernel smoothers
- Width of the kernel
- Local regression in  $\mathbb{R}^p$
- Structured local regression models in  $\mathbb{R}^p$
- Local likelihood and other models
- Kernel density estimation and classification
- Radial basis functions and kernels
- Mixture models for density estimation and classification
- Computational considerations

# $k$ NN – Recap

- $\hat{f}(x) = \text{Ave}(y_i | x_i \in N_k(x))$ , an estimate of the regression function  $E(Y|X = x)$ .
- Changes are discrete, discontinuous  $\hat{f}(x)$ .



# Kernel smoothers

---

- KNN - all neighborhood points get equal weight.
- Kernel - weights die off smoothly with distance.
- Nadaraya-Watson kernel weighted average

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

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

- Epanechnikov: 
$$D(t) = \begin{cases} \frac{3}{4}(1 - t^2) & \text{if } |t| \leq 1; \\ 0 & \text{otherwise.} \end{cases}$$

- $h_\lambda(x_0)$  is a width function.

- Smoothing parameter  $\lambda$  determines the width of the local neighborhood.

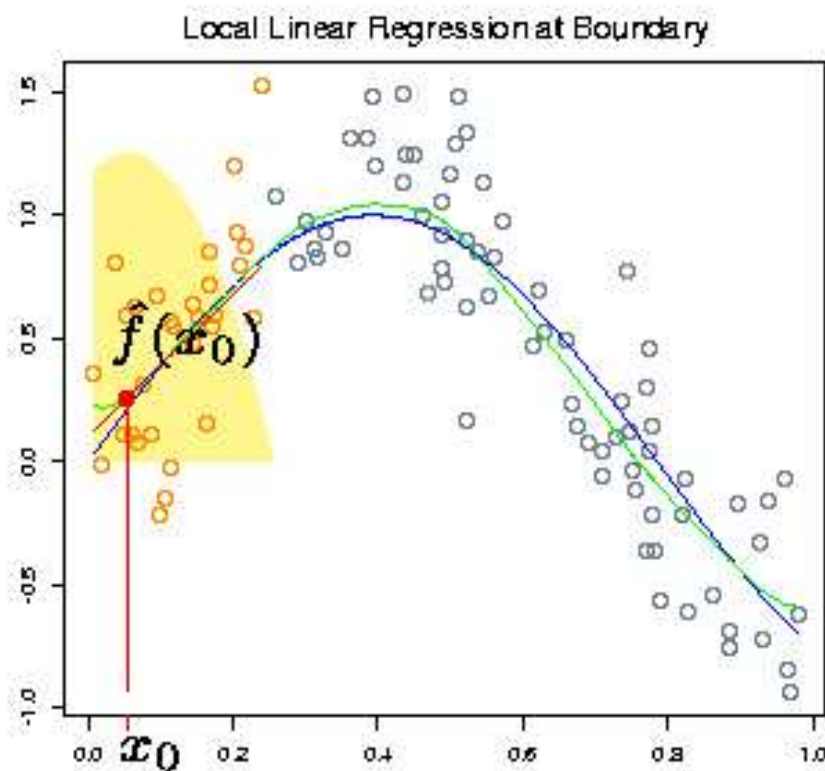
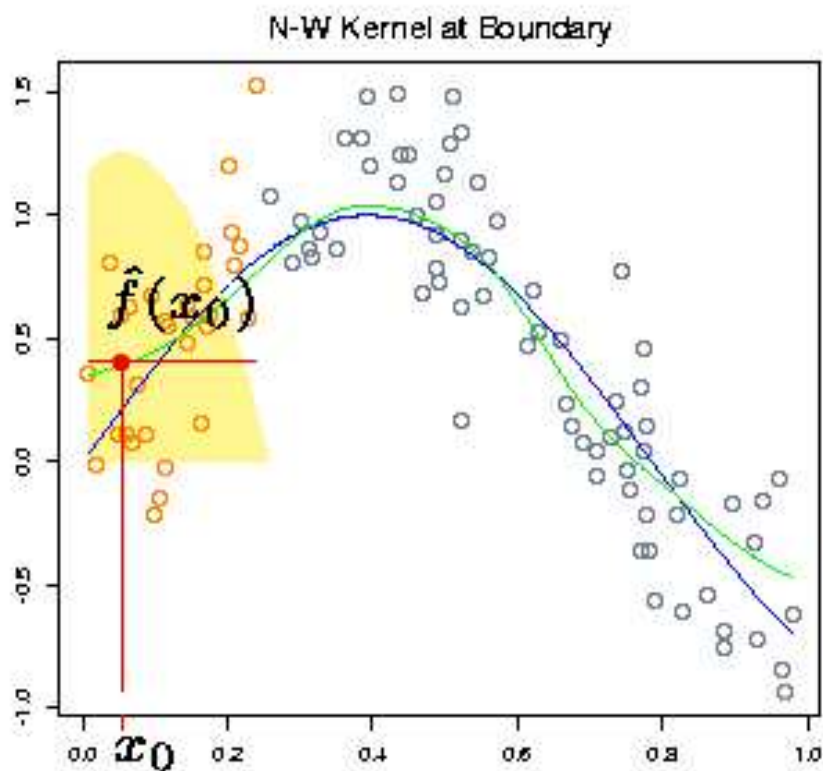
# Kernel smoothing – issues

---

- Large  $\lambda \Rightarrow$  lower variance, higher bias.
- Metric window widths (constant  $h_\lambda(x)$ )  $\Rightarrow$  bias constant, variance inversely proportional to local density.
- Nearest-neighbor window width  $\Rightarrow$  opposite behavior
- Observation weights: multiply them by the kernel weights.
- Boundary: metric neighborhoods have less points while nearest neighborhoods get wider.
  - Epanechnikov (compact)
  - Tri-cube (compact):  $D(t) = \begin{cases} (1 - t^3)^3 & \text{if } |t| \leq 1; \\ 0 & \text{otherwise.} \end{cases}$
  - Gaussian (noncompact): Standard deviation = window size.

# Local Linear Regression

- Problem: Bias at boundaries
- Cause: asymmetry of kernel at the boundary.
- Solution: Fit straight lines locally, first-order correction.



# Local Linear Regression

---

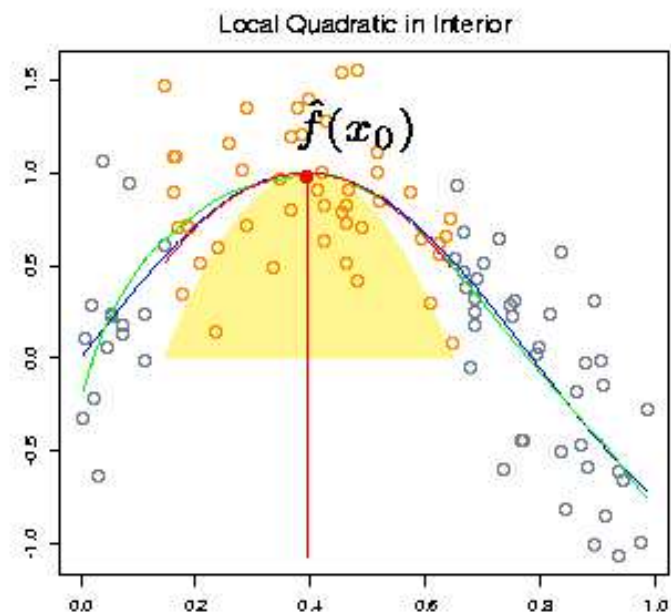
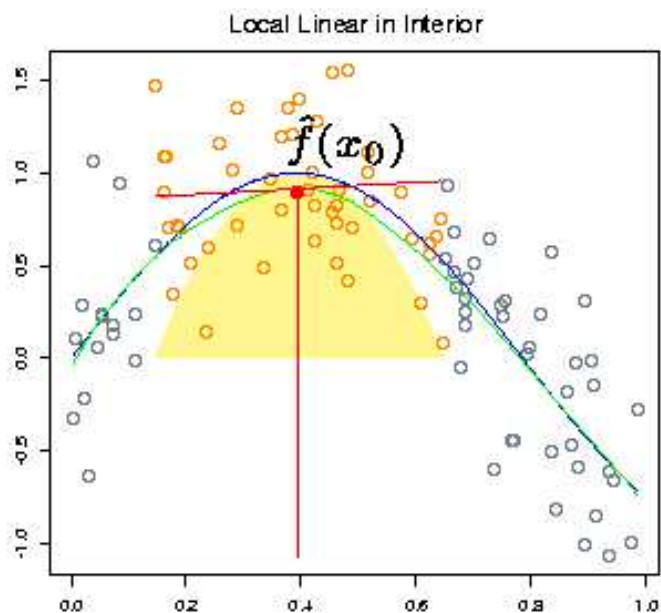
- If  $b(x)^T = (1, x)$ ,  $\mathbf{B} = N \times 2$  regression matrix with  $i$ th row  $b(x_i)^T$ ,  $\mathbf{W}(x_0) = N \times N$  diag matrix with  $i$ th element  $K_\lambda(x_0, x_i)$

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

- Weights  $l_i(x_0)$  combine weighting kernel and least squares, hence equivalent kernel.
- Exact and automatic modification  $\Rightarrow$  automatic kernel carpentry.

# Local Polynomial Regression

- Problem with linear: *trimming the hills and filling the valleys.*
- Polynomial of higher degree, but bias variance tradeoff.
- Quadratic fits – not useful at boundaries, good for reducing bias due to the curvature at in the domain interior.
- Odd degree dominates even degree asymptotically.



# Kernel Width

---

- $\lambda$  controls the width of  $K_\lambda$ .
  - Epanechnikov/tri-cube:  $\lambda =$  radius of support region.
  - Gaussian:  $\lambda =$  standard deviation.
  - $k$ NN:  $\lambda =$  number of nearest neighbors expressed as a fraction/span of the total training sample.
- Bias-variance tradeoff as width changes
  - Narrow window  $\Rightarrow$  high variance, low bias.
  - Wide window  $\Rightarrow$  small variance, higher bias.
- Local regression can be naturally generalized to two or more dimensions.

# Structured Local Regression

---

- Local regression of no help if dimension to sample-size ratio unfavorable.
- Modify kernel using a positive semidefinite matrix  $\mathbf{A}$  to weigh the different coordinates:

$$k_{\lambda, \mathbf{A}}(x_0, x) = d\left(\frac{(x - x_0)^T \mathbf{A} (x - x_0)}{\lambda}\right)$$

- Consider ANOVA decompositions of the form

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

and eliminate higher-order terms.

- Special case – varying coefficient models.

# Local Likelihood

---

- Observation  $y_i$ , parameter  $\theta_i = \theta(x_i) = x_i^T \beta$ , inference for  $\beta$  based on  $l(\beta) = \sum_{i=1}^N l(y_i, x_i^T \beta)$ .
- Use likelihood local to  $x_0$  for inference of  $\theta(x_0) = x_0^T \beta(x_0)$ :  
 $l(\beta(x_0)) = \sum_{i=1}^N K_\lambda(x_0, x_i) l(y_i, x_i^T \beta(x_0))$
- If different variables are associated with  $\theta$ , we have  
 $l(\theta(z_0)) = \sum_{i=1}^N K_\lambda(z_0, z_i) l(y_i, \eta(x_i, \theta(z_0)))$ .
- Autoregressive time series models of order  $k$  of the form  
 $y_t = \beta_0 + \beta_1 y_{t-1} + \dots + \beta_k y_{t-k} + \epsilon_t$ 
  - Linear model  $y_t = z_t^T \beta + \epsilon_t$ , where  
 $z_t = (y_{t-1}, y_{t-2}, \dots, y_{t-k})$ .
  - Local least squares with  $K(z_0, z_t)$  – variation according to short-term history.

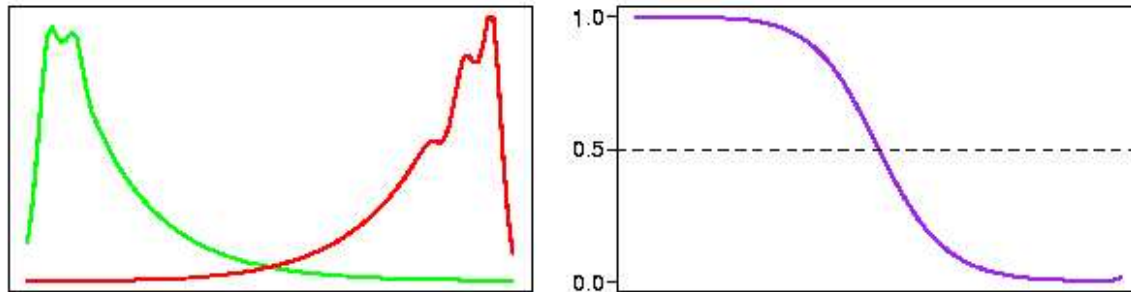
# Kernel Density Estimation

---

- Random sample  $x_1, \dots, x_N$  drawn from density  $f_X(x_0)$ , estimate  $f_X$  at  $x_0$ .
- Local estimate:  $\hat{f}_X(x_0) = \frac{\#x_i \in Nbd(x_0)}{N\lambda}$
- Parzen estimate:  $\hat{f}_X(x_0) = \frac{1}{N\lambda} \sum_{i=1}^N K_\lambda(x_0, x_i)$
- For a gaussian kernel  $K_\lambda(x_0, x) = \phi(|x - x_0|/\lambda)$ , the estimate is  $\frac{1}{N} \sum_{i=1}^N \phi_\lambda(x - x_i) = (\hat{F} * \phi_\lambda)(x)$ .
- In  $\mathbb{R}^p$ , the estimate generalizes to 
$$\hat{f}_X(x_0) = \frac{1}{N(2\lambda^2\pi)^{\frac{p}{2}}} \sum_{i=1}^N e^{-\frac{1}{2}(\|x_i - x_0\|/\lambda)^2}.$$

# Kernel Density Classification

- Learning separate class densities unnecessary.



- Naive Bayes Model: Given a class  $j$ , features  $X_k$  are independent i.e.  $f_j(X) = \prod_{k=1}^p f_{jk}(X_k)$ .
- $f_{jk}$  can be estimated separately using one-dimensional densities.
- If a component  $X_j$  of  $X$  is discrete, then an appropriate histogram estimate can be used.

# Radial Basis Functions

---

- Treat the kernel functions as basis functions.
- Model:  $f(x) = \sum_{j=1}^M K_{\lambda_j}(\xi, x)\beta_j = \sum_{j=1}^M D\left(\frac{\|x-\xi_j\|}{\lambda_j}\right)\beta_j$ , where  $\xi_j$  is the location/prototype parameter,  $\lambda_j$  is a scale parameter.
- Most used: least squares and Gaussian kernel.
  - Optimize the sum-of-squares w.r.t. all parameters:  
$$\sum_{i=1}^N \left( y_i - \beta_0 - \sum_{j=1}^M \beta_j \exp\left\{ -\frac{(x_i - \xi_j)^T (x_i - \xi_j)}{\lambda_j^2} \right\} \right)^2$$
  - Estimate  $\{\lambda_j, \xi_j\}$  separately from  $\beta_j$ . Given the former, estimating the latter is a simple least squares problem.
- Constant value for  $\lambda_j = \lambda \Rightarrow$  *holes*.
- Solution: Renormalized RBFs  $h_j(x) = \frac{D(\|x-\xi_j\|/\lambda)}{\sum_{k=1}^M D(\|x-\xi_k\|/\lambda)}$ .

# Mixture models

---

- Gaussian mixture model:  $f(x) = \sum_{m=1}^M \alpha_m \phi(x; \mu_m, \Sigma_m)$ , where  $\sum_m \alpha_m = 1$ .
- Usually, EM algorithm used for fitting.
- Special cases:
  - $\Sigma_m = \sigma_m \mathbf{I}$  –  $f(x)$  has the form of radial basis expansion.
  - Also, if  $\sigma_m = \sigma > 0$  is fixed and  $M \uparrow N$ , MLE approaches kernel density estimate, where  $\hat{\alpha}_m = 1/N$  and  $\hat{\mu}_m = x_m$ .
- Probability that observation  $i$  belongs to component  $m$  is

$$\hat{r}_{im} = \frac{\hat{\alpha}_m \phi(x_i; \hat{\mu}_m, \hat{\Sigma}_m)}{\sum_{k=1}^M \hat{\alpha}_k \phi(x_i; \hat{\mu}_k, \hat{\Sigma}_k)}.$$

# Computational Considerations

---

- Infeasible for many real-time applications.
- Single observation:  $O(N)$  flops.
- Expansion in  $M$  basis functions:
  - $O(M)$  for one evaluation
  - $M \approx O(\log N)$
  - Initial cost at least  $O(NM^2 + M^3)$ .
- $\lambda$  determined offline,  $O(N^2)$  flops.

# References

---

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning: data mining, inference, and prediction*. Springer-Verlag.