BCMI

Brain-like Computing & Machine Intelligence

Kernel Methods

Dept. Computer Science & Engineering,
Shanghai Jiao Tong University

Key Points in the Last Talk

- Good representation of function spaces
  - Easy to implement (efficient in space and time)
  - Good for generalization
  - Easy to select good models
- Good parameter for model selection
  - Effective degrees of freedom
  - CV for Model selection
- Reproducing Kernel Hilbert Space
  - Polynomial Kernel
- Spline & Wavelet

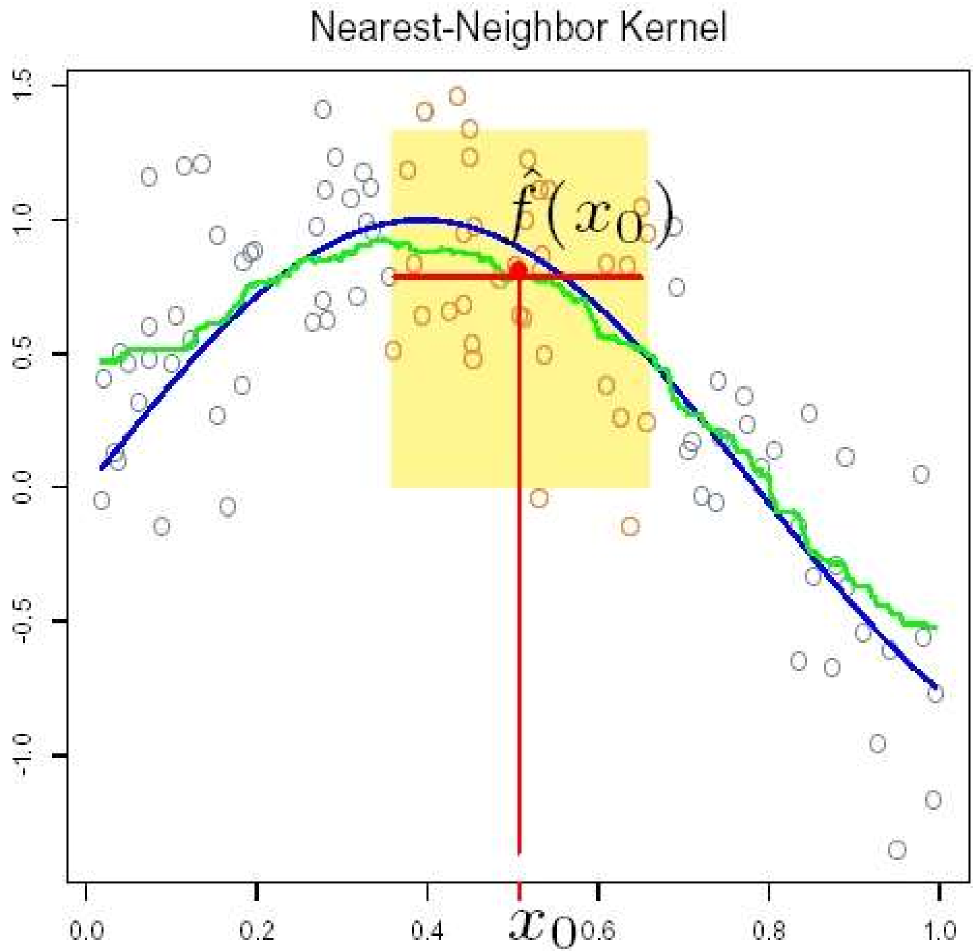
Outline

- One-Dimensional Kernel Smoothers
- Local Regression
- Local Likelihood
- Kernel Density estimation
- Naive Bayes
- Radial Basis Functions
- Mixture Models and EM

Objectives of OBE

- To understand why increasing the order of polynomials causes the increase of model variance
- How to kernel method to implement local regression
- Probability density estimation by kernel methods
- EM algorithm for estimating GMM

One-Dimensional Kernel Smoothers

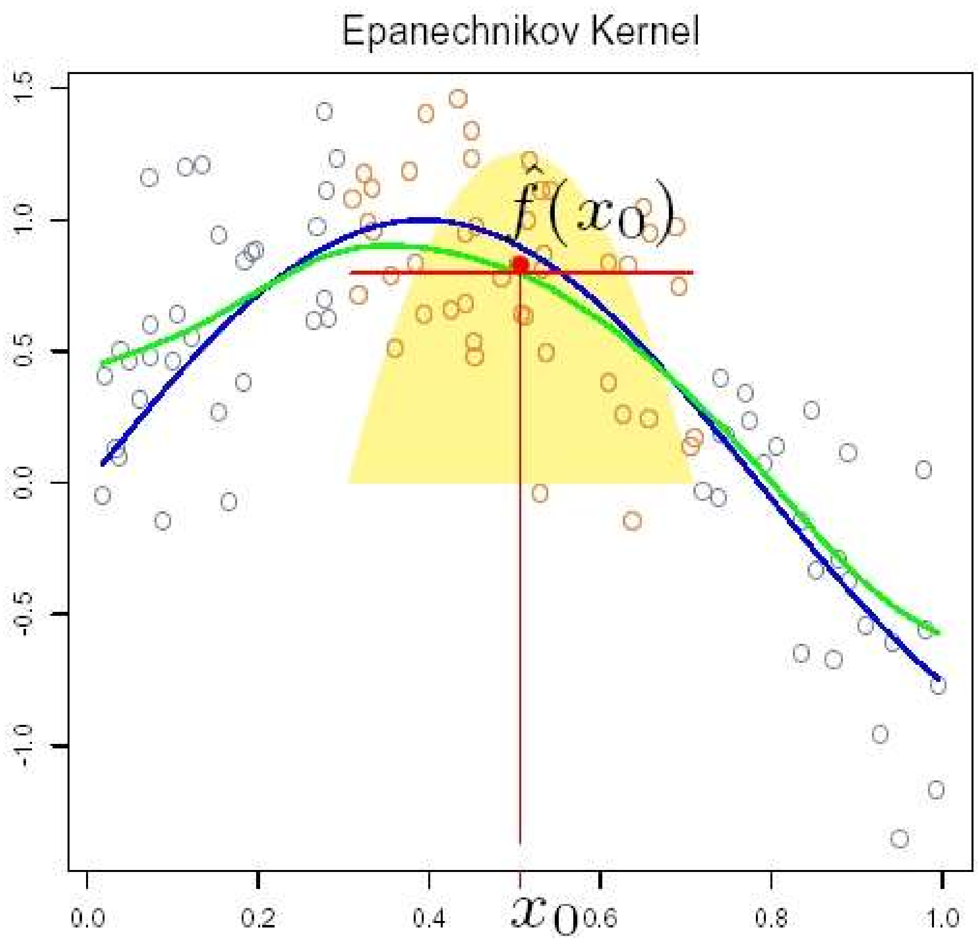


- k-NN:

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

- 30-NN curve is bumpy, since $\hat{f}(x)$ is discontinuous in x.
- The average changes in a discrete way, leading to a discontinuous $\hat{f}(x)$.

One-Dimensional Kernel Smoothers



- Nadaraya-Watson Kernel weighted average:

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

- Epanechnikov quadratic kernel:

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

One-Dimensional Kernel Smoothers

- More general kernel:

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

  – $h_\lambda(x_0)$: width function that determines the width of the neighborhood at x_0.

  – For quadratic kernel $h_\lambda(x_0) = \lambda,$ Bias $\approx$ constant

  – For k-NN kernel $\lambda \underset{replaced}{\leftrightarrow} k \quad h_k(x_0) = |x_0 - x_{[k]}|,$

  Variance $\approx$ constant

One-Dimensional Kernel Smoothers

- Three popular kernel for local smoothing:

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

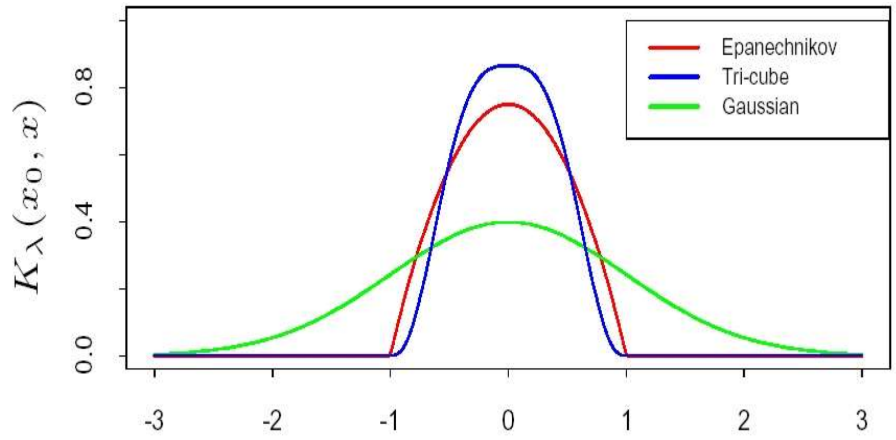
- Epanechnikov kernel and tri-cube kernel are compact but tri-cube has two continuous derivatives

- Gaussian kernel is infinite support

$$\text{Gaussian} : D(t) = \phi(t) = e^{-\frac{1}{2}t^2}$$

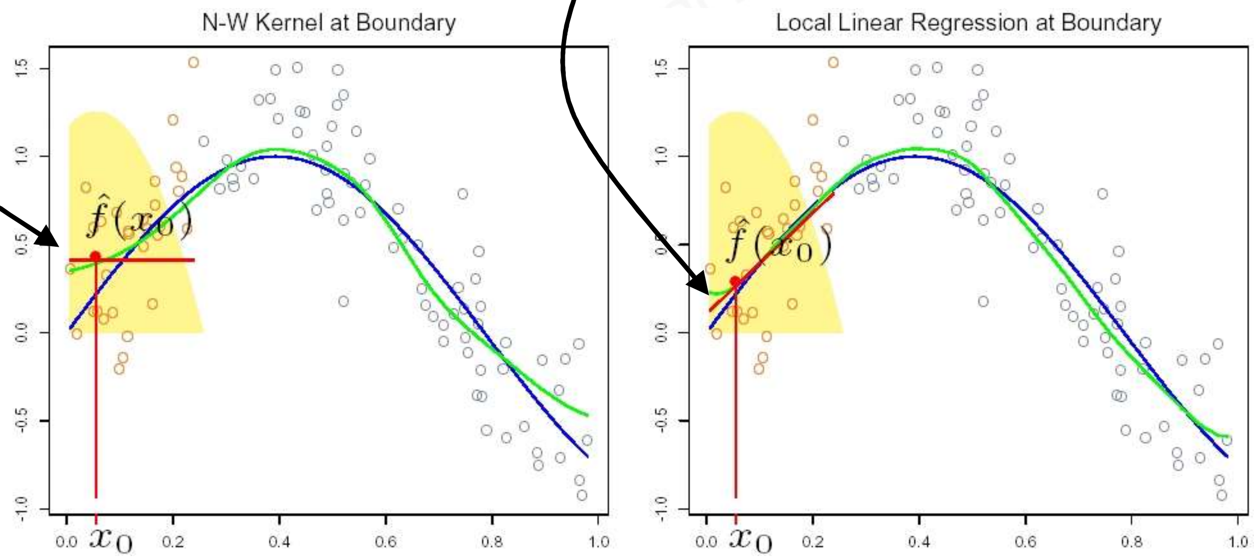
$$\text{Epanechnikov} : D(t) = \begin{cases} \frac{3}{4}(1 - t^2) & if\ |t| < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Tri-Cube} : D(t) = \begin{cases} (1 - t^3)^3 & if\ |t| < 1 \\ 0 & \text{otherwise} \end{cases}$$



Local Linear Regression

- Boundary issue
  - Badly biased on the boundaries because of the asymmetry of the kernel in the region.
  - Linear fitting remove the bias to first order



Local Linear Regression

- Locally weighted linear regression make a first-order correction
- Separate weighted least squares at each target point x_0:

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

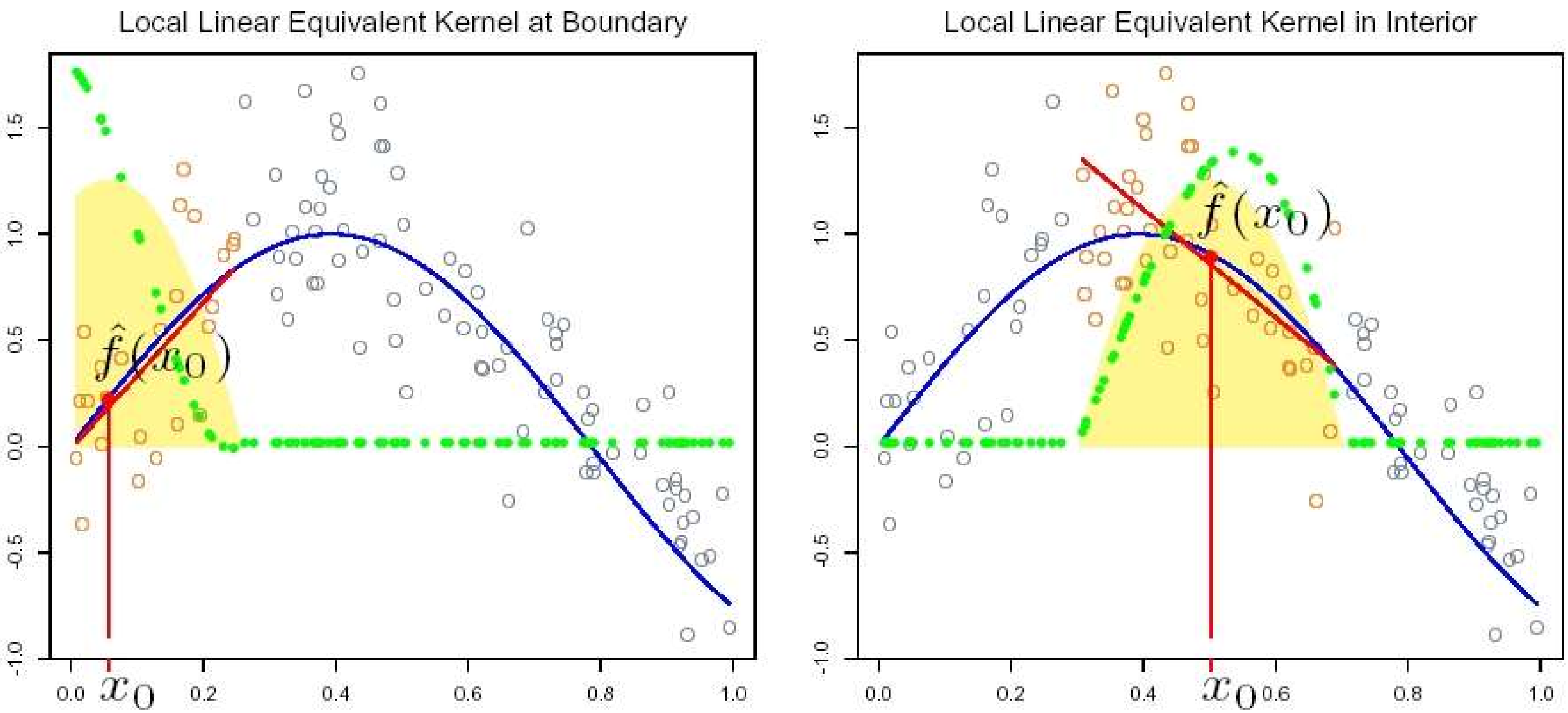
- The estimate: $\hat{f}(x_0) = \hat{\alpha}(x_0) + \hat{\beta}(x_0)x_0$
- *$b(x)^T=(1,x)$;* B: Nx2 regression matrix with i-th row $b(x_i)^T$;

$$W_{N\times N}(x_0) = diag\left(K_\lambda(x_0, x_i)\right), i = 1, \ldots, N$$

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

Local Linear Regression

- The weights $l_i(x_0)$ combine the weighting kernel $K_\lambda(x_0,\cdot)$ and the least squares operations——Equivalent Kernel



Local Linear Regression

- The expansion for $E\hat{f}(x_0)$, using the linearity of local regression and a series expansion of the true function f around x_0

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

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

For local regression

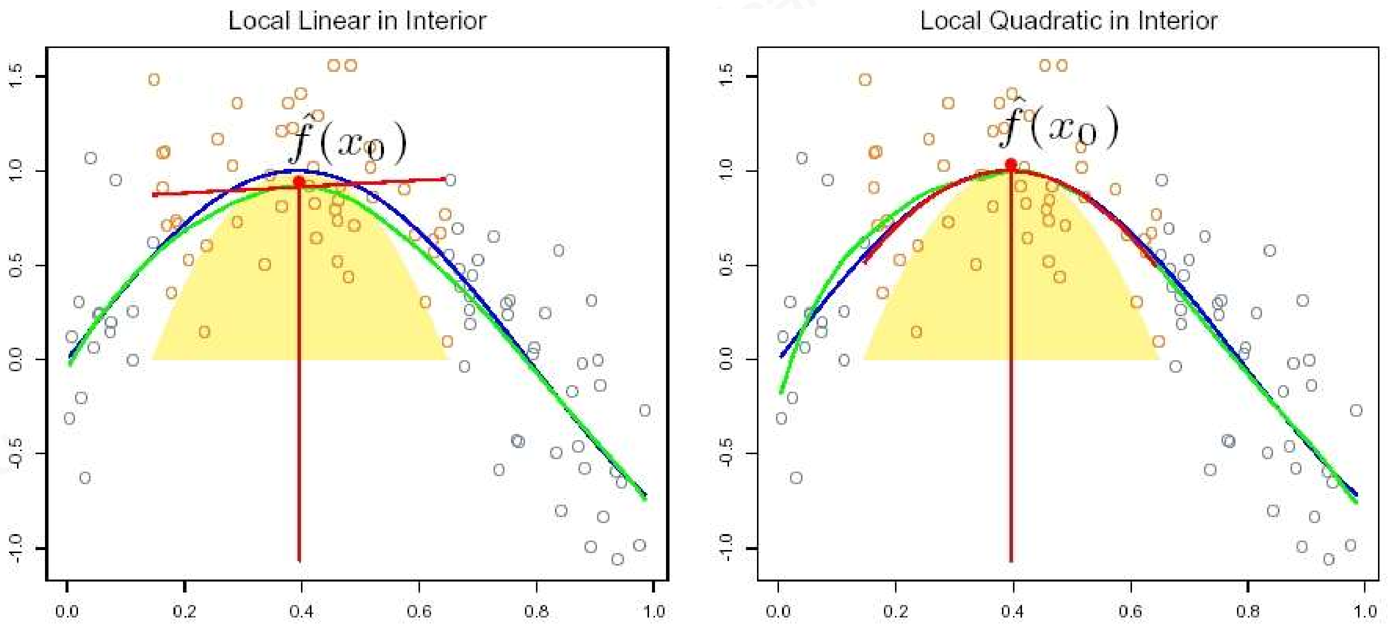
- The bias $E\hat{f}(x_0) - f(x_0)$ depends only on quadratic and higher-order terms in the expansion of f.

Local Polynomial Regression

- Fit local polynomial fits of any degree d

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

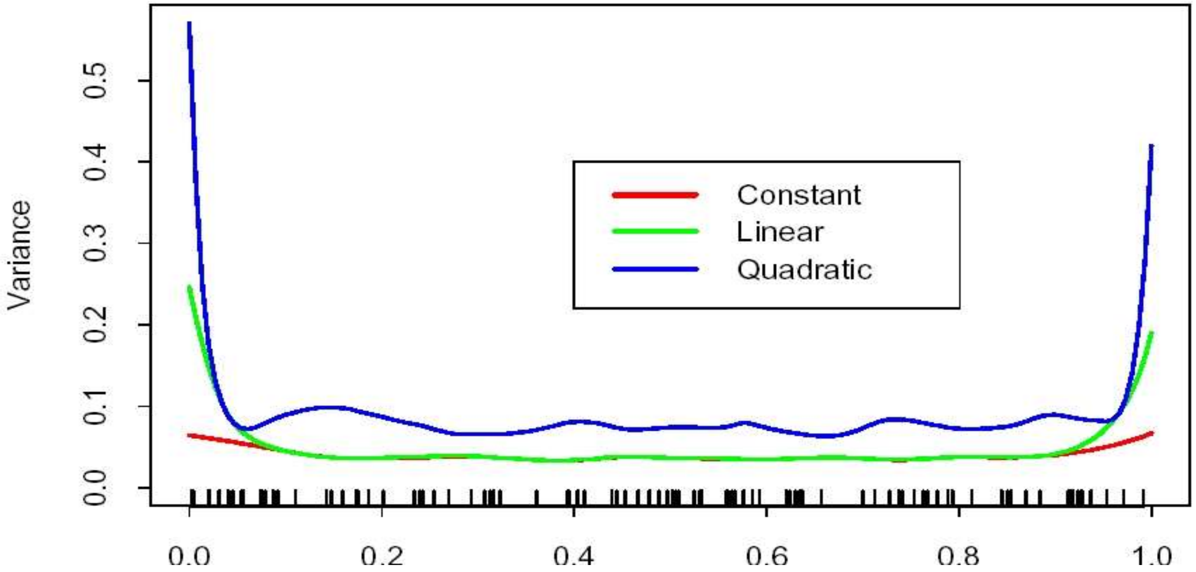
$$\hat{f}(x_0)=\hat{\alpha}(x_0)+\sum_{j=1}^{d}\hat{\beta}_j(x_0)x_0^j$$



Local Polynomial Regression

- Bias only have components of degree d+1 and higher.
- The reduction for bias costs the increased variance.

$$\mathrm{var}(\hat{f}(x_0)) = \sigma^2 \|l(x_0)\|^2, \boxed{\|l(x_0)\| \text{ increases with } d}$$



Kernel Width Selection

- λ is a parameter in kernel K_λ, which controls the width of the kernel.
  - λ takes the radius of supporting region for compact supporting kernel;
  - λ takes the variance for Gaussian Kernel;
  - λ takes k/N for KNN method.
- Kernel width is related to model selection
  - Wide width leads to large bias and small var.
  - Narrow width leads to small bias and large var.

Structured Local Regression

- Structured kernels

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

  – Introduce structure by imposing appropriate restrictions on A

- Structured regression function

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

  – Introduce structure by eliminating some of the higher-order terms

Local Likelihood & Other Models

- Any parametric model can be made local:
  - Parameter associated with y_i : $\theta_i = \theta(x_i) = x_i^T \beta$
  - Log-likelihood: $l(\beta) = \sum_{i=1}^{N} l(y_i, x_i^T \beta)$
  - Model $\theta(X)$ likelihood local to x_0 :

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

  - A varying coefficient model $\theta(z)$

$$l(\theta(z_0)) = \sum_{i=1}^{N} K_\lambda(z_0, z_i) l(y_i, \eta(x_0, \theta(z_0)))$$

$$e.g. \quad \eta(x, \theta) = x^T \theta$$

Local Likelihood & Other Models

- Logistic Regression

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

  – Local log-likelihood for the J class model

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

  – Center the local regressions at

$$\hat{\Pr}(G=j \mid X=x)=\frac{\exp(\hat\beta_{j0}(x_0))}{1+\sum_{k=1}^{J-1}\exp(\hat\beta_{k0}(x_0))}$$

Kernel Density Estimation

- A natural local estimate

$$\hat{f}_X(x_0) = \frac{\# x_i \in N(x_0)}{N\lambda}$$

- The smooth Parzen estimate

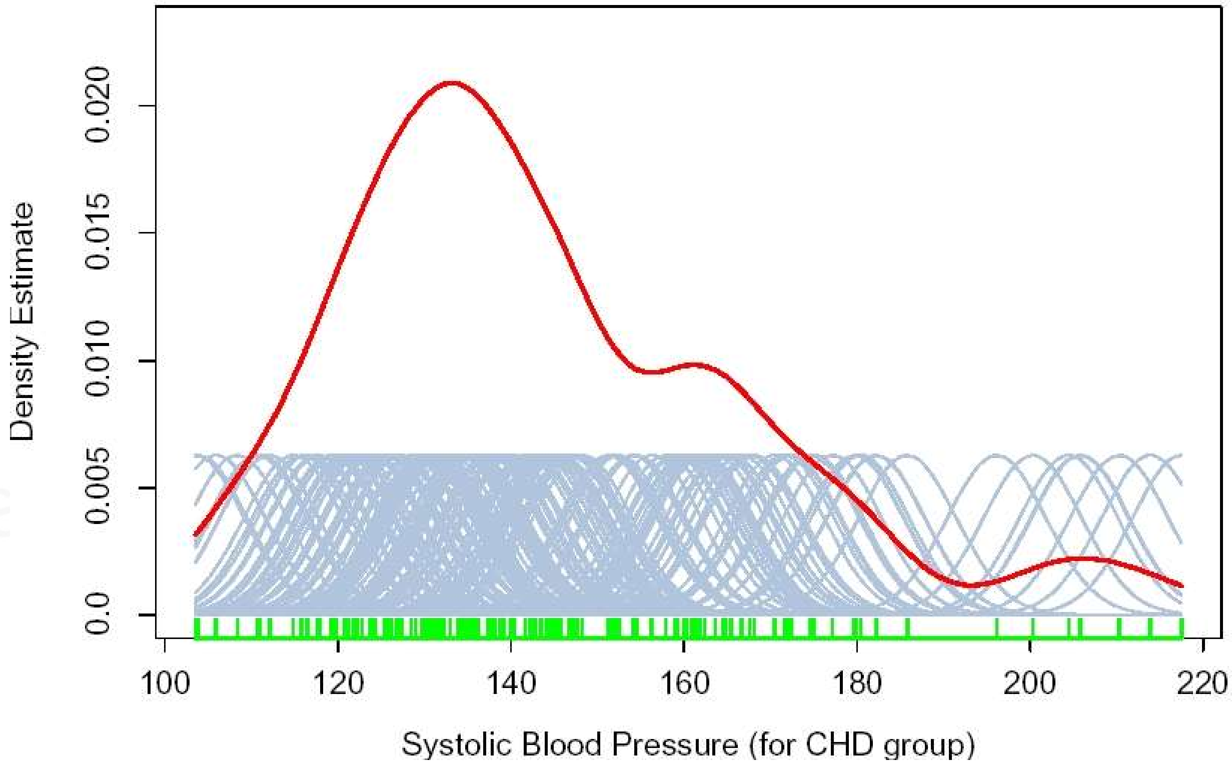
$$\hat{f}_X(x_0) = \frac{1}{N\lambda}\sum_{i=1}^{N} K_\lambda(x_0, x_i)$$

  - For Gaussian kernel $K_\lambda(x_0, x_i)/\lambda = \phi(x_i - x_0)$
  - The estimate become

$$\hat{f}_X(x) = \frac{1}{N}\sum_{i=1}^{N} \phi_\lambda(x_i - x_0)$$
$$= \frac{1}{N(2\lambda^2\pi)^{p/2}}\sum_{i=1}^{N} \exp(-\frac{1}{2}(\|x_i - x_0\|/\lambda)^2)$$

Kernel Density Estimation

- A kernel density estimate for systolic blood pressure. The density estimate at each point is the average contribution from each of the kernels at that point.

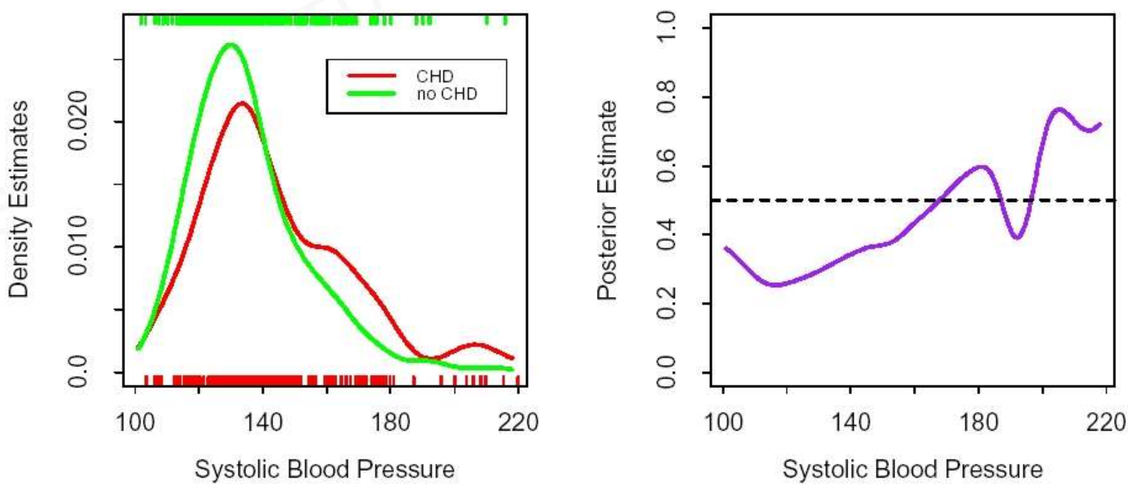


Kernel Density Classification

- Bayes' theorem: $$\hat{\Pr}\left(G = j \middle| X = x_0\right) = \frac{\hat{\pi}_j \hat{f}_j(x_0)}{\sum_{k=1}^{J} \hat{\pi}_k \hat{f}_k(x_0)}$$

$$f_j(x) = \Pr(X = x | G = j)$$

- The estimate for CHD（Coronary heart disease 冠心病） uses *a Gaussian kernel density estimate*.



Kernel Density Classification

- The population class densities and the posterior probabilities

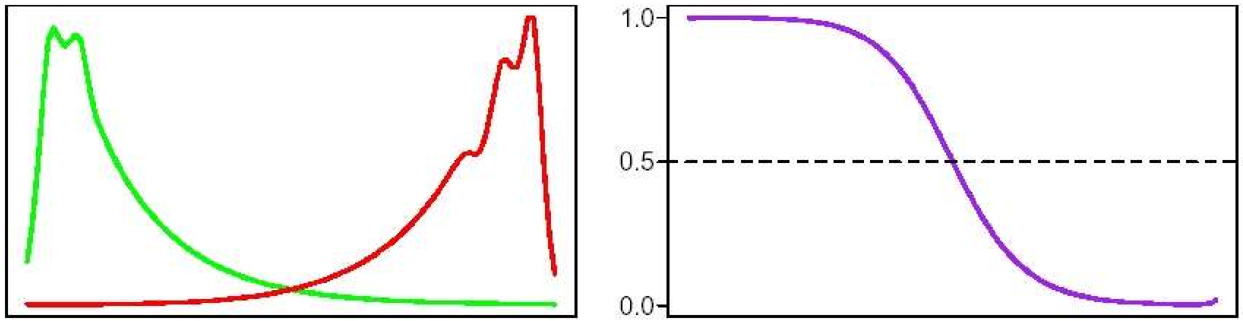


Figure 6.15: *The population class densities may have interesting structure (left) that disappears when the posterior probabilities are formed (right).*

Naïve Bayes

- Naïve Bayes model assume that given a class G=j, the features X_k are independent:

$$f_j(X) = \prod_{k=1}^{p} f_{jk}(X_k)$$

  - $\hat{f}_{jk}(X_k)$ is kernel density estimate, or Gaussian, for coordinate X_k in class j.
  - If X_k is categorical, use Histogram.

$$\text{logit}\frac{\Pr(G=\ell \mid X)}{\Pr(G=J \mid X)} = \log\frac{\pi_\ell f_\ell(X)}{\pi_J f_J(X)} = \log\frac{\pi_\ell \prod_{k=1}^{p} f_{\ell k}(X_k)}{\pi_J \prod_{k=1}^{p} f_{Jk}(X_k)}$$

$$= \log\frac{\pi_\ell}{\pi_J} + \sum_{k=1}^{p} \log\frac{f_{\ell k}(X_k)}{f_{Jk}(X_k)} = \alpha_\ell + \sum_{k=1}^{p} g_{\ell k}(X_k)$$

Radial Basis Function & Kernel

- Radial basis function combine the local and flexibility of kernel methods.

$$f(x) = \sum_{j=1}^{M} K_{\lambda_j}(\xi_j, x)\beta_j = \sum_{j=1}^{M} D\left(\frac{\left\|x-\xi_j\right\|}{\lambda_j}\right)\beta_j$$

  - Each basis element is indexed by a location or prototype parameter ξ_j and a scale parameter λ_j
  - D , a pop choice is the standard Gaussian density function.

Radial Basis Function & Kernel

- For simplicity, focus on least squares methods for regression, and use the Gaussian kernel.
- RBF network model:

$$\min_{\{\lambda_j,\xi_j,\beta_j\}_1^M} \sum_{i=1}^{N}\left(y_i - \beta_0 - \sum_{j=1}^{M}\beta_j \exp\left\{ -\frac{(x_i-\xi_j)^T(x_i-\xi_j)}{\lambda_j^2} \right\} \right)^2$$

- Estimate the $\{\lambda_j, \xi_j\}$ separately from the β_j .
- A undesirable side effect of creating holes——regions of IR^p where none of the kernels has appreciable support.

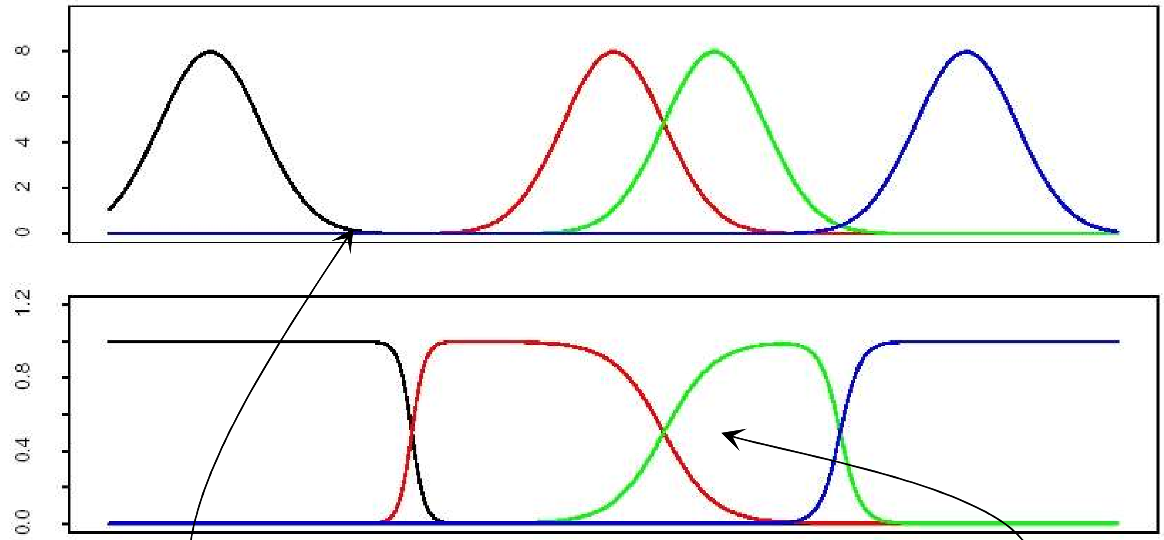
Radial Basis Function & Kernel

- Renormalized radial basis functions.

$$h_j(x) = \frac{D\left(\|x - \xi_j\| / \lambda\right)}{\sum_{k=1}^{M} D\left(\|x - \xi_k\| / \lambda\right)}$$

- The expansion in renormalized RBF

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



Gaussian radial basis function with fixed width can leave holes.
Renormalized Gaussian radial basis function produce basis functions similar in some respects to B-splines.

Mixture Models & EM

- Gaussian Mixture Model $f(x) = \sum_{m=1}^{M} \alpha_m \phi(x; \mu_m, \Sigma_m)$
  - α_m are mixture proportions, $\sum_{m=1}^{M} \alpha_m = 1$
- EM algorithm for mixtures
  - Given $x_1, x_2, \ldots, x_n,$ log-likelihood:

$$l(y,\theta) = \sum_{i=1}^{N} \log\left[\alpha\phi_{\theta_1}(x_i) + (1-\alpha)\phi_{\theta_2}(x_i)\right]$$

Bad

  - Suppose we observe Latent Binary

$$L(x,z,\theta) = \sum_{\substack{i=1 \\ z_i=1}}^{N} \log\left[\alpha\phi_{\theta_1}(x_i)\right] + \sum_{\substack{i=1 \\ z_i=0}}^{N} \log\left[(1-\alpha)\phi_{\theta_2}(x_i)\right]$$

z such that $z = 1 \Rightarrow x \sim \phi_{\theta_1}, \quad z = 0 \Rightarrow x \sim \phi_{\theta_2}$

Good

Mixture Models & EM

- Given θ^0 , compute

$$\theta^* = \text{var}\max \tilde{\ell}(\theta) = E[\ell(x, z, \theta) \mid (\theta^0, y)]$$

- For each sample

$$E(z_i \mid x_i, \theta^0) = \frac{\hat{\alpha}\phi_{\hat{\theta}_1}(x_i)}{\hat{\alpha}\phi_{\hat{\theta}_1}(x_i) + (1-\hat{\alpha})\phi_{\hat{\theta}_2}(x_i)} = w_i$$

$$\ell(\theta) = \sum_{i=1}^{N} w_i \log\left[\hat{\alpha}\phi_{\theta_1}(x_i)\right] + (1-w_i)\log\left[(1-\alpha)\phi_{\theta_2}(x_i)\right]$$

The EM Algorithm

- The EM algorithm for two-component Gaussian mixtures
  - Take initial guesses $\hat{\pi}, \hat{\mu}_1, \hat{\theta}_1, \hat{\mu}_2, \hat{\theta}_2$ for the parameters
  - Expectation Step: Compute the responsibilities

$$\hat{\gamma}_i = \frac{\hat{\pi}\phi_{\hat{\theta}_2}(y_i)}{(1-\hat{\pi})\phi_{\hat{\theta}_1}(y_i) + \hat{\pi}\phi_{\hat{\theta}_2}(y_i)}, \quad i = 1, \ldots, N$$

- $$E(z_i \mid x_i, \theta^0) = \frac{\hat{\alpha}\varphi_{\hat{\theta}_1}(x_i)}{\hat{\alpha}\varphi_{\hat{\theta}_1}(x_i) + (1-\hat{\alpha})\varphi_{\hat{\theta}_2}(x_i)} = w_i$$

$$\ell(\theta) = \sum_{i=1}^{N} w_i \log\left[\hat{\alpha}\varphi_{\theta_1}(x_i)\right] + (1-w_i)\log\left[(1-\alpha)\varphi_{\theta_2}(x_i)\right]$$

The EM Algorithm

– Maximization Step: Compute the weighted means and variances

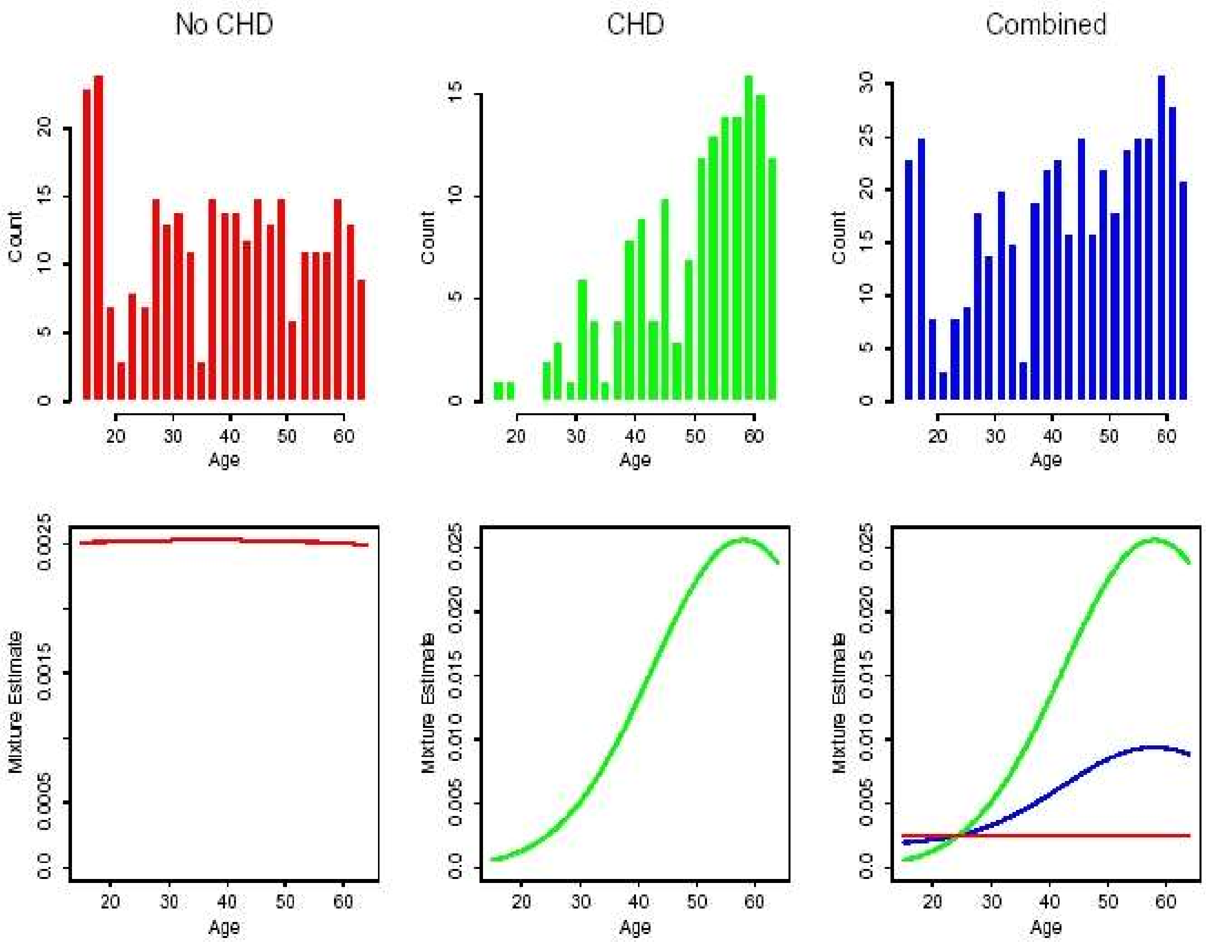
$$\hat{\mu}_1 = \frac{\sum_{i=1}^{N}(1-\hat{\gamma}_i)y_i}{\sum_{i=1}^{N}(1-\hat{\gamma}_i)}, \qquad \hat{\sigma}_1^2 = \frac{\sum_{i=1}^{N}(1-\hat{\gamma}_i)(y_i-\hat{\mu}_1)^2}{\sum_{i=1}^{N}(1-\hat{\gamma}_i)},$$

$$\hat{\mu}_2 = \frac{\sum_{i=1}^{N}\hat{\gamma}_i y_i}{\sum_{i=1}^{N}\hat{\gamma}_i}, \qquad \hat{\sigma}_2^2 = \frac{\sum_{i=1}^{N}\hat{\gamma}_i(y_i-\hat{\mu}_1)^2}{\sum_{i=1}^{N}\hat{\gamma}_i},$$

$$\hat{\pi} = \sum_{i=1}^{N}\hat{\gamma}_i/N$$

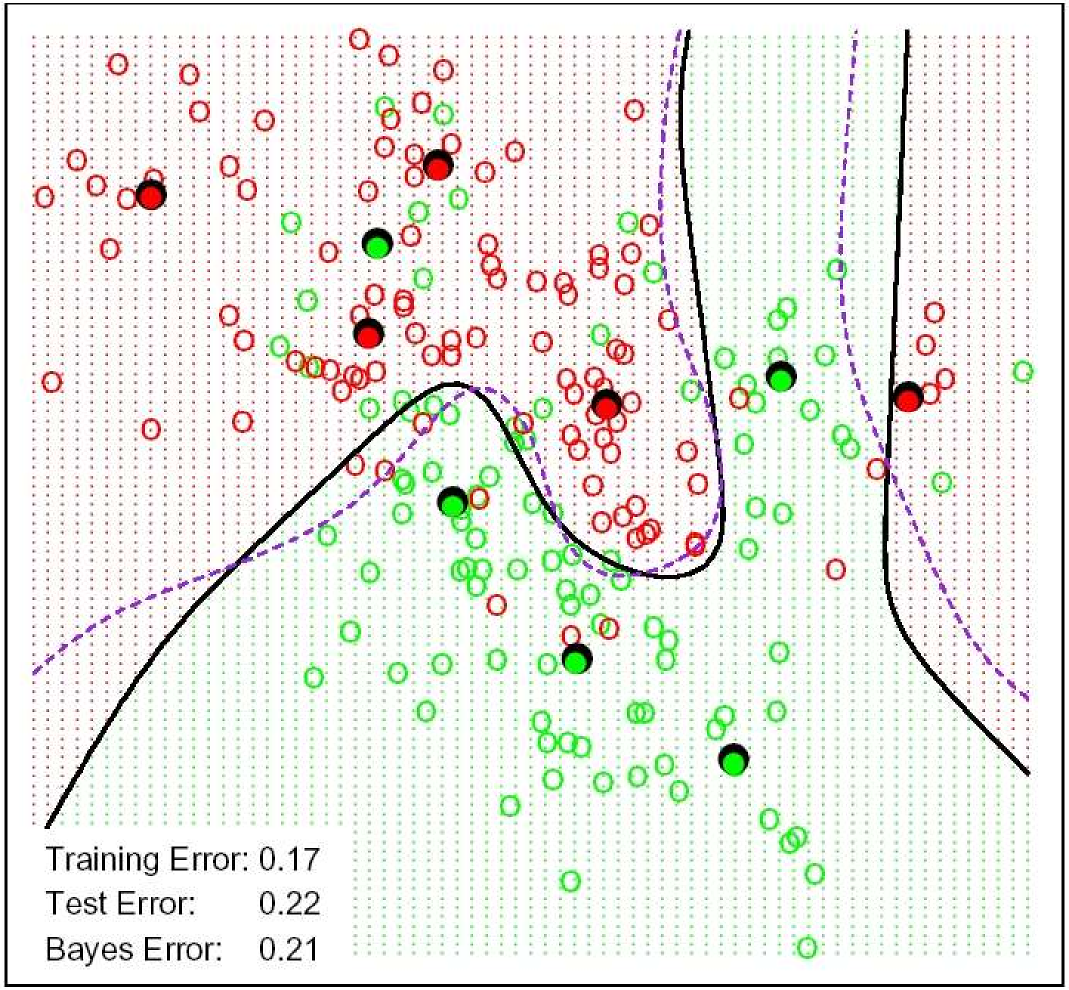
– Iterate 2 and 3 until convergence

Mixture Models & EM



- Application of mixtures to the heart disease risk factor study.

Mixture Models & EM



- Mixture model used for classification of the simulated data

Summary

- To understand why increasing the order of polynomials causes the increase of model variance
- How to kernel method to implement local regression
- Probability density estimation by kernel methods
- EM algorithm for estimating GMM

The End of Talk!