Basis expansions, kernels and support vector machines

Hastie, Tibshirani, Friedman Ch 5, 6, 12

Kevin Murphy Ch. 14

CS 6140
Machine Learning
Professor Olga Vitek

March 23, 2017

Basis Expansion

HTF Ch5

Move beyond linearity

- Linear regression, logistic regression, LDA
  - Classification by linear hyperplanes
  - Easy to fit and to interpret

- $f(Y|\mathbf{X})$ is typically non-linear and non-additive
  - augment X with transformations of $\mathbf{X}$
  - use as input in linear models
  - Denote mth transformation $h_m(\mathbf{X}) : \mathbb{R}^p \to \mathbb{R}^p$
  - Model $f(\mathbf{X}) = \sum_{m=1}^{M} \beta_m h_m(\mathbf{X})$
  - $f(\mathbf{X})$ is linear in $h_m(\mathbf{X})$
  - This is a *linear basis expansion* in $\mathbf{X}$

Linear basis expansion

- Linear model
  - $h_m(\mathbf{X}) = X_m$

- Polynomial terms (Taylor expansion)
  - $h_m(\mathbf{X}) = X_m^2$ or $h_m(\mathbf{X}) = X_i X_j^2$
  - # variables $\uparrow$ exponentially in p
  - tweak one region $\uparrow$ flap another

- Functions of a vector
  - $h_m(\mathbf{X}) = log(X_j)$, $h_m(\mathbf{X}) = \sqrt{X_j}$, $h_m(\mathbf{X}) = ||\mathbf{X}||$

- Indicators $h_m(\mathbf{X}) = I(L_m \leq X_k < U_m)$
  - Break the range of X_k into regions
  - Piecewise constant model in each region

- Piece-wise polynomials and splines
  - Dictionary $\mathcal{D}$ of basis functions
  - Method for controlling model complexity: *restriction, selection, regularization*
  - Restriction: $f(\mathbf{X}) = \sum_{j=1}^{p} f_j(X_j) = \sum_{j=1}^{p} \sum_{m=1}^{M_j} \beta_{jm} h_{jm}(X_j)$

Piecewise fits

- Assume $\mathbf{X}$ is one-dimensional (i.e., X)
  - Divide domain(X) into contiguous intervals
  - $f(X)$: a separate polynomial in each interval

- Example: piecewise constant
  - $h_1(X) = I(X < \xi_1),\ h_2(X) = I(\xi_i \le X \le \xi_2),$
    $h_3 = I(\xi_2 \le X)$
  - $f(X) = \sum_{m=1}^{3} \beta_m h_m(X)$
  - $\widehat{\beta}_m = \bar{Y}_m$, the mean of mth region

- Example: piecewise linear
  - Need extra basis $h_{m+3} = h_m(X) \cdot X,\ m = 1, \ldots, 3$

- Example: restricted piecewise linear
  - $f(\xi_1^-) = f(\xi_1^+) \ \rightarrow\ \beta_1 + \xi_1 \beta_4 = \beta_2 + \xi_1 \beta_5$
  - 3 intervals: 4 free parameters out of 6
  - Alternatively: $h_1(X) = 1,\ h_2(X) = X,$
    $h_3(X) = (X - \xi_1)_+,\ h_4(X) = (X - \xi_2)_+$

Piecewise fit

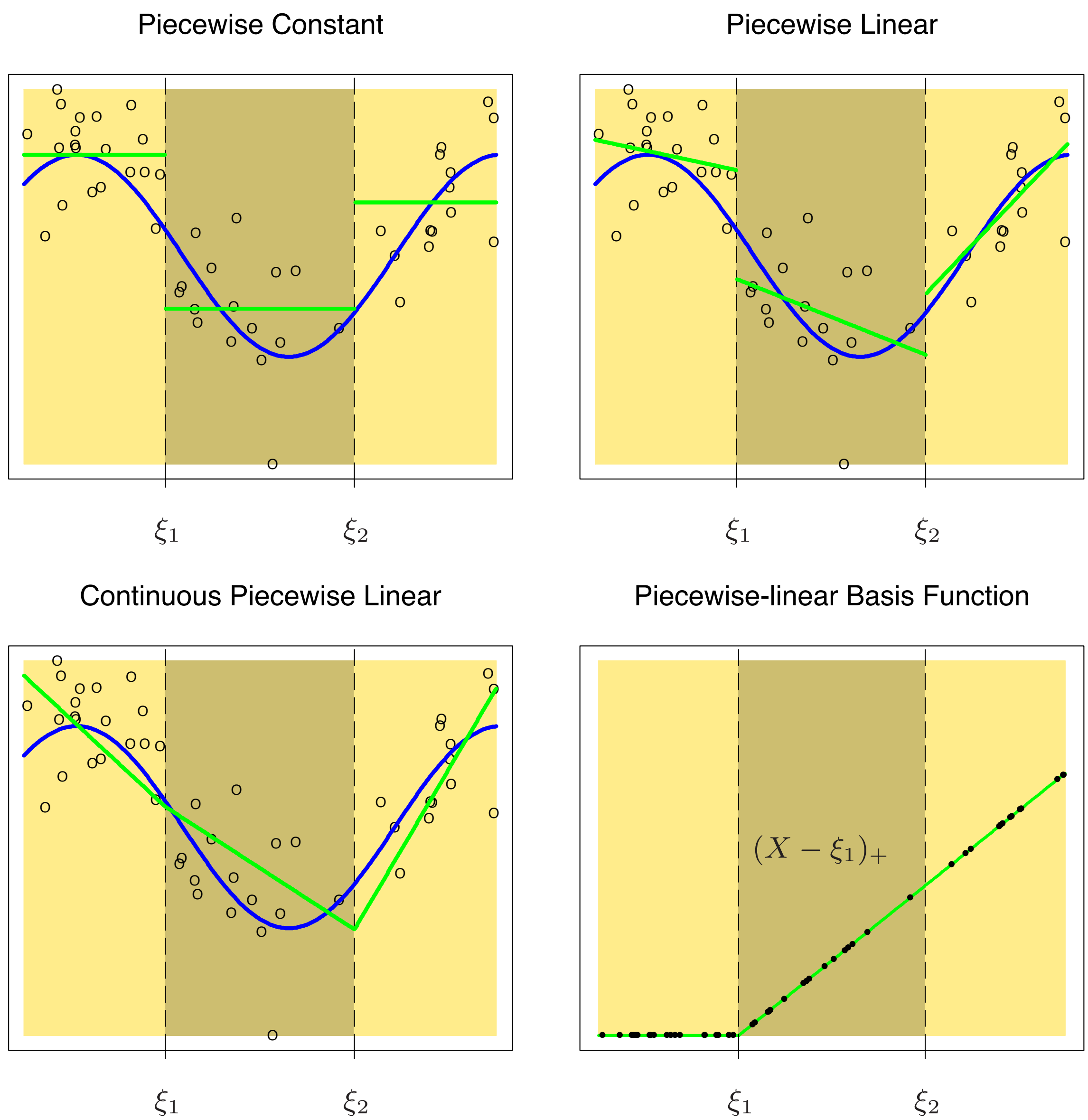


Fig. 5.1. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

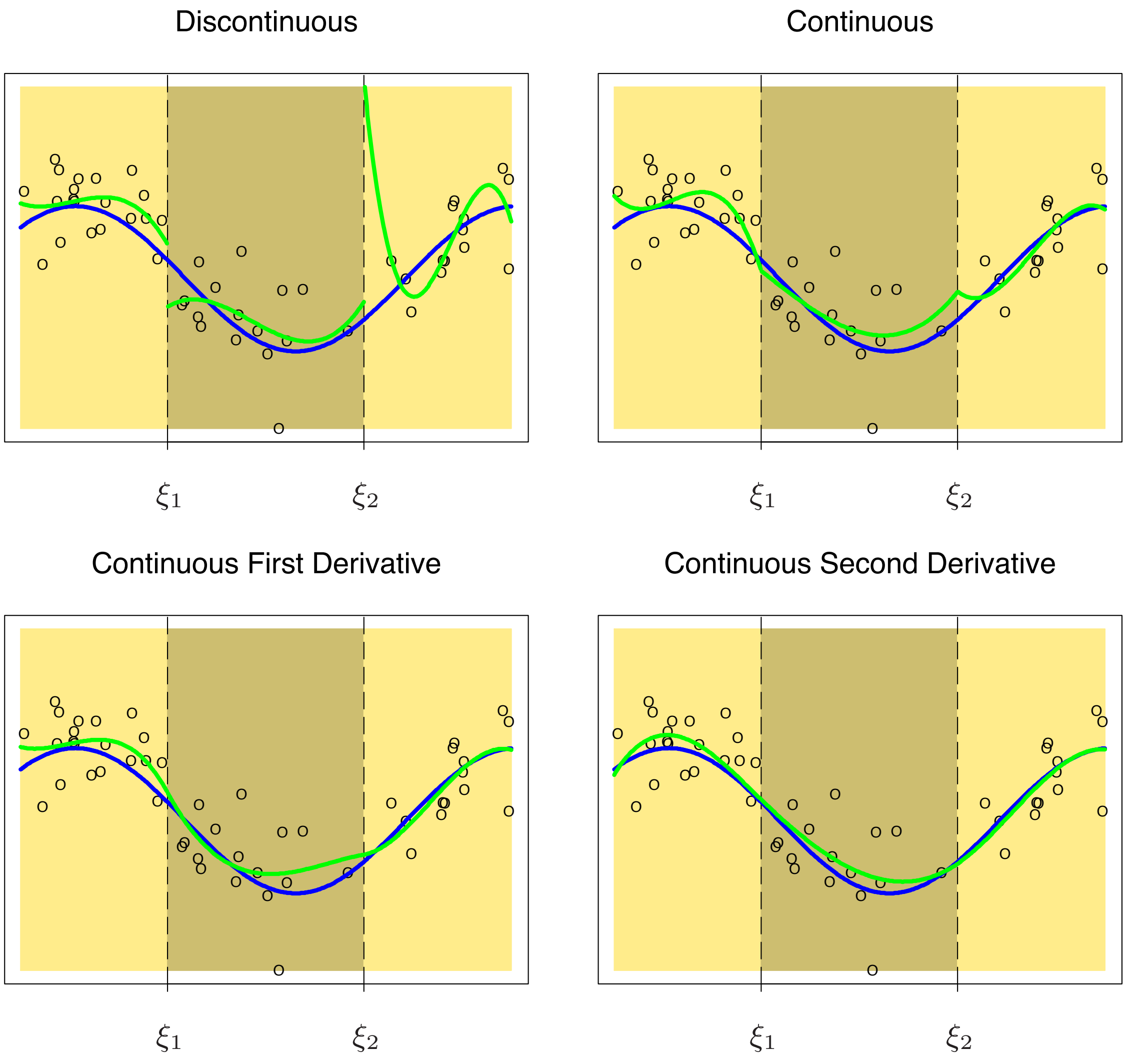
Splines

- Smoother functions
  - Increase the order of polynomials
  - Cubic splines: continuous to second derivative
  - $h_1(X) = 1, h_s(X) = X, h_3(X) = X^2, h_4(X) = X^3$
    $h_5(X) = (X - \xi_1)_+^3,\ h_6(X) = (X - \xi_2)_+^3$

- Example on the next page
  - 6 basis functions
  - 6-dimensional linear space of functions
  - (3 regions) $\times$ (4 parameters per region)
    - (2 knots) $\times$ (3 constrains per knot) $= 6$

- Order-M spline with knots $\xi_j,\ j = 1, \ldots, K$
  - Piecewise polynomial up to order M
  - Has continuous derivatives up to order $M - 2$.
  - Piecewise constant fit is order-1 spline.
    Piecewise continuous fit is order-2 spline.
    Cubic spline is order-4 spline.
  - Basis set: $h_j(X) = X^{j-1},\ j = 1, \ldots, M$ and
    $h_{M+l}(X) = (X - \xi_l)_+^{M-1},\ l = 1, \ldots, K$

Piecewise cubic polynomials



Cubic spline is the lowest-order spline for which the knot-discontinuity is not visible to human eye

Fig. 5.2. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Natural cubic splines

- Need for extra stability
  - Polynomials and splines have erratic behavior near the boundaries
  - Variance explodes
  - Extrapolation is problematic

- Natural cubic splines: extra constraints
  - Linear functions beyond boundary knots
  - K knots $=$ K basis functions
  - Basis for cubic splines $\rightarrow$ impose constraints
  - Start from basis set, impose boundary constraint, derive reduced basis:

$$N_1(X) = 1,\ N_2(X) = X,$$
$$N_{k+2}(X) = d_k(X) - d_{K-1}(X)$$
$$\text{where } d_k(X) = \left\{(X - \xi_k)_+^3 - (X - \xi_K)_+^3\right\} / \{\xi_K - \xi_k\}$$

  - Second and third derivatives are 0 for $X \geq \xi_K$

Pointwise variance curve

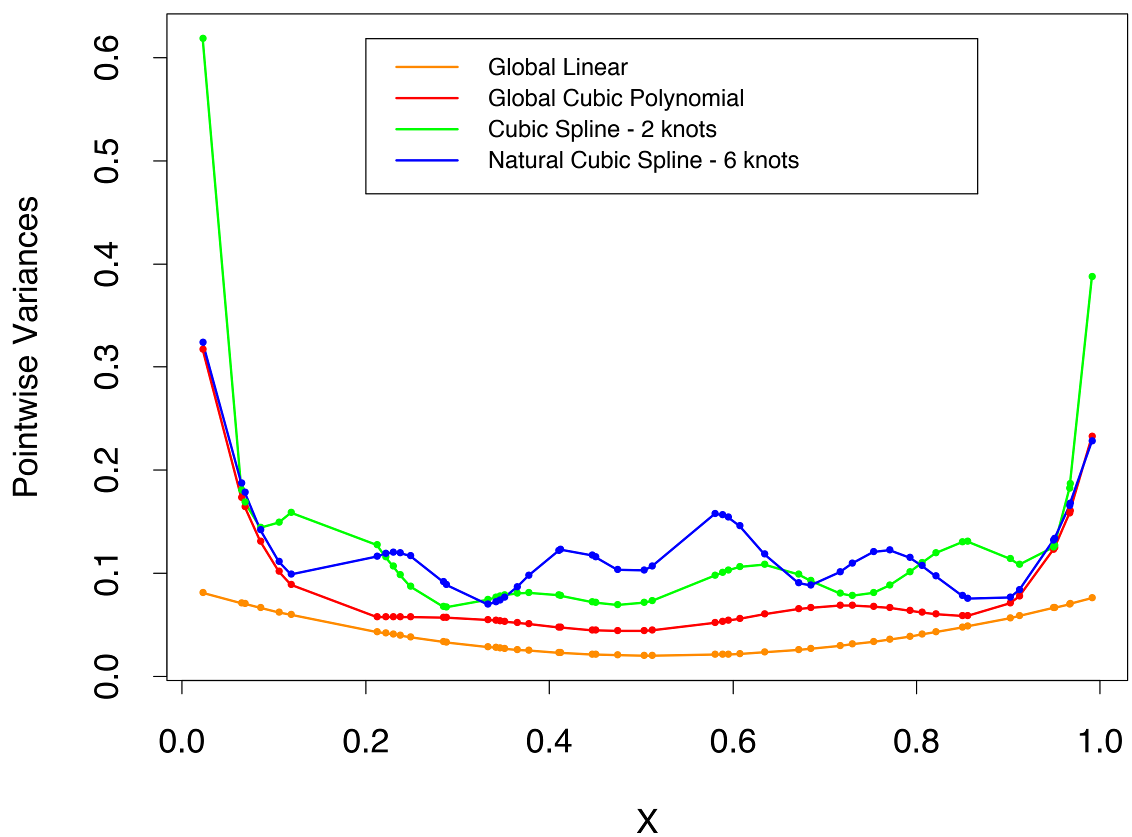


Fig. 5.3. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Example: South African Heart Disease

- Logistic regression

$$\frac{\Pr\{\texttt{chd} = 1|X\}}{1 - \Pr\{\texttt{chd} = 1|X\}} = \theta_0 + h_1(X_1)'\theta_1 + \ldots + h_p(X_p)'\theta_p$$

  - θ_j are vectors of coefficients multiplying the vector of natural spline basis functions h_j
  - Use 4 natural spline basis functions for each term in the model
  - Knots chosen at uniform quantiles of X_j:

    3 internal knots +

    2 boundary knots at the extremes of X_j
  - Binary predictor has a single coefficient
  - More compactly: combine p vectors of basis functions + constant term in a big vector $h(X)$, and model $h(X)'\theta$
  - Backward stepwise selection + AIC to drop terms
  - Plot prediction $\pm 2 \cdot SE$

Fit: $\hat{f}_j(X_j) = h_j(X_j)'\hat{\theta}_j$

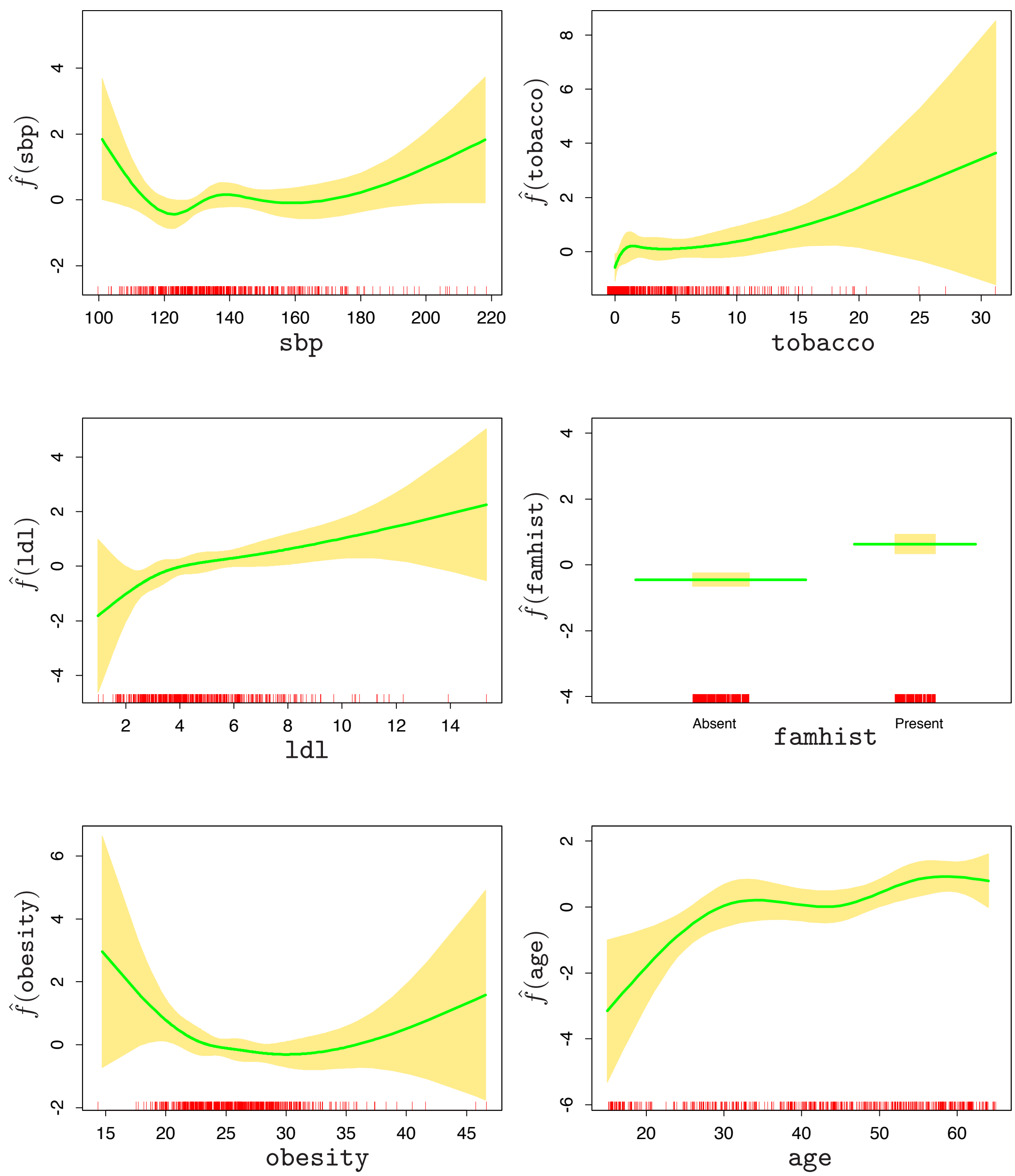
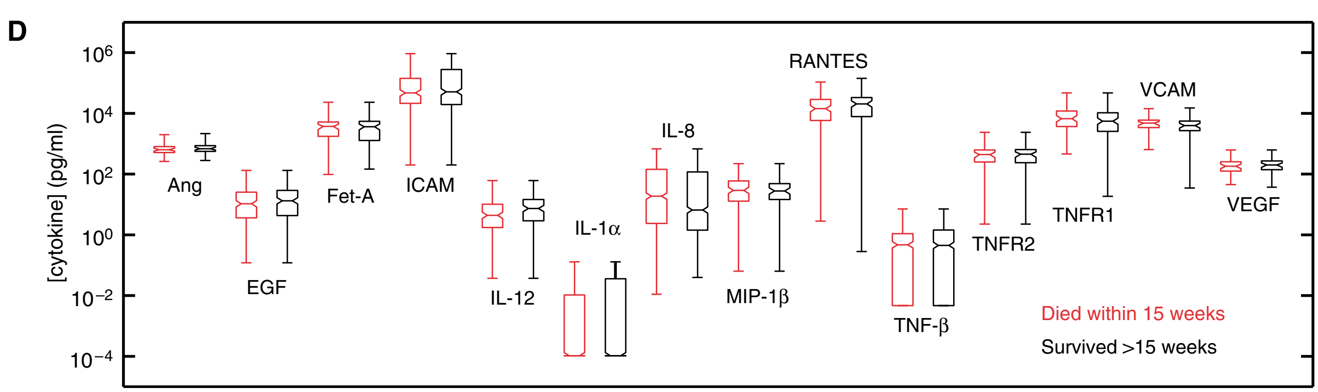


Fig. 5.4. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Example: disease classification

Knickerbocker *et al.* "An integrated approach to prognosis using protein microarrays and nonparametric methods". *Molecular Systems Biology*, 2007

- Goal: predict mortality in patients w/kidney dialysis
  - 468 patients, 208 died within 15 weeks of diagnoses
  - 14 proteins ("messenger" molecule that allows cells to communicate and alter function)
  - 11 clinical characteristics (age, race, bmi...)
  - Proteins were uninformative in isolation



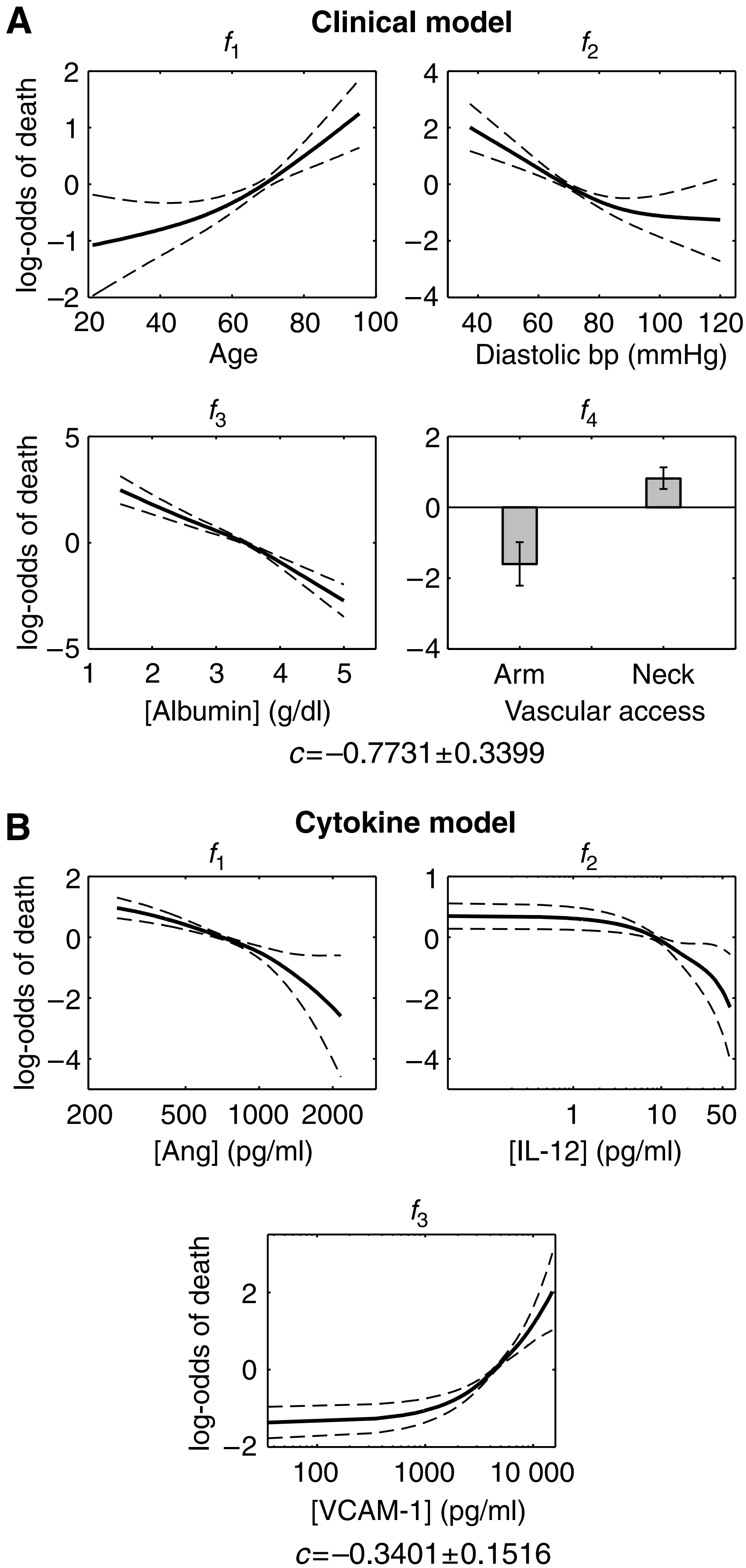
Approach

- Separate predictive model for clinical and molecular measurements
  - Logistic regression (Y = death within 15 weeks)

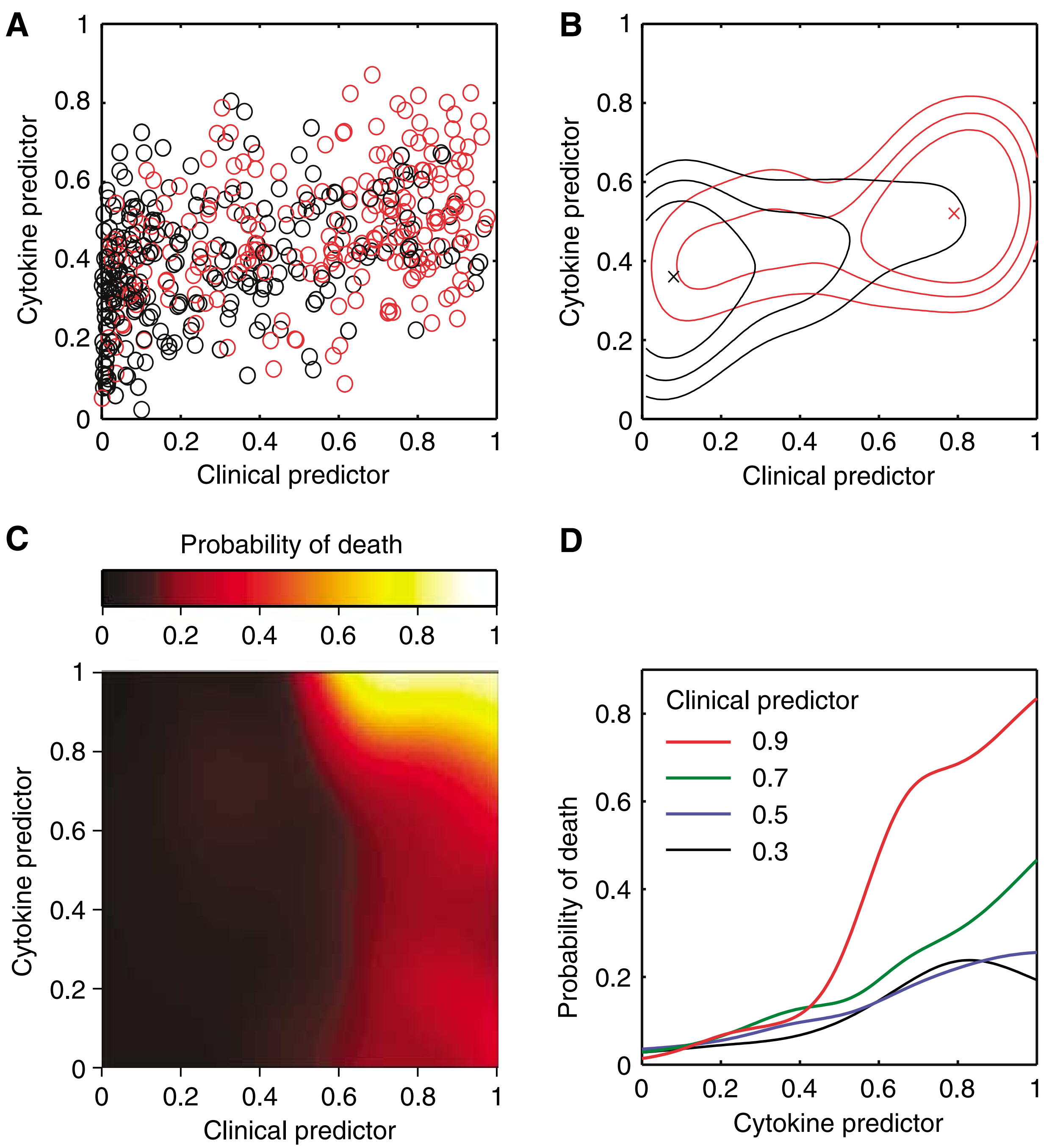
$$\begin{aligned}\text{log-odds of death} &= \log(P_{\text{sample}}(\text{death})/P_{\text{sample}}(\text{survival})) \\ &= c + \sum_{p=1}^{M} b_p x_p\end{aligned}$$

  - Additive model to reduce feature space
  - Exhaustive search for best M-variable model
  - Non-parametric version of the best model: splines

$$\begin{aligned}\log - \text{odds of death} &= \log(P_{\text{sample}}(\text{death})/P_{\text{sample}}(\text{survival})) \\ &= c + \sum_{p=1}^{M} f_p(x_p)\end{aligned}$$


A
Clinical model
f1
f2
log-odds of death
2
1
0
−1
−2
20
40
60
80
100
Age
4
2
0
−2
−4
40
60
80
100
120
Diastolic bp (mmHg)
f3
f4
5
0
−5
1
2
3
4
5
[Albumin] (g/dl)
2
0
−2
−4
Arm
Neck
Vascular access
c=−0.7731±0.3399
B
Cytokine model
f1
f2
2
0
−2
−4
200
500
1000
2000
[Ang] (pg/ml)
1
0
−2
−4
1
10
50
[IL-12] (pg/ml)
f3
2
0
−2
100
1000
10 000
[VCAM-1] (pg/ml)
c=−0.3401±0.1516

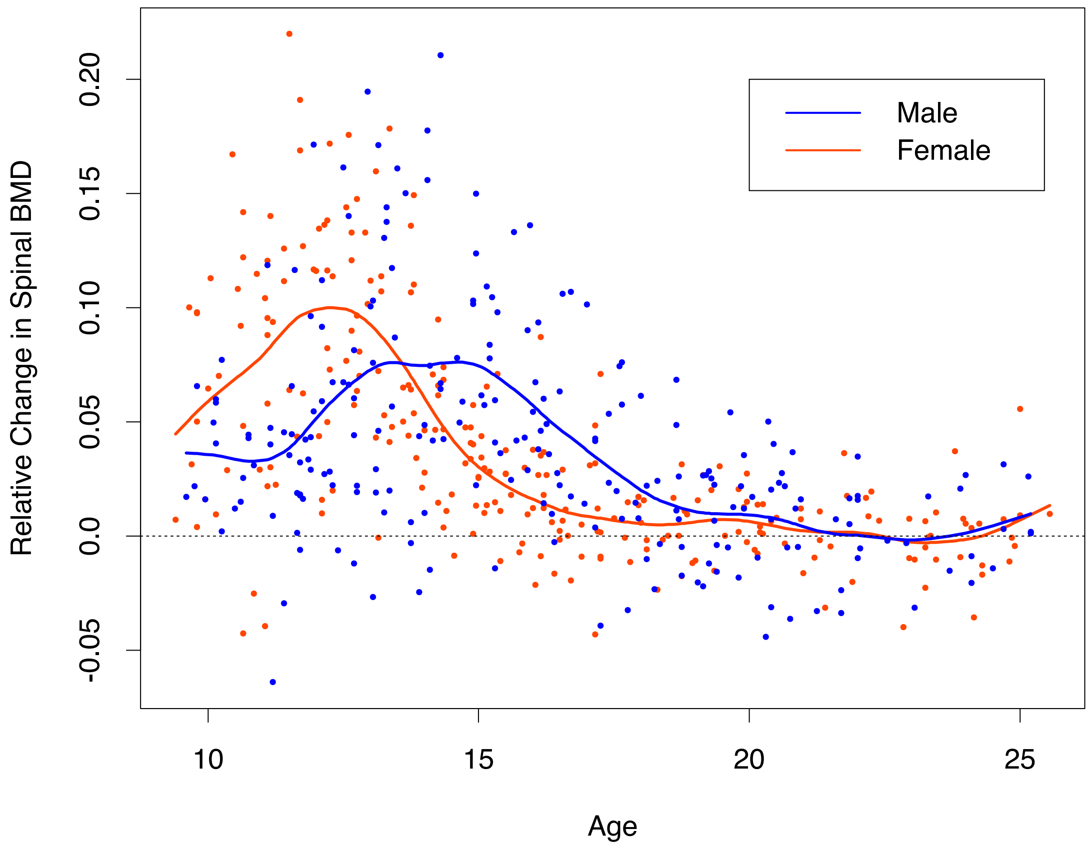
Combined prediction



Controlling model complexity (fixed knots)

- Restriction
  - Limit the class of functions
    $$f(\mathbf{X}) = \sum_{j=1}^{p} f_j(X_j) = \sum_{j=1}^{p} \sum_{m=1}^{M_j} \beta_{jm} h_{jm}(X_j)$$
  - Model complexity $\sim$ number of basis functions
- Selection
  - AIC, BIC, significance testing
  - Adaptively include basis functions that contribute to prediction (include CART, boosting...)
- Regularization
  - Include the entire dictionary of basis functions, but restrict the coefficients
    $$\text{RSS}(f, \lambda) = \sum_{i=1}^{N} \{y_i - f(x_i)\}^2 + \lambda \int \left\{f''(t)\right\}^3 dt$$
  - λ is a smoothing parameter ($\lambda = 0$: any function; $\lambda = \infty$: linear fit)
  - Maximized with a natural cubic spline
  - Equivalent to generalized ridge regression

Example



Rel. change in bone mineral density $\sim$ age, $\lambda = 0.00022$

$$\begin{aligned}\text{EPE}(\hat{f}_\lambda) &= E\left(Y - \hat{f}_\lambda\right)^2 \\ &= \text{Var}(Y) + E\left[\text{Bias}^2(\hat{f}_\lambda) + \text{Var}(\hat{f}_\lambda)\right] \\ &= \sigma^2 + \text{MSE}(\hat{f}_\lambda)\end{aligned}$$

Choose λ by cross-validation

Fig. 5.6. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Fitting multivariate models

Example: linear additive model

$$Y = \alpha + \sum_{j=1}^{p} f_j(X_j) + \varepsilon$$

Algorithm 9.1 *The Backfitting Algorithm for Additive Models.*

---

1. Initialize: $\hat{\alpha} = \frac{1}{N}\sum_1^N y_i$, $\hat{f}_j \equiv 0, \forall i, j$.
2. Cycle: $j = 1, 2, \ldots, p, \ldots, 1, 2, \ldots, p, \ldots,$

$$\hat{f}_j \leftarrow \mathcal{S}_j\left[\{y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik})\}_1^N\right],$$

$$\hat{f}_j \leftarrow \hat{f}_j - \frac{1}{N}\sum_{i=1}^{N} \hat{f}_j(x_{ij}).$$

until the functions $\hat{f}_j$ change less than a prespecified threshold.

$\mathcal{S}_j$ is a cubic spline. Iteratively smooth the residual fit for one predictor at a time, until convergence.

See Algorithm 9.2 for logistic regression

Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Kernel Methods

HTF Ch6

One-dimensional kernel smoothers

- Different model at each point x_0
  - Only use observations close to target x_0
  - Weigh the neighbors x_1 with a kernel $K_\lambda(x_0, x_i)$

  - Weight based on distance from x_0
  - λ is a parameter
  - The resulting function is smooth

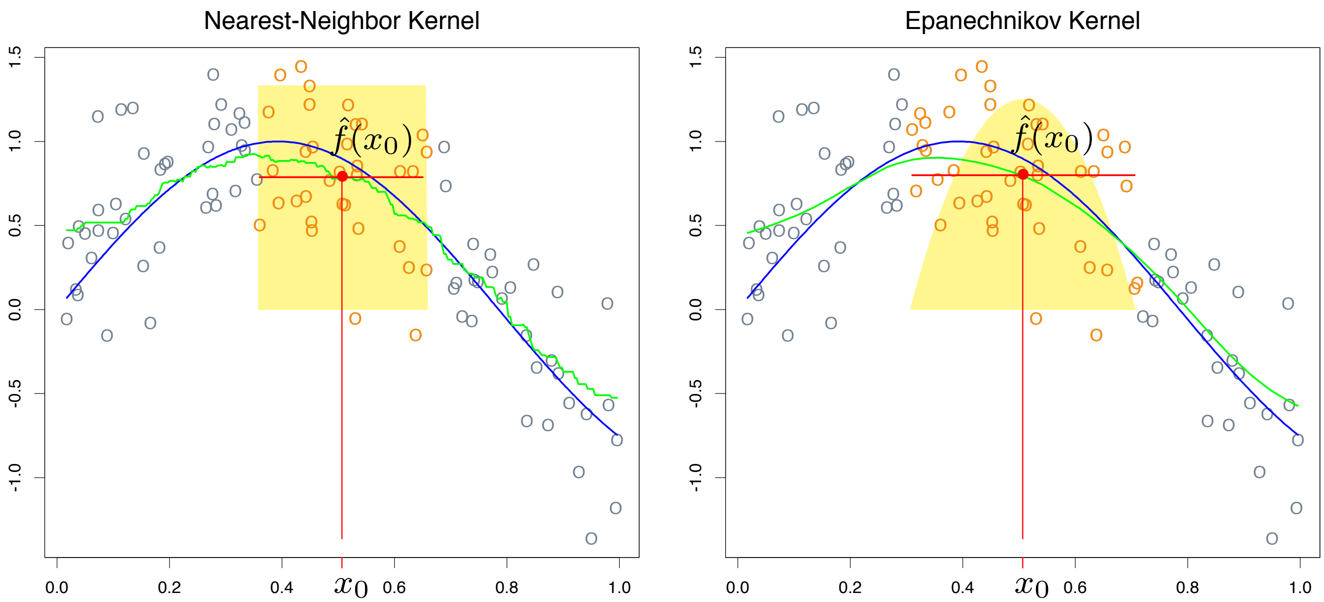
- K nearest neighbors
  - $\hat{f}(x_0) = \text{Ave}(y_i | x_i \in N_k(x_0))$ (discontinuous in x)

- Nadaraya-Watson kernel-weighted average
  - $\hat{f}(x_0) = \frac{\sum_{i=1}^{N} K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^{N} K_\lambda(x_0, x_i)}$, where
    $K_\lambda(x_0, x) = D\left(\frac{|x - x_0|}{\lambda}\right)$ and

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

  - Points near the boundary have weight ~ 0
    $\rightarrow$ smoothing

Example



- 100 pairs (x_i, y_i)
- Green: Left: 30-NN running mean.
  Right: Kernel-weighted average, $\lambda = 0.2$
- Orange: observations contributing to the fit

Fig. 6.1. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Adaptive width

- Define $h_\lambda(x_0)$ a width function that determines the neighborhood of x_0

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

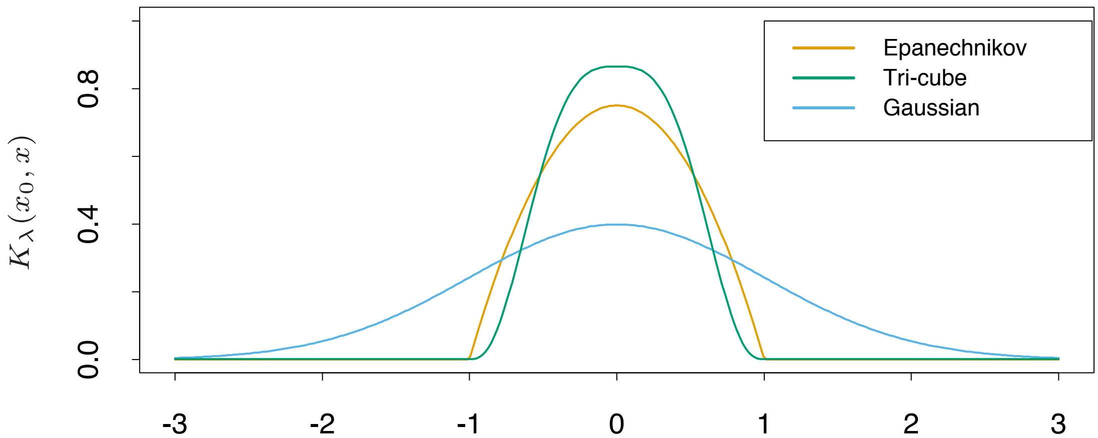
- Concerns:

  - Determine λ. Larger $\lambda \rightarrow$ lower variance but higher bias

  - $h_\lambda(x)$ constant $\rightarrow$ variance inversely proportional to density of points

  - Nearest neighbor $\rightarrow$ bias inversely proportional to density of points

Kernel examples



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

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

- Tri-cube function

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

Fig. 6.2. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Local linear regression

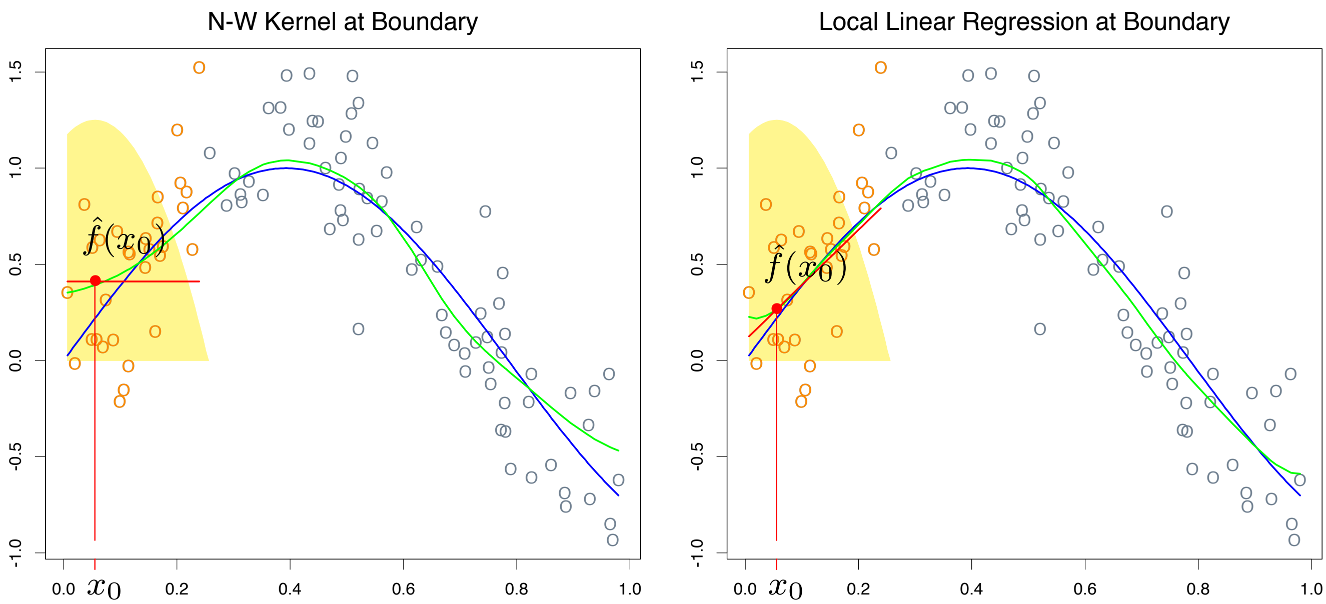
- Smoothly varying locally weighted average
  - Problem: asymmetry of the kernel
    on the boundary of the domain $\rightarrow$ bias
  - Solution: fit straight lines rather than constants
  - Also helps if values of x are unequally spaced

- Separate weighted least squares problem
  - At each x_0: $\displaystyle\min_{\alpha(x_0),\beta(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i)\,[y_i - \alpha(x_0) - \beta(x_0)x_i]^2$
  - Define $b(x)' = (1, x)$
    - $\mathbf{B}$ the $X \times 2$ regression matrix with ith row $b(x)'$
    - $\mathbf{W}(x_0)$ the $N \times N$ diagonal matrix diag$\{K_\lambda(x_0, x_i)\}$.

$$
\begin{aligned}
\hat{f}(x_0) &= b(x_0)' \left(\mathbf{B}'\mathbf{W}(x_0)\mathbf{B}\right)^{-1} \mathbf{B}'\mathbf{W}(x_0)\mathbf{y} \\
&= \sum_{i=1}^{N} l_i(x_0) y_i, \quad l_i(x_0) \text{ is a local function}
\end{aligned}
$$

  - Prediction performed at a single point x_0:
    $\hat{f}(x_0) = \hat{\alpha}(x_0) + \hat{\beta}(x_0)x_0$

Example



- True function is linear. Most observations in the neighborhood exceed the target point.
- Left: locally weighted average:
  bias near the boundaries of X
- Right: locally weighted linear regression removes the bias to first order

Fig. 6.3. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Extensions / comments

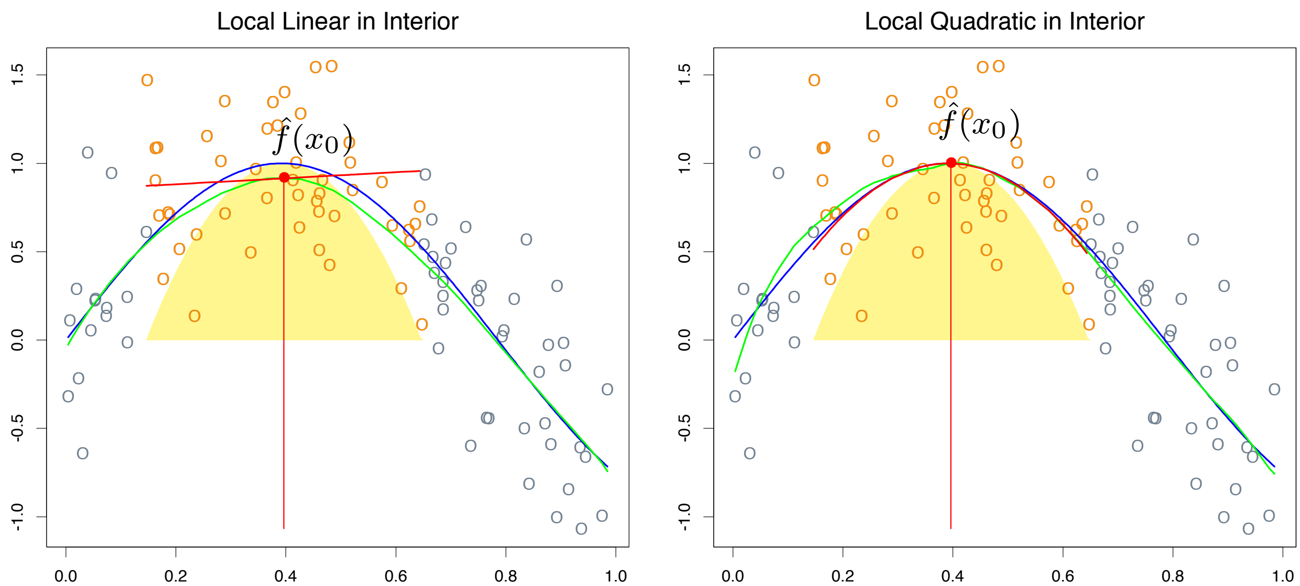
- Local polynomial regression
  - Do not have to stop at linear terms
  - Minimize
    $$\min_{\alpha(x_0),\beta_j(x_0),j=1,\ldots,d} \sum_{i=1}^{N} K_\lambda(x_0,x_i)\left[y_i-\alpha(x_0)-\sum_{j=1}^{d}\beta_j(x_0)x_i^j\right]^2$$
  - Solution $\hat{f}(x_0)=\hat{\alpha}(x_0)+\sum_{j=1}^{d}\hat{\beta}_j(x_0)x_0^j$
  - Avoid "trimming the hills and filling the valleys"

- Bias-variance tradeoff
  - Smaller window $\rightarrow$ lower bias, higher variance

- Default hoice of λ
  - Epanechnikov/tri-cube: radius of support reg.
  - Gaussian: standard deviation
  - Better option: cross-validation

- Model complexity
  - Estimated degrees of freedom $df = trace(\mathbf{S}_\lambda)$, where $\{\mathbf{S}_\lambda\}_{ij} = l_i(x_j)$

Example: polynomial regression

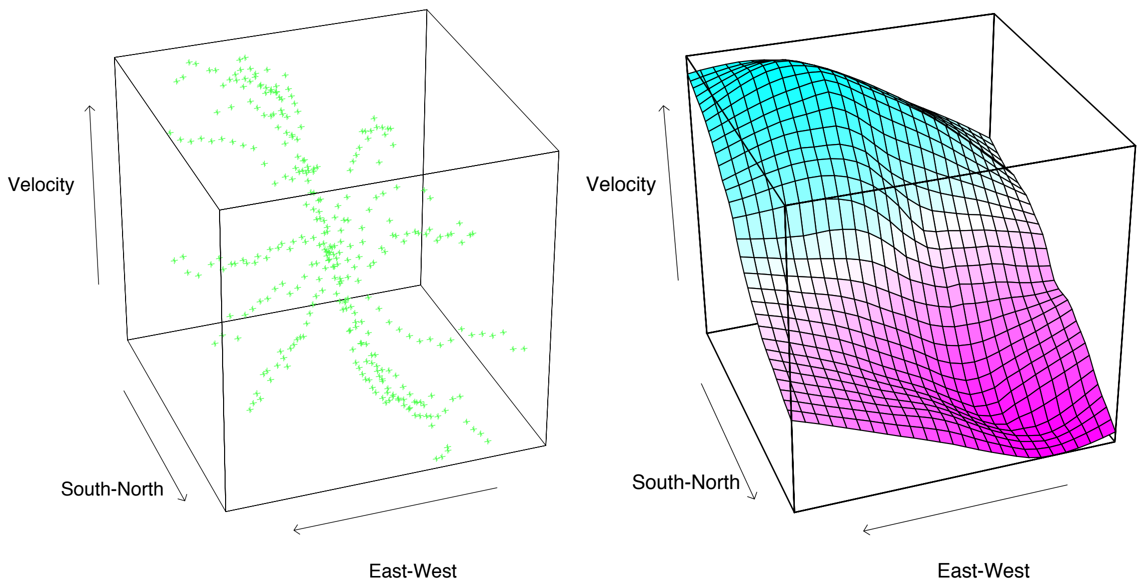


If $Y = f(X) + \varepsilon,\ \varepsilon \overset{iid}{\sim} (0, \sigma^2)$, then

- $Var\{\widehat{f}(x_0)\} = \sigma^2 \,||l(x_0)||$,
  where $l(x_0)$ is vector of kernel weights at x_0
- Can show that $l(x_0)$ increases with the polynomial degree d
- Implies greater bias-variance trade-off

Fig. 6.5. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Local regression in $\mathbb{R}^p$



- p-dimensional Kernel; polynomial fit of degree d
- Define $b(X)'$:
  - $d = 1$, $p = 2$: $(1, X_1, X_2)$
  - $d = 2$, $p = 2$: $(1, X_1, X_2, X_1^2, X_2^2, X_1 X_2)$
- At each x_0, solve
  $\min\limits_{\beta(x_0)} \sum\limits_{i=1}^{N} K_\lambda(x_0, x_i)\,(y_i - b(x_i)'\beta(x_0))^2$, where
  $K_\lambda(x_0, x) = d\left(\frac{||x - x_0||}{\lambda}\right)$

Fig. 6.8. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Extensions

- Structured Kernels
  - As dimensions $\uparrow$, # neighbor points $\downarrow$
  - Need additional constraints on local regression, to counter curse of dimensionality
  - Option: modify the kernel

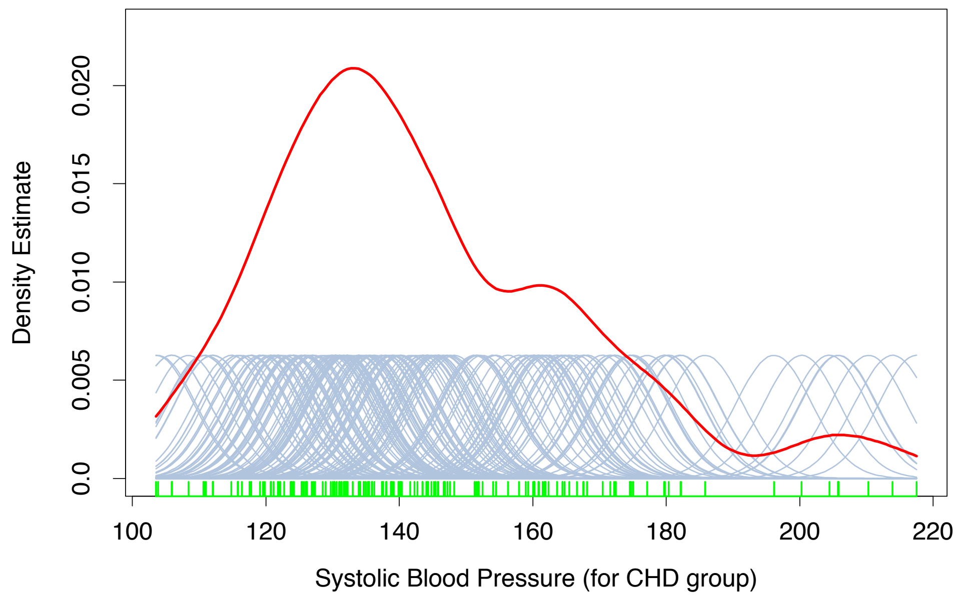
$$K_{\lambda,A}(x_0, x) = d\left(\frac{(x - x_0)'\mathbf{A}(x - x_0)}{\lambda}\right)$$

  - Restrict $\mathbf{A}$ to downgrade or omit directions

- Local likelihood: local fit of any other model
  - E.g., logistic regression

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

Kernel density estimation



- Natural (bumpy) estimate $\hat{f}_X(x_0) = \frac{\# x_i \in \mathcal{N}(x_0)}{N\lambda}$ where $\mathcal{N}(x_0)$ is a neighborhood of x_0 of width λ

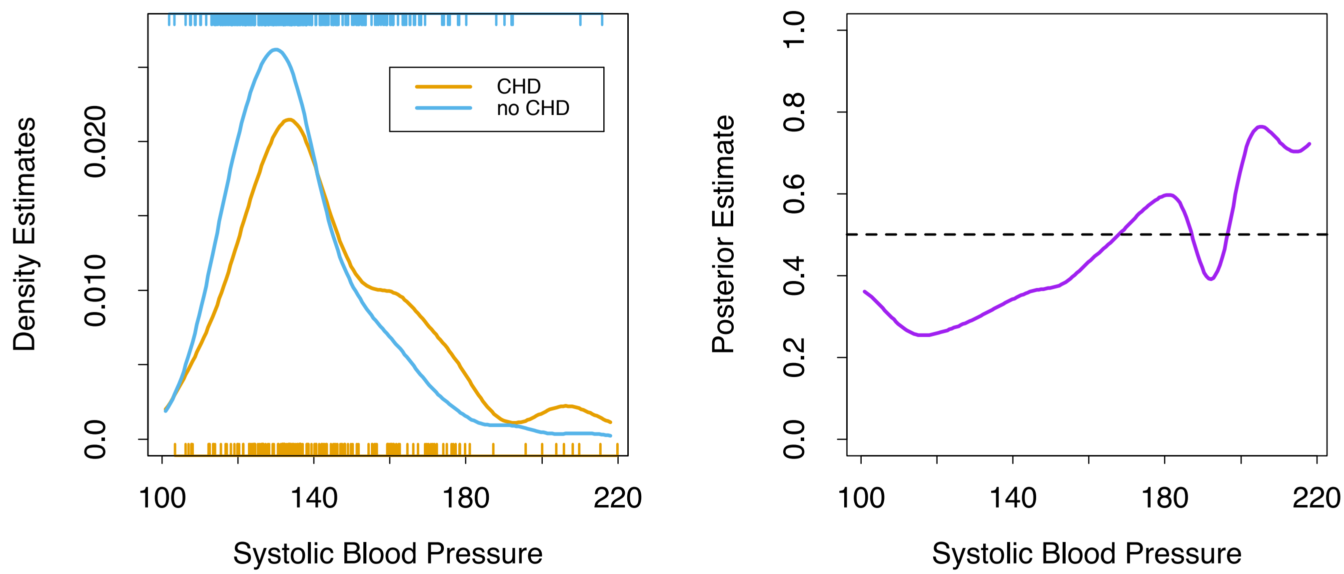
- Smoothed estimate $\hat{f}_X(x_0) = \frac{1}{N\lambda} \sum_{i=1}^{N} K_\lambda(x_0, x_i)$

  E.g., Gaussian kernel $K_\lambda(x_0, x_i) = \phi(|x - x_0|/\lambda)$ where ϕ is pdf of $\mathcal{N}(0, 1)$

- Equivalent to local weighted average

Fig. 6.13. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Kernel density classification



- Nonparametric classification

$$\widehat{Pr}\{G = j | X = x_0\} = \frac{\hat{\pi}_j \hat{f}_j(x_0)}{\sum_{k=1}^{J} \hat{\pi}_k \hat{f}_k(x_0)}$$

- Naïve Bayes

$$\widehat{Pr}\{G = k | X = x_0\} = \frac{\hat{\pi}_k \hat{f}_k(x_0)}{\sum_{j=1}^{J} \hat{\pi}_j \hat{f}_j(x_0)} = \frac{\hat{\pi}_k \prod_{l=1}^{p} \hat{f}_{kl}(x_0)}{\sum_{j=1}^{J} \hat{\pi}_j \prod_{l=1}^{p} \hat{f}_{jl}(x_0)}$$

Fig. 6.14. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

More on naïve Bayes

- Often works well
  - Biased class densities, but less variance
  - Bias may be small near the decision boundary

- Connection to GAMs

$$
\begin{aligned}
\text{logit}\frac{Pr\{G=k|X\}}{Pr\{G=J|X\}} &= \log\frac{\pi_k \prod_{l=1}^{p} f_{kl}(X)}{\pi_J \prod_{l=1}^{p} f_{Jl}(X)} \\
&= \log\frac{\pi_k}{\pi_J} + \sum_{l=1}^{p} \log\frac{f_{kl}(X)}{f_{Jl}(X)} \\
&= \beta_{0k} + \sum_{l=1}^{p} g_l(X)
\end{aligned}
$$

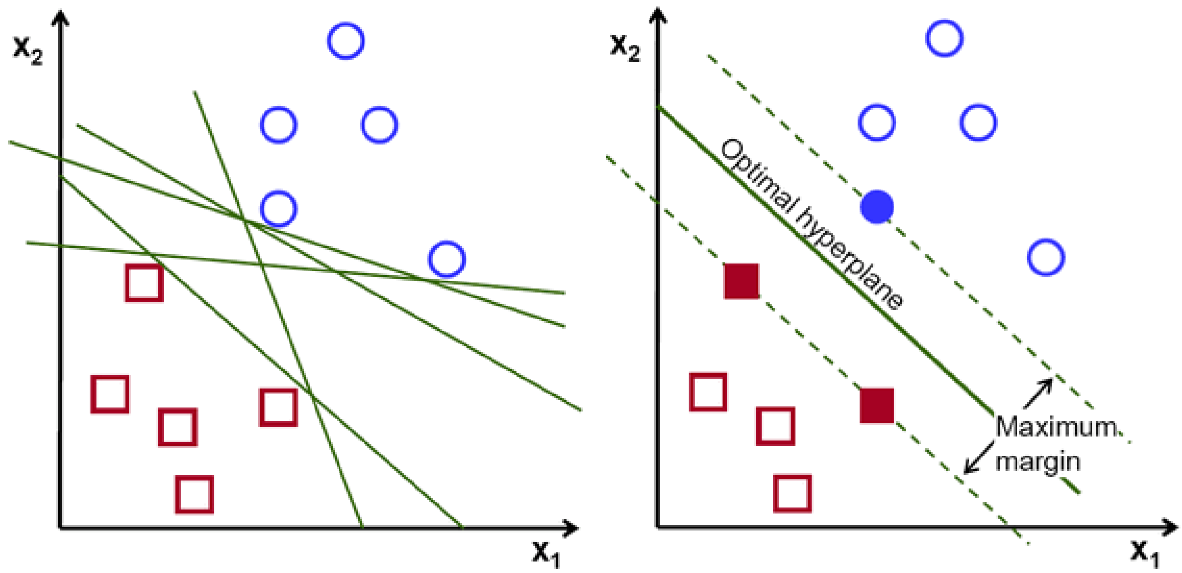
  - Similar to GAM
  - Difference: GAMs do not specify probability distribution on X, while Naïve Bayes does
  - Same distinction as logistic regression vs LDA

Support Vector Machines

HTF 12.1-12.3

KM Ch14.1-14.7

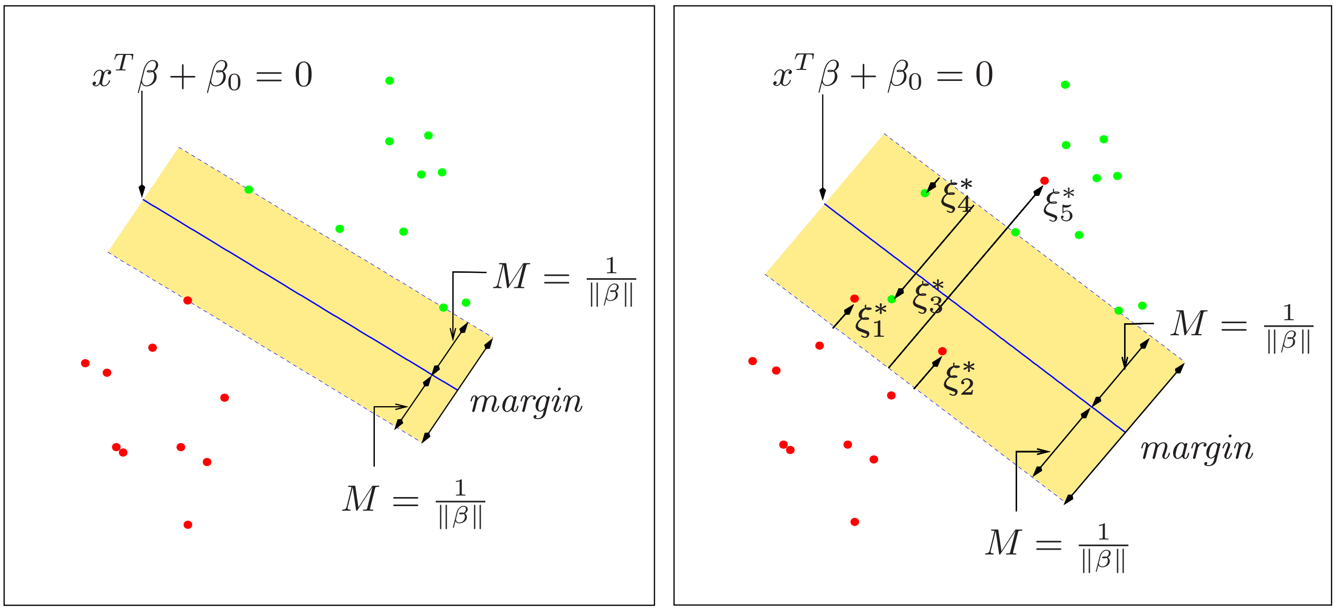
Example: two dimensions



- In a separable case, multiple lines separate classes
- A bad decision boundary passes too close to the points, because it is more likely to result in miscallsifications on new observations
- Goal: find a line passing as far as possible from all points.
- Observations closest to the hyperplane are *support vectors*

`http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html`

Support vector classifiers



- Kernel trick
  - Defines high-dimensional feature vector
  - Prevents underfitting

- Sparsity and large margin principle
  - Ensure that we do not use all dimensions
  - Prevent overfitting

Fig. 12.1. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Separable: HTF 12.2

- Data: $\{(x_i, y_i)\}_{i=1,\ldots,N}$, $x_i \in \mathbb{R}^p$, $y_i \in \{-1, 1\}$

- Define a hyperplane
  - $\{x : f(x) = x'\beta + \beta_0 = 0\}$, $||\beta|| = 1$
  - Distance from a point x to hyperplane: $f(x) = x'\beta + \beta_0$
  - Classification rule $G(x) = \text{sign}(x'\beta + \beta_0)$

- Since the data are separable:
  - Can find β such that $y_i f(x_i) > 0$ for all i
  - Can find β to maximize the margin between training points with class 1 and -1
  - Solve optimization problem
    $$\max_{\beta,\beta_0,||\beta||=1} M \text{ s.t. } y_i(x_i'\beta + \beta_0) \geq M,\ i = 1, \ldots, N$$
  - Can show that, equivalently,
    $$\min_{\beta,\beta_0} ||\beta|| \quad \text{s.t. } y_i(x_i'\beta + \beta_0) \geq 1,\ i = 1, \ldots, N$$
    (drop constraint on $||\beta||$; $M = 1/||\beta||$)

Non-separable: HTF 12.2

- Classes overlap in feature space
- Goal: maximize $||M||$
  - Some points can be on the wrong side of margin
- Solution: slack variables $\xi = (\xi_1, \ldots, \xi_N)$
  - Modify previous optimization problem:

    $\max_{\beta, \beta_0, ||\beta||=1} M$ s.t. $\xi_i \geq 0$, $\sum_{i=1}^{N} \xi_i \leq$ constant

    $y_i(x_i'\beta + \beta_0) \geq M - \xi_i$ or $y_i(x_i'\beta + \beta_0) \geq M(1 - \xi_i)$
  - Lead to different solutions; second is "standard"
  - In second formulation, $\xi_i \propto$ amount by which prediction $f(x_i) = x_i'\beta + \beta_0$ is on the wrong side of *margin*
  - $\xi > 1 \rightarrow$ misclassification
  - Bound $\sum \xi \leq$ constant $\rightarrow$ bound this proportion
  - Can show that, equivalently,

    $\min_{\beta, \beta_0} ||\beta||$ s.t. $y_i(x_i'\beta + \beta_0) \geq 1, i = 1 \ldots, N$

    $\xi_i \geq 0, \sum \xi_i \leq$ constant

Computation

- Convex optimization problem
  - Quad. optimization; linear inequality constraints
  - Solution: quadratic programming

- Equivalent form

$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N} \xi_i \text{ s.t. } \xi_i \geq 0,$$
$$y_i(x_i'\beta + \beta_0) \geq 1 - \xi_i,\ i = 1, \ldots, N$$

  - C: cost parameter replacing the bound constant
  - Separable case $C = \infty$

- Solution: $\widehat{\beta} = \sum_{i=1}^{N} \widehat{\alpha}_i y_i x_i$ (HTF Sec 12.2)
  - $\alpha_i \neq 0$ only for support vectors, i.e. observations where $y_i(x_i'\beta + \beta_0) \geq M(1 - \xi_i)$
  - $\xi_i = 0 \to 0 < \alpha_i < C$; $\quad \xi_i > 0 \to \alpha_i = C$

- Decision

$$\widehat{G}(x) = \text{sign}[\widehat{f}(x)] = \text{sign}[x'\,\widehat{\beta} + \widehat{\beta}_0]$$

Computation details

- Objective function

$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N}\xi_i \text{ s.t. } \xi_i \geq 0,$$
$$y_i(x_i'\beta + \beta_0) \geq 1 - \xi_i,\ \ i = 1, \ldots, N$$

  - The Lagrange (**primal**) function $L_P =$
$$\frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i[y_i(x_i'\beta + \beta_0) - (1 - \xi_i)] - \sum_{i=1}^{N}\mu_i\xi_i$$
  - To maximize wrt β, β_0, ξ_i, set derivatives to 0:
$$\beta = \sum_{i=1}^{N}\alpha_i y_i x_i;\ \ 0 = \sum_{i=1}^{N}\alpha_i y_i;\ \ \alpha_i = C - \mu_i,\ \ \alpha_i, \mu_i, \xi_i \geq 0\ \forall i$$
  - Substitute to primal, to obtain **dual** objective
$$L_D \;\; = \;\; \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N}\alpha_i\alpha_{i'}\ y_i y_{i'}\ x_i' x_{i'} \text{ and constraints}$$

- Unique solution $\widehat{\beta} = \sum_{i=1}^{N}\widehat{\alpha}_i y_i x_i$ (multiple for $\widehat{\beta}_0$)

The kernel trick

- Create non-linear classifiers
  - Transform feature space in higher-dimensions
  - Separate the transformed features by maximum-margin hyperplanes
  - $\uparrow$ # of dimensions $\rightarrow$ $\uparrow$ overfitting

- Back to the optimization problem:
  - **Dual** objective function

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} \, y_i y_{i'} \, x_i' x_{i'}$$

  - Replace dot-products with kernel functions

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} \, y_i y_{i'} \, K(x_i, x_{i'})$$

  - Defines inner product in transformed space $\rightarrow$ do not need to compute the original transformations

Kernel trick examples

- Polynomial kernel $K(x_i, x_{i'}) = (x_i' x_{i'} + 1)^d$
  - d is a tunable parameter
  - Requires one addition and one exponentiation more than the original dot product
- Radial basis function (gaussians)
  $K(x_i, x_{i'}) = \exp\left\{-\frac{||x_i - x_{i'}||^2}{\sigma}\right\}$
  - σ is a parameter
- Sigmoid (neural net activation function)
  $K(x_i, x_{i'}) = \tanh(\kappa_1 x_i' x_{i'} - \kappa_2)$
  - Only works with some constants

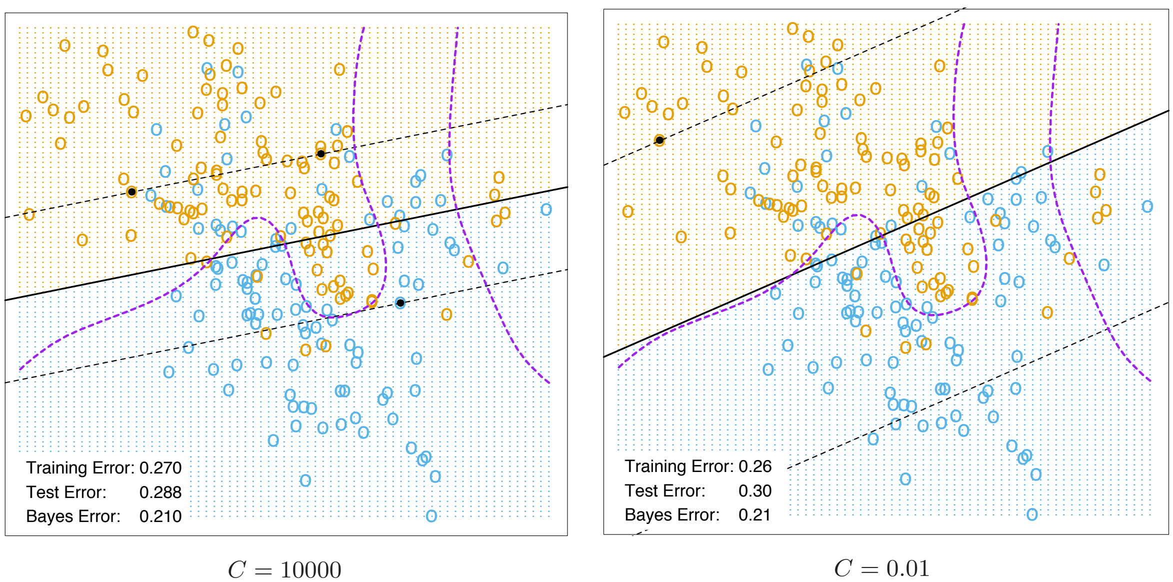
Example: polynomial kernel

- 2-dimensional features $x = (x_1, x_2)$
- Kernel of degree 2: $K(x, x') = (\langle x, x' \rangle + 1)^2$
  - Expanding the inner product

$$\begin{aligned} K(x, x') &= (\langle x, x' \rangle + 1)^2 \\ &= (1 + x_1 x'_1 + x_2 x'_2)^2 \\ &= 1 + 2x_1 x'_1 + 2x_2 x'_2 + x_1^2 \, {x'_1}^2 \\ &\quad + x_2^2 \, {x'_2}^2 + 2 \, x_1 x_2 \, x'_1 x'_2 \end{aligned}$$

  - Equivalent to the basis functions
    $h_1(x) = 1$
    $h_2(x) = \sqrt{2} \, x_1$
    $h_3(x) = \sqrt{2} \, x_2$
    $h_4(x) = x_1^2$
    $h_5(x) = x_2^2$
    $h_6(x) = \sqrt{2} \, x_1 x_2$
  - Then $K(x, x') = (\langle x, x' \rangle + 1)^2 = \langle h(x), h(x') \rangle$
  - With Kernel trick no need to explicitly specify h
  - Less flexibility, easier computation
  - (Notation of HTF Sec. 12.3.1)

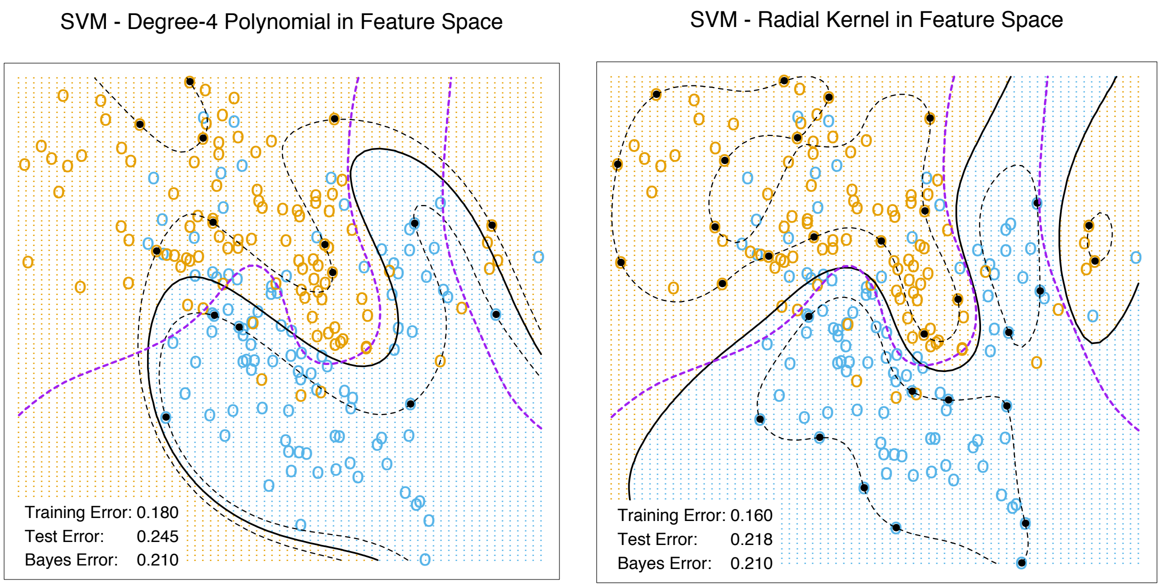
SV classifier (linear) Gaussian mixtures



- Support points are on the wrong side of the margin (68% on the left, 85% on the right)
- Black solid dots are exactly on the margin
- Large $C \to$ small margin
- Small $C \to$ large margin

Fig. 12.2. Hastie, Tibshirani, Friedman *The Elements of Statistical Learning* 2008

SVM
Gaussian mixtures



- Large C discourage $\xi_i > 0$

  $\rightarrow$ fewer support points
  $\rightarrow$ more overfitting
  $\rightarrow$ more wiggly boundary in the original space

- Small C has more support points
  $\rightarrow$ smoother boundary

- Radial kernel performs best, as the data are simulated from mixture of Gaussians

Fig. 12.3. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Performance in higher dimensions

	Method	Test Error (SE) No Noise Features	Six Noise Features
1	SV Classifier	0.450 (0.003)	0.472 (0.003)
2	SVM/poly 2	0.078 (0.003)	0.152 (0.004)
3	SVM/poly 5	0.180 (0.004)	0.370 (0.004)
4	SVM/poly 10	0.230 (0.003)	0.434 (0.002)
5	BRUTO	0.084 (0.003)	0.090 (0.003)
6	MARS	0.156 (0.004)	0.173 (0.005)
	Bayes	0.029	0.029

- Simulation:
  - one class surrounds the other as skin of an orange
  - four informative features
  - 1,000 test observations
  - tune C on validation set
  - smaller error is better
- Linear SV does poorly (true boundary non-linear)
- Larger degree polynomials overfit
- Noise features negatively affect every method

Table. 12.2. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

SVM: regularization

- Solution to convex constraint optimization in SVM is same as solution to

$$\max_{\beta,\beta_0} \sum_{i=1}^{N} \left[1 - y_i \left\{h(x)'\beta + \beta_0\right\}\right]_+ + \lambda \left|\left|\beta\right|\right|^2$$

  - '+': positive part; $\lambda = 1/C$
  - Same as ridge regression, different loss function
  - $h(x)$: transformation corresponding to kernel
  - See HTF Sec 12.3.5 for path algorithm

Loss Function	$L[y, f(x)]$	Minimizing Function
Binomial Deviance	$\log[1 + e^{-yf(x)}]$	$f(x) = \log \frac{\Pr(Y = +1\|x)}{\Pr(Y = -1\|x)}$
SVM Hinge Loss	$[1 - yf(x)]_+$	$f(x) = \text{sign}[\Pr(Y = +1\|x) - \frac{1}{2}]$
Squared Error	$[y - f(x)]^2 = [1 - yf(x)]^2$	$f(x) = 2\Pr(Y = +1\|x) - 1$

Table. 12.2. Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008

Comparison of loss functions

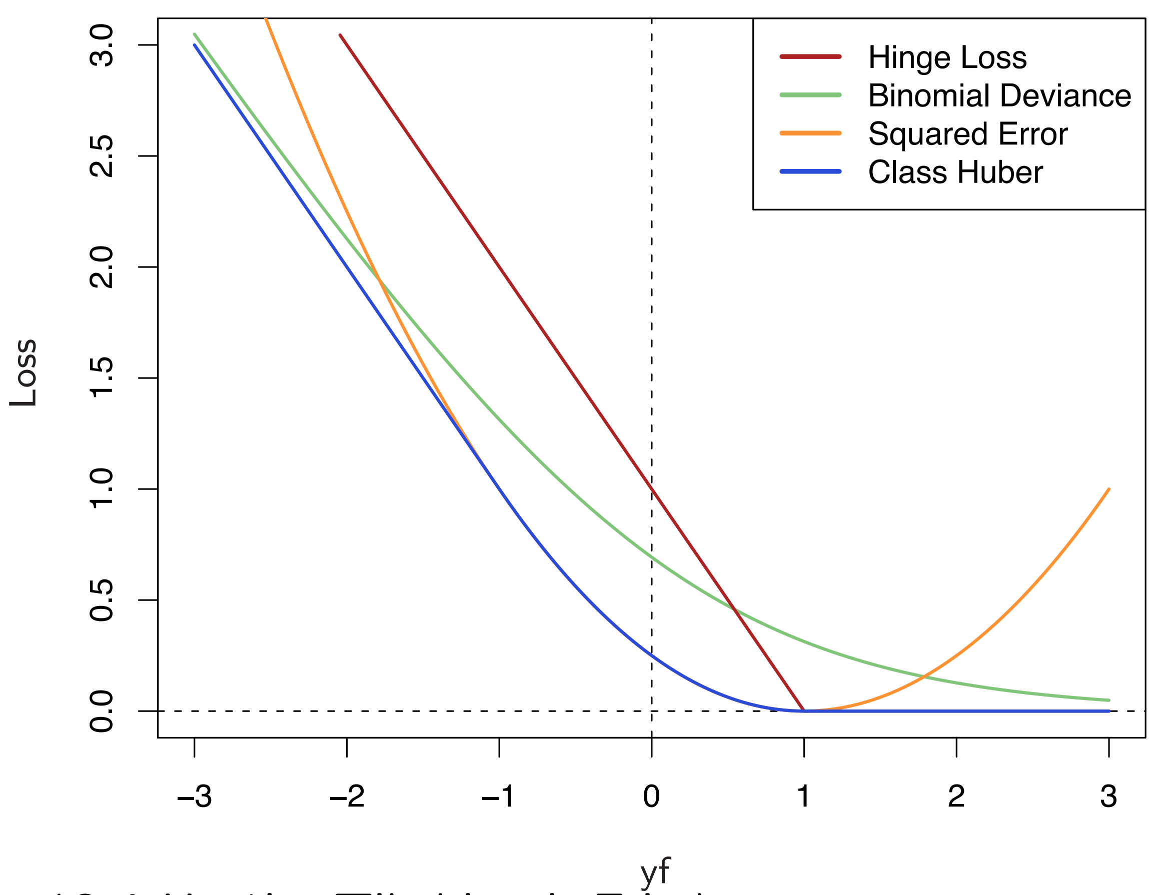


Fig 12.4 Hastie, Tibshirani, Friedman
The Elements of Statistical Learning 2008