
CN700 – Additive Models and Trees

Chapter 9: Hastie et al. (2001)

Madhusudana Shashanka

Department of Cognitive and Neural Systems

Boston University

Overview

- Generalized additive models
- Tree-based models
- PRIM – bump hunting
- MARS

Generalized Additive Models

- Techniques that use predefined basis functions achieve nonlinearity.
- Another approach – generalized additive models.
- More automatic and flexible.
- In the regression setting, it can be expressed as
 - $E(Y|X_1, X_2, \dots, X_p) = \alpha + f_1(X_1) + f_2(X_2) + \dots + f_p(X_p)$.
 - f_j 's are unspecified smooth (nonparametric) functions.
- Fit each function using a scatterplot smoother (cubic smoothing spline or kernel smoother).
- Simultaneously estimate all p functions.

Examples

- In general, the conditional mean $\mu(X)$ of a response Y is related to an additive function of the predictors via a *link* function g : $g[\mu(X)] = \alpha + f_1(X_1) + \dots + f_p(X_p)$.
 - $g(\mu) = \mu$ is the identity link. Linear and additive models for Gaussian response data.
 - $g(\mu) = \text{logit}(\mu)$ or $g(\mu) = \text{probit}(\mu)$ for binomial probabilities. Probit is the inverse Gaussian cumulative distribution function.
 - $g(\mu) = \log(\mu)$ for log-linear or log-additive models for Poisson count data.
- More flexibility
 - Mix linear and other parametric forms
 - Nonlinear components in two or more variables
 - Separate curves in X_j for each level of the factor X_k .

Examples

- $g(\mu) = X^T \beta + \alpha_k + f(Z)$. A *semiparametric* model, where α_k is the effect for the k th level of a qualitative input V .
- $g(\mu) = f(X) + g_k(Z)$, where $g_k(Z) = g(V, Z)$ is an interaction term for the effect of V and Z .
- $g(\mu) = f(X) + g(Z, W)$, where g is nonparametric in two features.
- Example where additive models apply – additive decomposition of time series, $Y_t = S_t + T_t + \epsilon_t$, where S_t is a seasonal component, T_t is a trend and ϵ is an error term.

Fitting additive models

- The model: $Y = \alpha + \sum_{j=1}^p f_j(X_j) + \epsilon$.
- Criterion: penalized sum of squares
 - $PRSS(\alpha, f_1, \dots, f_p) =$
$$\sum_{i=1}^N \left\{ y_i - \alpha - \sum_{j=1}^p f_j(x_{ij}) \right\}^2 + \sum_{j=1}^p \lambda_j \int f_j''(t_j)^2 dt_j.$$
 - $\lambda_j \geq 0$ are tuning parameters.
 - Minimizer is an additive cubic spline model – each f_j is a cubic spline in X_j and knots are at each unique $x_{ij}, i = 1, \dots, N$. However, solution not unique.
- More restrictions
 - Assume $\sum_1^N f_j(x_{ij}) = 0 \quad \forall j$, and
 - the matrix of input values is nonsingular.

Backfitting Algorithm

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

2. Cycle: $j = 1, 2, \dots, p, \dots, 1, 2, \dots, p, \dots$,

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

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

until the functions \hat{f}_j changes less than a specified threshold.

- Algorithm analogous to multiple regression for linear models.

Backfitting Algorithm

- Can accommodate other fitting methods by specifying appropriate smoothing operators S_j :
 - Univariate regression smoothers – local polynomial regression and kernel methods.
 - Linear regression operators – polynomial fits, piecewise constant fits, parametric spline fits, series and Fourier fits.
 - Others – surface smoothers for second (higher) order interactions, and periodic smoothers for seasonal effects.

Eg: Additive Logistic Regression

- $\log \frac{\Pr(Y=1|X)}{\Pr(Y=0|X)} = \alpha + f_1(X_1) + \dots + f_p(X_p).$

- Local Scoring Algorithm

1. Compute starting values: $\hat{\alpha} = \log[\bar{y}/(1 - \bar{y})]$, where $\bar{y} = \text{ave}(y_i)$. Set $\hat{f}_j \equiv 0 \forall j$.

2. Define $\hat{\eta}_i = \hat{\alpha} + \sum_j \hat{f}_j(x_{ij})$ and $\hat{p}_i = 1/[1 + \exp(-\hat{\eta}_i)]$. Iterate

- Construct $z_i = \hat{\eta}_i + \frac{(y_i - \hat{p}_i)}{\hat{p}_i(1 - \hat{p}_i)}$.

- Construct weights $w_i = \hat{p}_i(1 - \hat{p}_i)$.

- Fit an additive model to the targets z_i with weights w_i using a weighted backfitting algorithm. New estimates $\hat{\alpha}, \hat{f}_j, \forall j$.

3. Continue step 2. until change is less than a specified threshold.

Summary: Additive Models

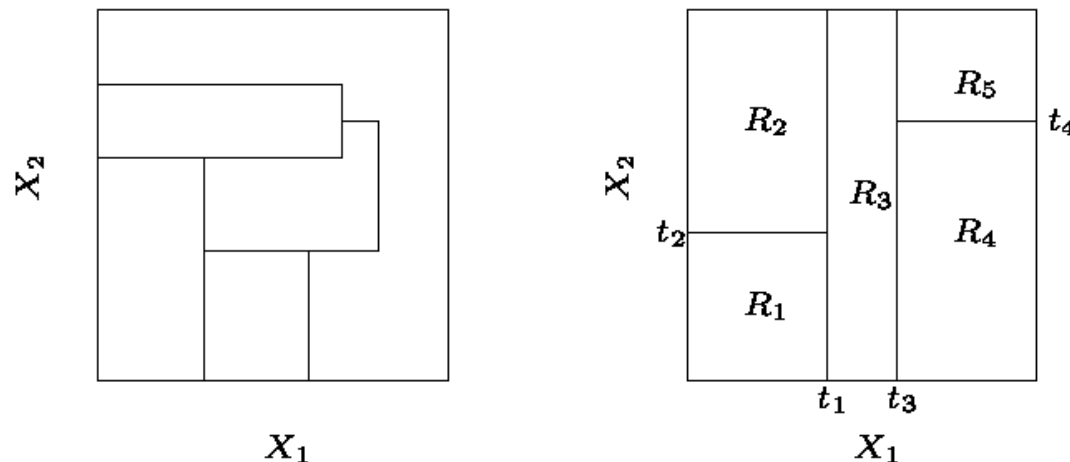
- Flexible, yet interpretable.
- Familiar tools for modelling and inference in linear models also available here.
- Backfitting simple and modular, can choose a fitting method appropriate for each input variable.
- Limitations for large data-mining applications.
- Backfitting fits all predictors – not feasible or desirable with large data.

Overview

- Generalized additive models
- Tree-based models
- PRIM – bump hunting
- MARS

Introduction

- Partition the feature space into a set of rectangles.
- Fit a simple model (like a constant) in each one.
- Key advantage – interpretability.



- Prediction: $\hat{f}(X) = \sum_{m=1}^5 c_m I\{(X_1, X_2) \in R_m\}$.

Regression Trees

- Data: (x_i, y_i) for $i = 1, \dots, N$, with $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$.
- Aim: algorithm to automatically decide splitting variables and split points; and the tree topology.
- Model: M regions R_1, R_2, \dots, R_M and a constant response c_m in each region – $f(x) = \sum_{m=1}^M c_m I(x \in R_m)$.
- Criterion: minimization of sum of squares $\sum (y_i - f(x_i))^2$.
- Best \hat{c}_m : average of y_i in R_m , i.e. $\hat{c}_m = \text{ave}(y_i | x_i \in R_m)$.
- Best binary partition: Computationally infeasible.
- How to proceed? Greedy algorithm.

Best Split

- Consider a splitting variable j and a split point s .
- Define the pair of half-planes $R_1(j, s) = \{X | X_j \leq s\}$ and $R_2(j, s) = \{X | X_j > s\}$.
- Find j and s that solve

$$\min_{j,s} \left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right].$$

- Inner minimization is solved by $\hat{c}_1 = \text{ave}(y_i | x_i \in R_1(j, s))$ and $\hat{c}_2 = \text{ave}(y_i | x_i \in R_2(j, s))$.
- Find the best pair (j, s) by scanning through all split points for each splitting variable and then scanning through all variables.

Tree Size

- Adaptively chosen from the data.
 - Grow a large tree T_0 till some minimum node size is reached.
 - Prune this tree using *cost-complexity pruning*.
- Cost complexity criterion: $C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T|$,
where $Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2$ and $\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i$.
- Idea: For each α , find the subtree $T_\alpha \subseteq T_0$ to minimize $C_\alpha(T)$.
- Tuning parameter $\alpha \geq 0$ governs tradeoff between tree-size and goodness of fit.

Tree Size – Tuning parameter

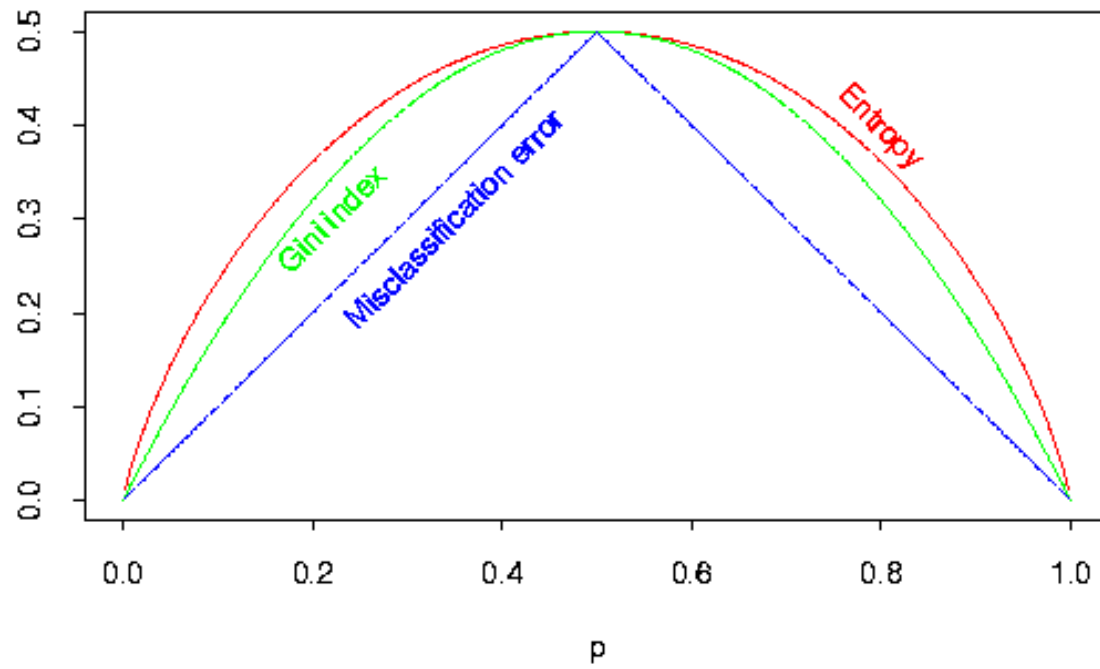
- For each α , there is a unique smallest subtree T_α that minimizes $C_\alpha(T)$.
- Use *weakest link pruning* to find T_α .
 - Successively collapse the internal node that produces the smallest per-node increase in $\sum_m N_m Q_m(T)$, and continue until the single-node tree.
 - This sequence must contain T_α .
- Estimation of α is by cross-validation.
 - Choose $\hat{\alpha}$ to minimize the cross-validated sum of squares.
- Final tree is $T_{\hat{\alpha}}$.

Classification Trees – K classes

- The proportion of class k observations in node m is given by
$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k).$$
- Observations in node m classified to class
$$k(m) = \arg \max_k \hat{p}_{mk}.$$
- Different measures $Q_m(T)$ of node impurity
 - Misclassification error:
$$\frac{1}{N_m} \sum_{i \in R_m} I(y_i \neq k(m)) = 1 - \hat{p}_{mk(m)}.$$
 - Gini index: $\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^K \hat{p}_{mk} (1 - \hat{p}_{mk}).$
 - Cross-entropy (deviance): $\sum_{k=1}^K \hat{p}_{mk} \log \hat{p}_{mk}.$
- Cross-entropy and Gini index are differentiable. Hence, more amenable to numerical optimization.

Classification Trees

- When growing the tree, either gini index or cross-entropy should be used.
- To guide cost-complexity pruning, typically misclassification rate is used.



Other issues and modifications

- Categorical Predictors: Given a predictor with q possible unordered values and a binary outcome -
 - Order predictor classes according to the proportion falling in outcome class 1.
 - Split predictor as if it were ordered.
- Loss Matrix:
 - In the multi-class case, modify Gini index to $\sum_{k \neq k'} L_{kk'} \hat{p}_{mk} \hat{p}_{mk'}$.
 - For two classes, weight observations in class k by $L_{kk'}$.
- Missing Predictor Values:
 - Categorical predictors - make a new “missing” category.
 - General approach - make surrogate variables.

Disadvantages

- Instability and high variance: hierarchical nature.
- Lack of smoothness: can degrade performance in regression setting.
- Difficulty with additive structures
 - Consider $Y = c_1 I(X_1 < t_1) + c_2 I(X_2 < t_2) + \epsilon$.
 - First split on X_1 near t_1 . The next split at both nodes should be on X_2 at t_2 .

Overview

- Generalized additive models
- Tree-based models
- PRIM – bump hunting
- MARS

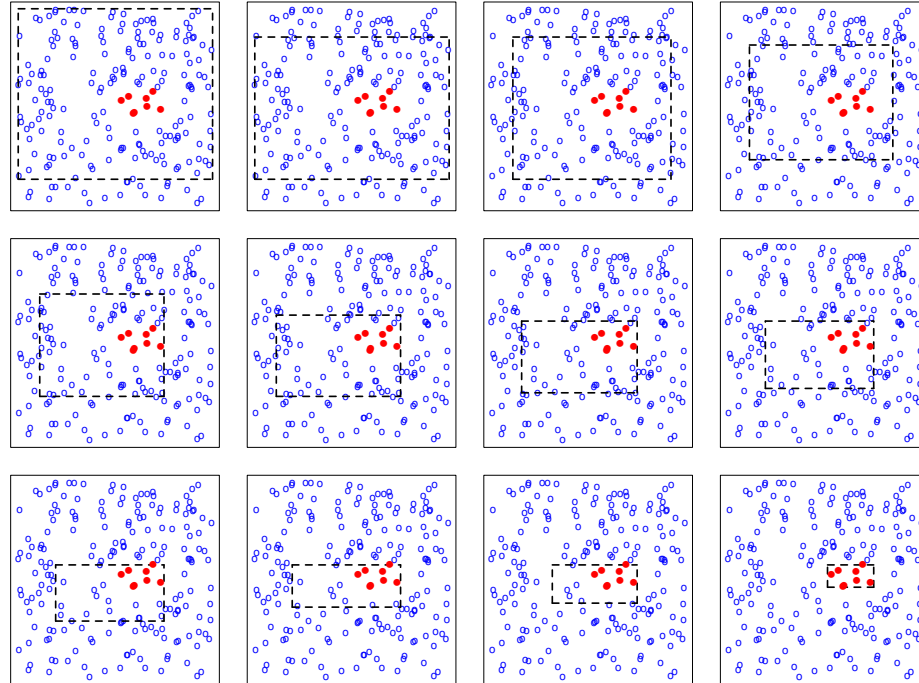
Introduction

- PRIM – patient rule induction method.
- Boxes in feature space where response average is high.
- Looks for maxima in target function – *bump hunting*.
- Box definitions not defined by a binary tree.
- Characterized by *peeling* and *pasting*.

Algorithm

1. Start with a maximal box containing all training data.
 2. Shrink box by compressing one face, so as to peel off proportion α of observations such that the peeling produces the highest response mean in the remaining box.
 3. Repeat step 2 until some minimal number of observations remain in the box.
 4. Expand along any face, as long as the resulting box mean increases.
 5. Steps 1-4 give a sequence of boxes, with different numbers of observations in each box. Use cross-validation to choose a member of the sequence and call the box B_1 .
 6. Remove the data in box B_1 from the dataset and repeat steps 2-5 to obtain a second box, and continue to get as many boxes as desired.
-

Algorithm – Illustration



- Two classes – blue (class 0) and red (class 1).

PRIM and CART

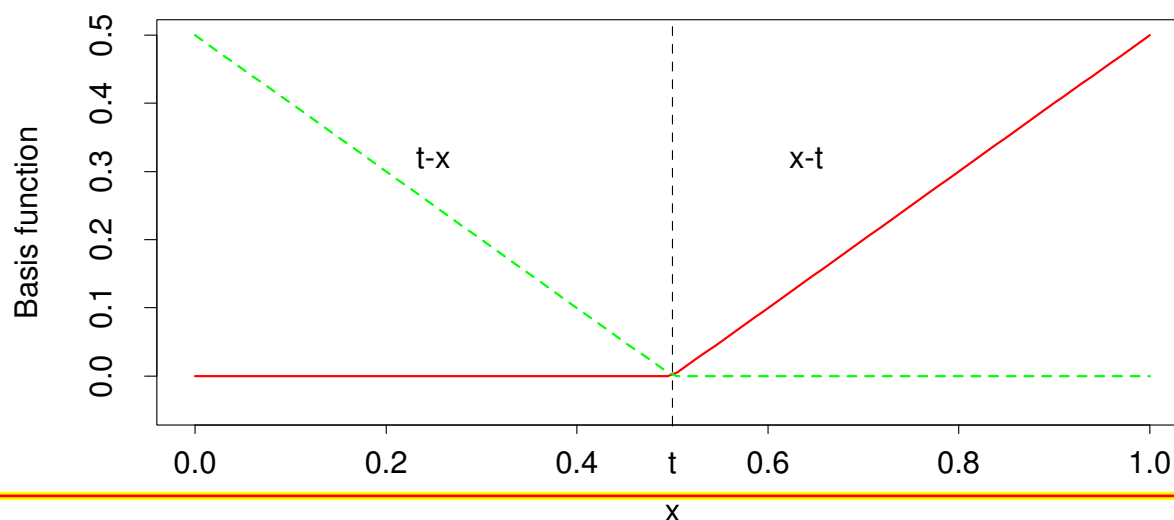
- PRIM handles categorical variables and missing values like CART.
- PRIM: No simple way to deal with $k > 2$ classes simultaneously. Run PRIM separately for each class versus a baseline class.
- Advantage of PRIM over CART: patience.
 - CART fragments data quite quickly. $\log_2(N) - 1$ steps before running out of data.
 - PRIM: approx $\log(N) / \log(1 - \alpha)$ peeling steps before running out of data.

Overview

- Generalized additive models
- Tree-based models
- PRIM – bump hunting
- MARS

Introduction

- MARS – Multivariate Adaptive Regression Splines.
 - Generalization of stepwise linear regression.
 - Modification of CART for the regression setting.
- Well-suited for high-dimensional problems.
- Uses expansions in piecewise linear basis functions of the form $(x - t)_+$ and $(t - x)_+$. The functions form a *reflected pair* with a *knot* at value t .



MARS – description

- Idea: Form reflected pairs for each input X_j with knots at each observed value x_{ij} of that input.
- Collection of basis functions is

$$\mathcal{C} = \left\{ (X_j - t)_+, (t - X_j)_+ \right\}_{\substack{t \in \{x_{1j}, x_{2j}, \dots, x_{Nj}\} \\ j=1, 2, \dots, p}}$$

- Model has the form

$$f(X) = \beta_0 + \sum_{m=1}^M \beta_m h_m(X),$$

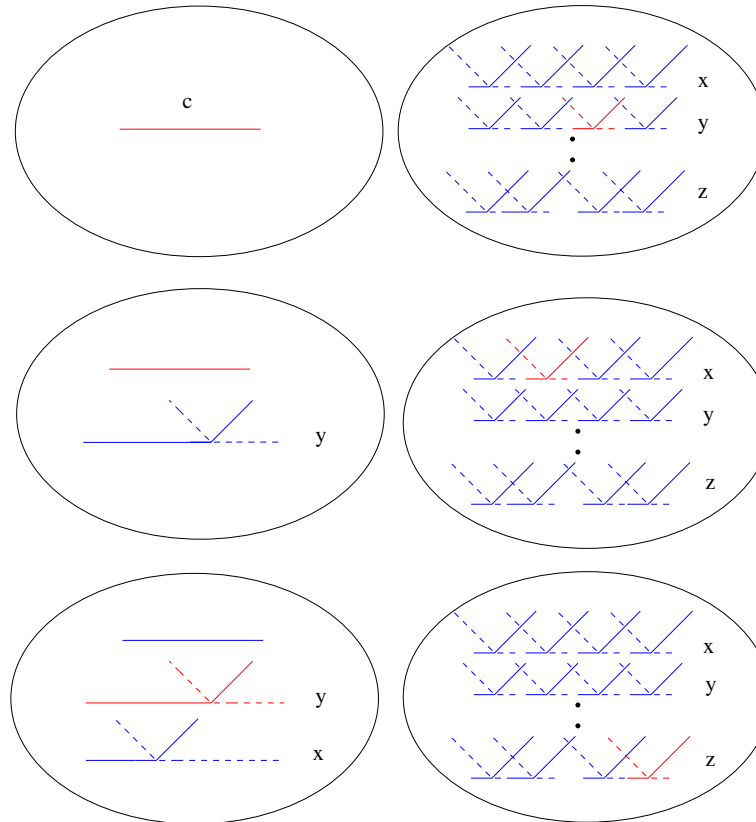
where each $h_m(X)$ is a function in \mathcal{C} , or a product of two or more such functions.

- Given h_m, β_m estimated by standard linear regression.

Basis Functions

- Start with the constant function $h_0(X) = 1$ in the model set \mathcal{M} .
- At each stage, consider as a new basis function pair – all products of a function h_m in \mathcal{M} with one of the reflected pairs in \mathcal{C} .
- Add to \mathcal{M} the term of the form $\hat{\beta}_{M+1} h_\ell(X) \cdot (X_j - t)_+ + \hat{\beta}_{M+2} h_\ell(X) \cdot (t - X_j)_+$, $h_\ell \in \mathcal{M}$ that produces the largest decrease in training error.
- Estimate coefficients by least-squares.
- Continue until \mathcal{M} contains some preset maximum number of terms.
- Restriction: each input at most once in a product.

MARS illustration



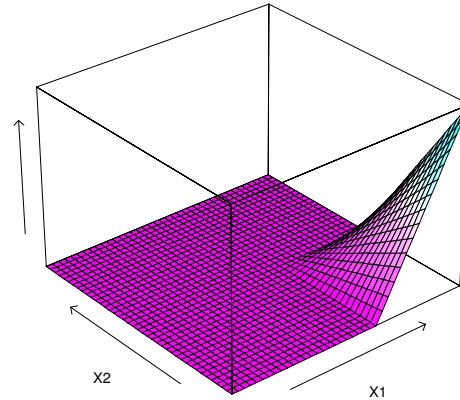
- \mathcal{M} on the left column and \mathcal{C} on the right. Selected functions shown in red.

Backward deletion

- Model \mathcal{M} is large and typically overfits data.
- At each stage
 - Term whose removal causes the smallest increase in residual squared error is deleted.
 - Estimated best model \hat{f}_λ , where $\lambda =$ number of terms.
- Use generalized cross-validation (GCV) for optimal λ .
- GCV criterion:
$$GCV(\lambda) = \frac{\sum_{i=1}^N (y_i - \hat{f}_\lambda(x_i))^2}{(1 - M(\lambda)/N)^2}.$$
- $M(\lambda)$ is the effective number of parameters:
 - $M(\lambda) = r + cK$, r L.I. basis functions in \mathcal{M} , K knots and $c = 3$.
 - $c = 2$ when model is restricted to be additive.

Advantages

- Piecewise linear functions operate locally.



- Computations: consider pdt of a function in \mathcal{M} with each of the N reflected pairs for an input X_j .
 - $O(N)$ operations to try every knot!
- Multiway products built up from products involving terms already in model – reasonable working assumption.

MARS for classification

- Two classes: 0/1 response, treat as regression.
- Multiclass: 0/1 indicator variables, use multiresponse MARS regression.
- Masking problems with the above approach.
- PolyMARS specifically designed for classification:
 - Multiple logistic framework.
 - Use quadratic approximation to the multinomial log-likelihood to search for the next basis-function pair.
 - Fit enlarged model by maximum likelihood.

References

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