Additive Models，Trees，and Related Models

Prof. Liqing Zhang

Dept. Computer Science & Engineering,

Shanghai Jiao Tong University

Introduction

9.1 Generalized Additive Models

- In the regression setting, a generalized additive models has the form:

$$E(Y \mid X_1, X_2, \ldots, X_p) = \alpha + f_1(X_1) + f_2(X_2) + \ldots + f_p(X_p)$$

Here f_i s are unspecified smooth and nonparametric functions.

Instead of using LBE(Linear Basis Expansion) in chapter 5, we fit each function using a scatter plot smoother(e.g. a cubic smoothing spline)

GAM(cont.)

- For two-class classification, the additive logistic regression model is:

$$\log\left(\frac{\mu(X)}{1-\mu(X)}\right)=\alpha+f_1(X_1)+f_2(X_2)+\ldots+f_p(X_p) \qquad (9.2)$$

Here

$$\mu(X)=\Pr(Y=1|X)$$

GAM(cont)

- In general, the conditional mean U(x) of a response y is related to an additive function of the predictors via a link function g:

$$g[\mu(X)] = \alpha + f_1(X_1) + f_2(X_2) + \ldots + f_p(X_p)$$

Examples of classical link functions:

- Identity: $g(\mu) = \mu$, for Gaussian response data.
- Logit: $g(\mu) = \log[\mu/(1-\mu)]$
- Probit: $g(\mu) = \Phi^{-1}(\mu)$ or modeling binomial probability
- Log: $g(\mu) = \log(\mu)$ for Poisson count data

Fitting Additive Models

- The additive model has the form:

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

Here we have $\mathbf{E}(\varepsilon)=0$

Given observations x_i y_i, a criterion like penalized sum squares can be specified for this problem:

$$PRSS(\alpha, f_1, f_2, \ldots, f_p) = \sum_{i=1}^{N}\left(y_i - \alpha - \sum_{j=1}^{p} f_j(x_{ij}) \right)^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t_j)^2 dt_j$$

where $\lambda_j \geq 0$ are tuning parameters.

FAM(cont.)

Conclusions:

- The solution to minimize PRSS is cubic splines, however without further restrictions the solution is not unique.
- If $\sum_1^N f_j(x_{ij}) = 0$ holds, it is easy to see that :

$$\hat{\alpha} = mean(y)$$

- If in addition to this restriction, the matrix of input values has full column rank, then it is a strict convex criterion and has an unique solution. If the matrix is singular, then the linear part of f_j cannot be uniquely determined. (Buja 1989)

Learning GAM: Backfitting

Backfitting algorithm

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

2. Cycle: $j = 1,2,\ldots,p,\ldots,1,2,\ldots,p,\ldots,$ (*m* cycles)

$$\text{Fit } \hat{f}_j \text{ with data } \{x_{ij}\}_1^N, \quad \{y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik})\}_1^N$$

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

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

Candidate fitting Model : $\hat{f} = N(N^T N + \lambda \Omega_N)^{-1} N^T y = S_\lambda y$

Backfitting: Points to Ponder

$$Y = \alpha + \sum_{j=1}^{p} f_j(X_j) + \varepsilon, \quad \varepsilon \sim N(0, \sigma^2)$$

$$E(Y - \alpha - \sum_{k \neq j} f_k(X_k) \mid X_j) = f_j(X_j)$$

Computational Advantage?

Convergence?

How to choose fitting functions?

FAM(cont.)

Algorithm 9.1 *The Backfitting Algorithm for Additive Models.*

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

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

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

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

$$\hat{f} = N(N^T N + \lambda\Omega_N)^{-1} N^T y = S_\lambda y$$

Other Alternatives

- Other fitting methods in exactly the same way
  - Other univariate regression smoothers such as local polynomial regression and kernel methods;
  - Linear regression operators yielding polynomial fits, piecewise constant fits, parametric spline fits, series and Fourier fits;
  - More complicated operators such as surface smoothers for second or higher-order interactions or periodic smoothers for seasonal effects.

Additive Logistic Regression

The generalized additive logistic model has the form

$$\log \frac{\Pr(Y=1 \mid X_1,\ldots,X_p)}{\Pr(Y=0 \mid X_1,\ldots,X_p)} = \alpha + f_1(X_1) + \cdots + f_p(X_p)$$

The functions $f_1, f_2, \ldots, f_p$ are estimated by a backfitting within a Newton-Raphson procedure.

Logistic Regression

- Model the class posterior $\Pr(G=k \mid X=x)$ in terms of *K*-1 log-odds

$$\log\frac{\Pr(G=k \mid X=x)}{\Pr(G=K \mid X=x)} = \beta_{k0} + \beta_k^T x, k = 1,\ldots,K-1$$

- Decision boundary is set of points

$$\{x : \Pr(G=k \mid X=x) = \Pr(G=l \mid X=x)\} = \{x : \log\frac{\Pr(G=k \mid X=x)}{\Pr(G=l \mid X=x)} = 0\}$$

$$= \{x : (\hat{\beta}_{k0} - \hat{\beta}_{l0}) + (\hat{\beta}_k - \hat{\beta}_l)^T x = 0\}$$

- Linear discriminant function for class *k*

$$\delta_k(x) = \beta_{k0} + \beta_k^T x$$

- Classify to the class with the largest value for its $\delta_k(x)$

$$\hat{G}(x) = \arg\max_{k\in g} \delta_k(x)$$

Logistic Regression con't

- Parameters estimation
  - Objective function

$$\hat{\beta} = \arg\max_{\beta} \sum_{i=1}^{N} \log \Pr_{\beta}(y_i \mid x_i)$$

  - Parameters estimation

    IRLS (iteratively reweighted least squares)

    Particularly, for two-class case, using Newton-Raphson algorithm to solve the equation, the objective function:

$$p(x,\beta) = \Pr_{\beta}(y=1 \mid x); \Pr_{\beta}(y=0 \mid x) = 1 - p(x,\beta)$$

$$p(x,\beta) = \Pr_{\beta}(y=1 \mid x) = \frac{\exp(\beta^T x)}{1+\exp(\beta^T x)};$$

$$l(\beta) = \sum_{i=1}^{N} y_i \log p(x_i,\beta) \;+ (1-y_i)\log(1-p(x_i,\beta))$$

Logistic Regression con’t

Probability

$$p(x,\beta) = \Pr_\beta(y=1\,|\,x) = \frac{\exp(\beta^T x)}{1+\exp(\beta^T x)};$$

Log-Likelihood Function

$$l(\beta) = \sum_{i=1}^{N} y_i \log p(x_i,\beta) \; + (1-y_i)\log(1-p(x_i,\beta))$$

$$= \sum_{i=1}^{N} \{y_i \beta^T x_i - \log(1+e^{\beta^T x_i})\}$$

First order derivative

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{N} x_i(y_i - p(x_i;\beta)) = 0$$

$$\frac{\partial^2 l(\beta)}{\partial\beta\partial\beta^T} = -\sum_{i=1}^{N} x_i x_i^{\ T} p(x_i;\beta)(1-p(x_i;\beta))$$

Logistic Regression con't

Derivative

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{N} x_i (y_i - p(x_i;\beta)) = X^T (y - p)$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} x_i x_i^T p(x_i;\beta)(1 - p(x_i;\beta)) = -X^T W X$$

Newton Method:

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} \right)^{-1} \frac{\partial l(\beta)}{\partial \beta}$$

$$\beta^{new} = \beta^{old} + (X^T W X)^{-1} X^T (y - p) = \boxed{(X^T W X)^{-1} X^T W z}$$

$$z = X\beta^{old} + W^{-1}(y - p), \qquad p_i = p(x_i;\beta^{old}), \qquad W_i = p_i(1 - p_i),$$

Logistic Regression con’t

$$\beta^{new} = \beta^{old} + (X^T W X)^{-1} X^T (y - p)$$
$$= (X^T W X)^{-1} X^T W z$$

Solving a linear regression Problem：

$$\beta^{new} \leftarrow \arg\min_{\beta} (z - X\beta)^T W (z - X\beta)$$

What is the advantage of using such formulation?

Possible to use all tricks in Linear Regression:

Regularization and Error Analysis

Logistic Regression con’t

- Feature selection
  - Find a subset of the variables that are sufficient for explaining their joint effect on the response.
  - Two ways to select features:
    - Backward selection, drop features one by one
    - Forward selection, add features one by one
- Regularization
  - Maximum penalized likelihood
  - Shrinking the parameters via an L_1 constraint, imposing a margin constraint in the separable case

$$\sum_{i=1}^{N} \log \Pr_{\beta}(y_i \mid x_i) - \frac{C}{2}\|\beta\|^2$$

Additive Logistic Regression: Backfitting

Fitting logistic regression

1. $\hat{\beta}_j \equiv 0 \forall j$

2. $\hat{\eta}_i = \sum_j \hat{\beta}_j x_{ij}, \hat{p}_i = \frac{1}{1+e^{-\hat{\eta}_i}}$

Iterate:

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

b. $z_i = \hat{\eta}_i + w_i^{-1}(y_i - \hat{p}_i)$

c. Using weighted least squares to fit a linear model to z_i with weights w_i, give new estimates $\hat{\beta}_j \forall j$

3. Continue step 2 until $\hat{\beta}_j$ converge

Fitting additive logistic regression

1. $\hat{\alpha} = \log \frac{\bar{y}}{1-\bar{y}}$ where $\bar{y} = avg(y_i), \hat{f}_j \equiv 0 \forall j$

2. $\hat{\eta}_i = \hat{\alpha} + \sum_j \hat{f}_j(x_{ij}), \hat{p}_i = \frac{1}{1+e^{-\hat{\eta}_i}}$

Iterate:

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

b. $z_i = \hat{\eta}_i + w_i^{-1}(y_i - \hat{p}_i)$

c. Using weighted backfitting algorithm to fit an additive model to z_i with weights w_i, give new estimates $\hat{f}_j \forall j$

3.Continue step 2 until $\hat{f}_j$ converge

Additive Logistic Regression

Algorithm 9.2 *Local Scoring Algorithm for the Additive Logistic Regression Model.*

1. Compute starting values: $\hat{\alpha} = \log\left[\bar{y}/(1-\bar{y})\right]$, where $\bar{y} = \text{ave}(y_i)$, the sample proportion of ones, and set $\hat{f}_j \equiv 0 \; \forall j$.
2. Define $\hat{\eta}_i = \hat{\alpha} + \sum_j \hat{f}_j(x_{ij})$ and $\hat{p}_i = 1/\left[1+\exp(-\hat{\eta}_i)\right]$

   Iterate:

a) Construct the working target variable

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

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

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

3. Continue step 2. until the change in the functions falls below a prespecified threshold.

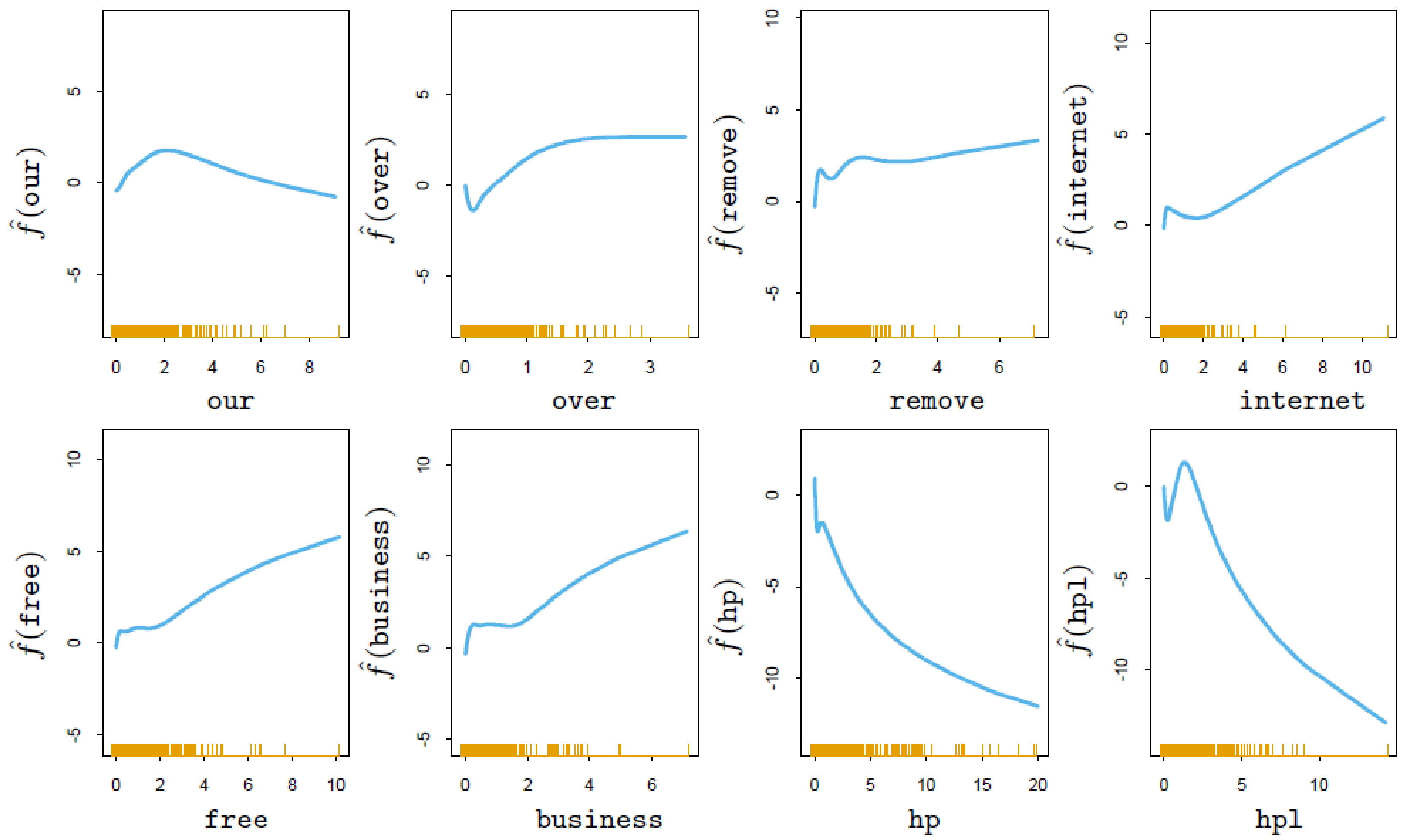
SPAM Detection via Additive Logistic Regression

- Input variables (predictors):
  - 48 quantitative predictors—the percentage of words in the email that match a given word. Examples include **business**, **address**, **internet**, **free**, and **george**.
  - 6 quantitative predictors—the percentage of characters in the email that match a given character. The characters are **ch;**, **ch(**, **ch[**, **ch!**, **ch$**, and **ch#**.
  - The average length of uninterrupted sequences of capital letters: **CAPAVE**.
  - The length of the longest uninterrupted sequence of capital letters: **CAPMAX**.
  - The sum of the length of uninterrupted sequences of capital letters: **CAPTOT**.
- Output variable: SPAM (1) or Normal Email (0)
- f_j' s are taken as cubic smoothing splines

TABLE 9.2. *Significant predictors from the additive model fit to the spam training data. The coefficients represent the linear part of $\hat{f}_j$, along with their standard errors and Z-score. The nonlinear P-value is for a test of nonlinearity of $\hat{f}_j$.*

Name	Num.	df	Coefficient	Std. Error	Z Score	Nonlinear P-value
			Positive effects			
`our`	5	3.9	0.566	0.114	4.970	0.052
`over`	6	3.9	0.244	0.195	1.249	0.004
`remove`	7	4.0	0.949	0.183	5.201	0.093
`internet`	8	4.0	0.524	0.176	2.974	0.028
`free`	16	3.9	0.507	0.127	4.010	0.065
`business`	17	3.8	0.779	0.186	4.179	0.194
`hpl`	26	3.8	0.045	0.250	0.181	0.002
`ch!`	52	4.0	0.674	0.128	5.283	0.164
`ch$`	53	3.9	1.419	0.280	5.062	0.354
`CAPMAX`	56	3.8	0.247	0.228	1.080	0.000
`CAPTOT`	57	4.0	0.755	0.165	4.566	0.063
			Negative effects			
`hp`	25	3.9	−1.404	0.224	−6.262	0.140
`george`	27	3.7	−5.003	0.744	−6.722	0.045
`1999`	37	3.8	−0.672	0.191	−3.512	0.011
`re`	45	3.9	−0.620	0.133	−4.649	0.597
`edu`	46	4.0	−1.183	0.209	−5.647	0.000

*


$\hat{f}(\text{our})$
our
$\hat{f}(\text{over})$
over
$\hat{f}(\text{remove})$
remove
$\hat{f}(\text{internet})$
internet
$\hat{f}(\text{free})$
free
$\hat{f}(\text{business})$
business
$\hat{f}(\text{hp})$
hp
$\hat{f}(\text{hpl})$
hpl

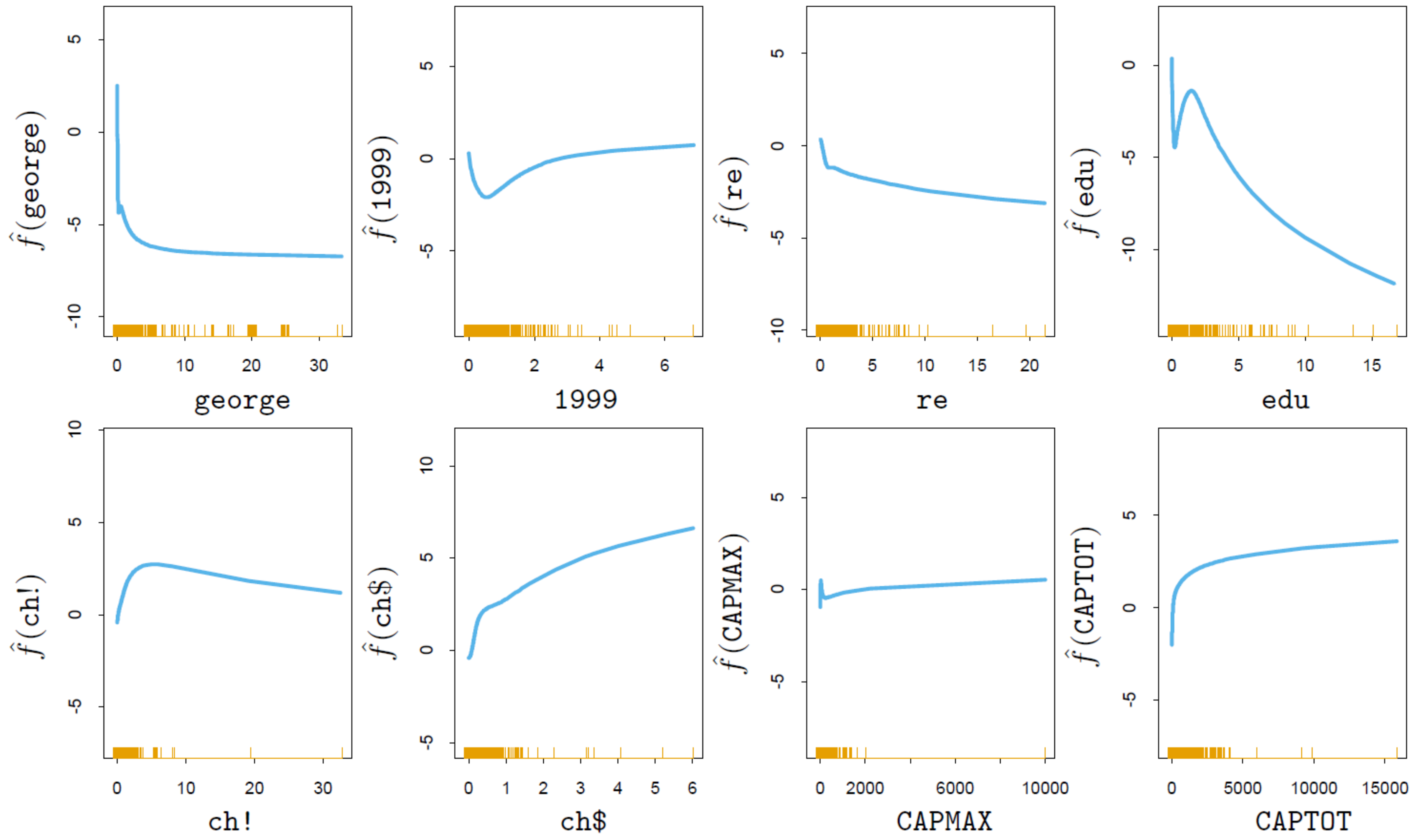


FIGURE 9.1. *Spam analysis: estimated functions for significant predictors. The* rug plot *along the bottom of each frame indicates the observed values of the corresponding predictor. For many of the predictors the nonlinearity picks up the discontinuity at zero.*

SPAM Detection: Results

True Class	Predicted Class	
	Email (0)	SPAM (1)
Email (0)	58.5%	2.5%
SPAM (1)	2.7%	36.2%

Sensitivity: Probability of predicting spam given true state is spam = $\frac{36.2}{36.2+2.7}=0.93$

Specificity: Probability of predicting email given true state is email = $\frac{58.5}{58.5+2.5}=0.96$

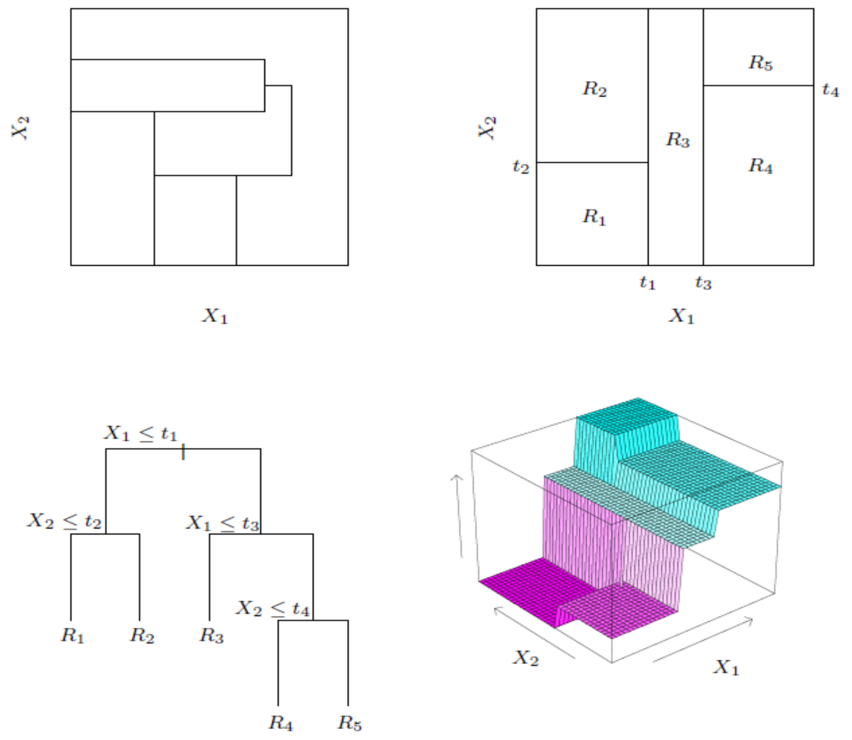
GAM: Summary

- Useful flexible extensions of linear models
- Backfitting algorithm is simple and modular
- Interpretability of the predictors (input variables) are not obscured
- But not suitable for very large data mining applications (why?)

Introduction

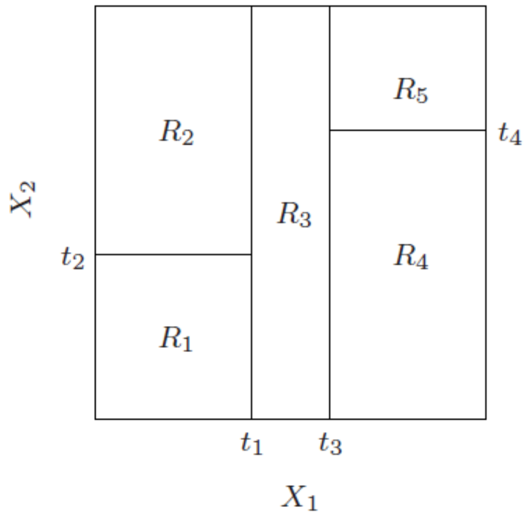
9.2 Tree-Based Method

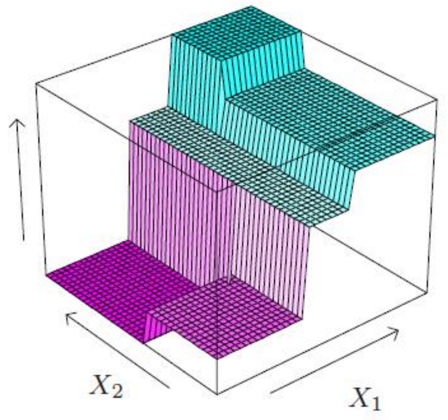
- Background: Tree-based models partition the feature space into a set of rectangles, and then fit a simple model(like a constant) in each one.
- A popular method: CART(classification and regression tree)



CART

- Example: Let’s consider a regression problem:
  - inputs X1 and X2 ;   Output: continuous response Y





$$\hat{f}(X) = \sum_{m=1}^{5} c_m I\left\{(X_1, X_2) \in R_m\right\}$$

For illustration, we choose *C1=-5, C2=-7, C3=0,C4=2, C5=4*

Regression Tree

- Suppose we have a partition into M regions: $R_1, R_2, \ldots, R_M$. We model the response Y with a constant C_m in each region:

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m).$$

If we adopt as our criterion minimization of RSS, it is easy to see that:

$$\hat{c}_m = ave(y_i \mid x_i \in R_m)$$

Regression Tree(cont.)

- Finding the best binary partition in term of minimum RSS is computationally infeasible
- A greedy algorithm is used.

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

Here

$$R_1(j,s)=\{X\mid X_j\le s\}\quad and\quad R_2(j,s)=\{X\mid X_j>s\}. \qquad (9.12)$$

Regression Tree(cont.)

- For any choice j and s, the inner minimization is solved by:

$$\hat{c}_1 = ave(y_i \mid x_i \in R_1(j,s)) \quad \hat{c}_2 = ave(y_i \mid x_i \in R_2(j,s))$$

- For each splitting variable X_j , the determination of split point s can be done by scanning through all of the input, determination of the best pair (j,s) is feasible.
- Partition the data into two regions and repeat the splitting progress in each of the two regions.

Regression Tree(cont.)

- We index terminal nodes by *m*, with node m representing region Rm.
- Let |T| denotes the number of terminal notes in T.

$$N_m = \#\{x_i \in R_m\},$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i,$$

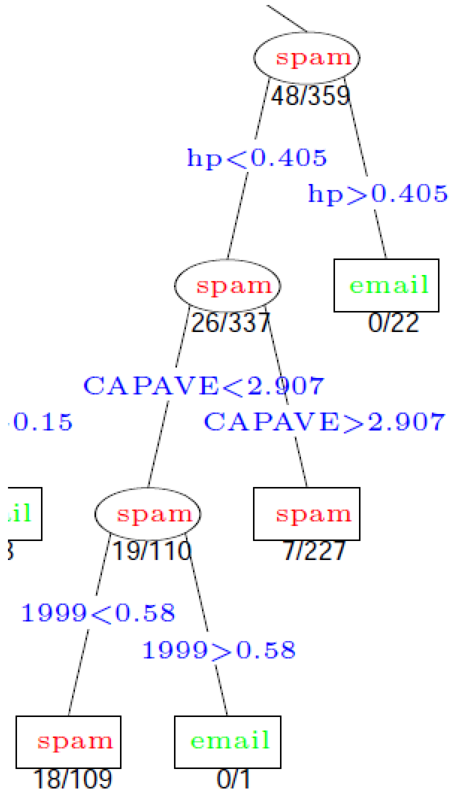
$$Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2,$$

- We define the cost complexity criterion:

$$c_\alpha(T) = \sum_{m=1}^{|x|} N_m \hat{Q}_m(T) + \alpha |T|,$$

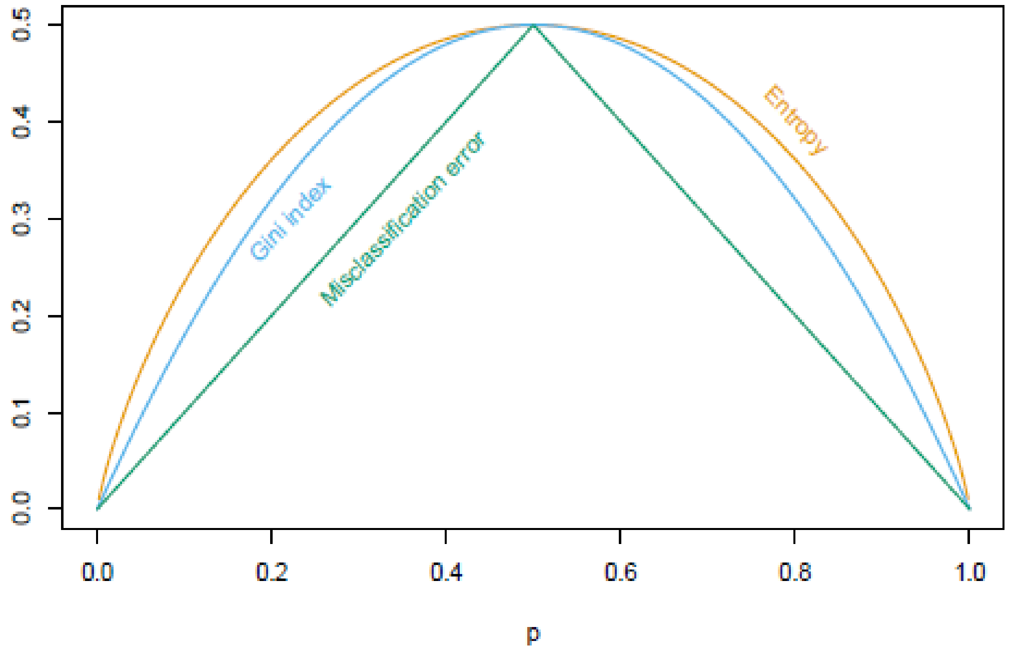
Classification Tree

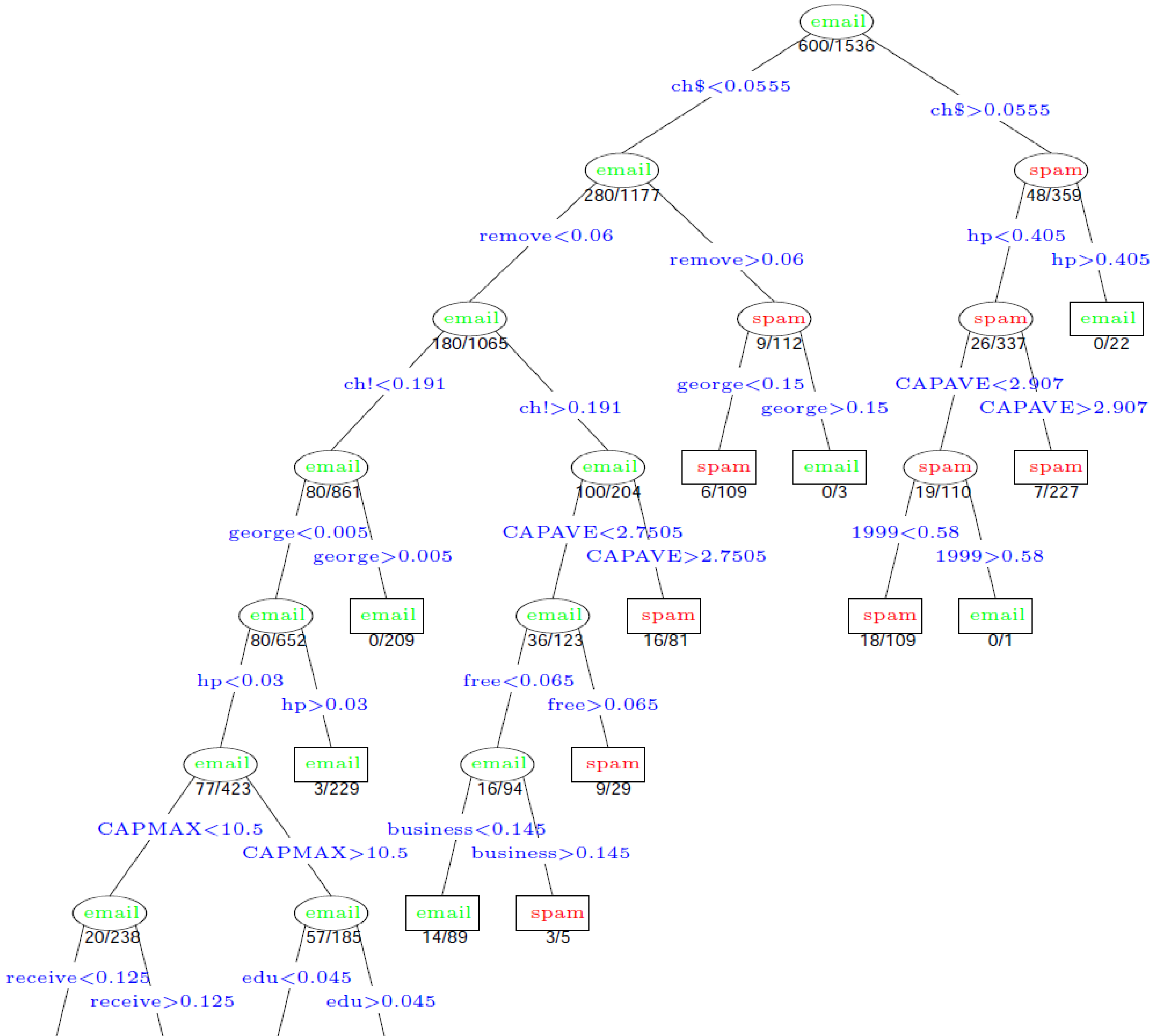
- $\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k),$: the proportion of class k on mode
- $k(m) = \arg\max_k \hat{p}_{mk}$ *: the majority class on node m*
- Instead of Qm(T) defined in(9.15) in regression, we have different measures Qm(T) of node impurity including the following:
- Misclassification 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): $-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$



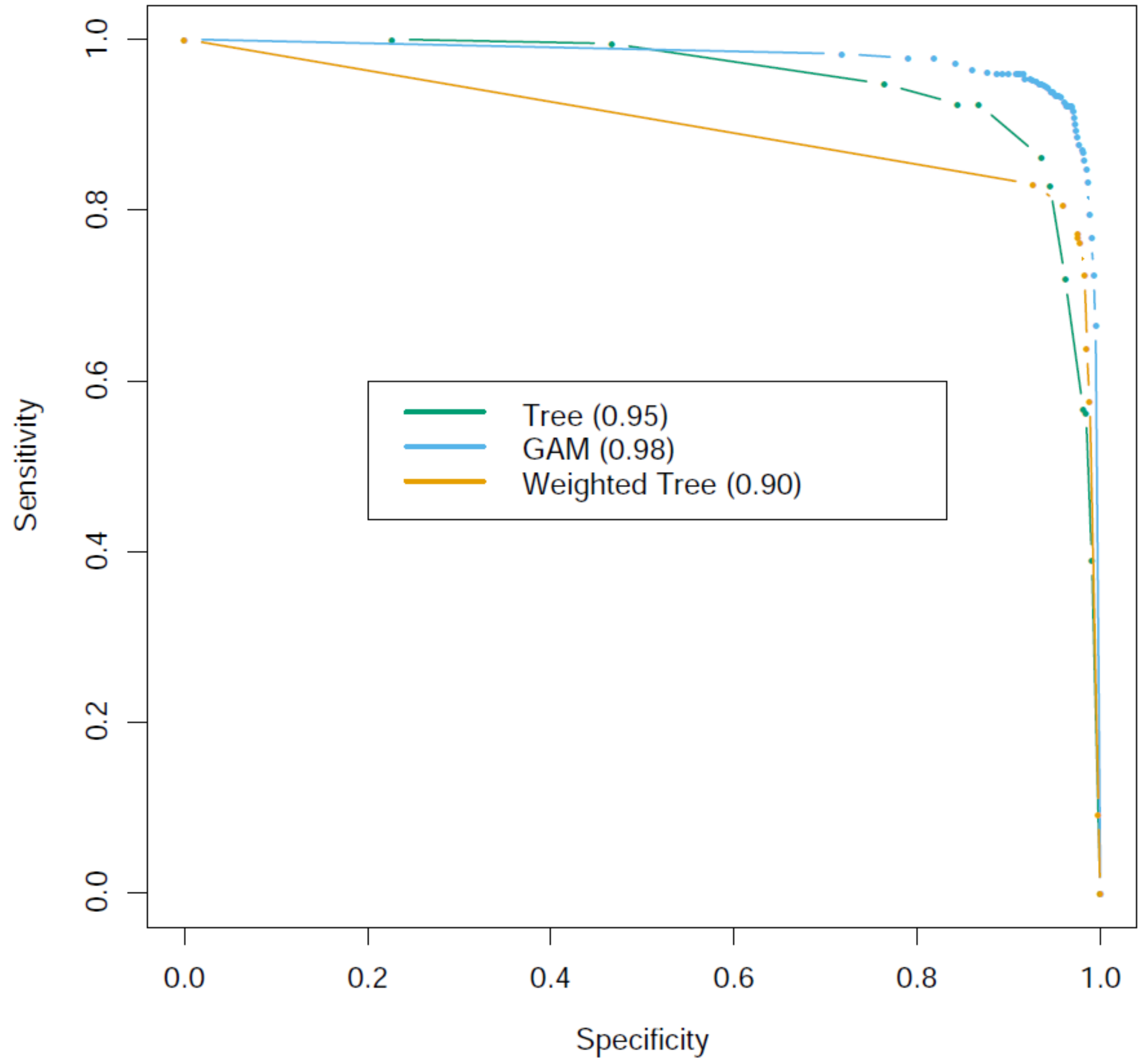
Classification Tree(cont.)

- Example: For two classes, if p is the proportion of in the second class, these three measures are:
  - $1-\max(p,1-p)$,
  - $2p(1-p)$,
  - $-p\log(p) -(1-p)\log(1-p)$




email 600/1536
ch$<0.0555
ch$>0.0555
email 280/1177
spam 48/359
remove<0.06
remove>0.06
hp<0.405
hp>0.405
email 180/1065
spam 9/112
spam 26/337
email 0/22
ch!<0.191
ch!>0.191
george<0.15
george>0.15
CAPAVE<2.907
CAPAVE>2.907
email 80/861
email 100/204
spam 6/109
email 0/3
spam 19/110
spam 7/227
george<0.005
george>0.005
CAPAVE<2.7505
CAPAVE>2.7505
1999<0.58
1999>0.58
email 80/652
email 0/209
email 36/123
spam 16/81
spam 18/109
email 0/1
hp<0.03
hp>0.03
free<0.065
free>0.065
email 77/423
email 3/229
email 16/94
spam 9/29
CAPMAX<10.5
CAPMAX>10.5
business<0.145
business>0.145
email 20/238
email 57/185
email 14/89
spam 3/5
receive<0.125
receive>0.125
edu<0.045
edu>0.045

*



Sensitivity: Probability of predicting spam given true state is spam

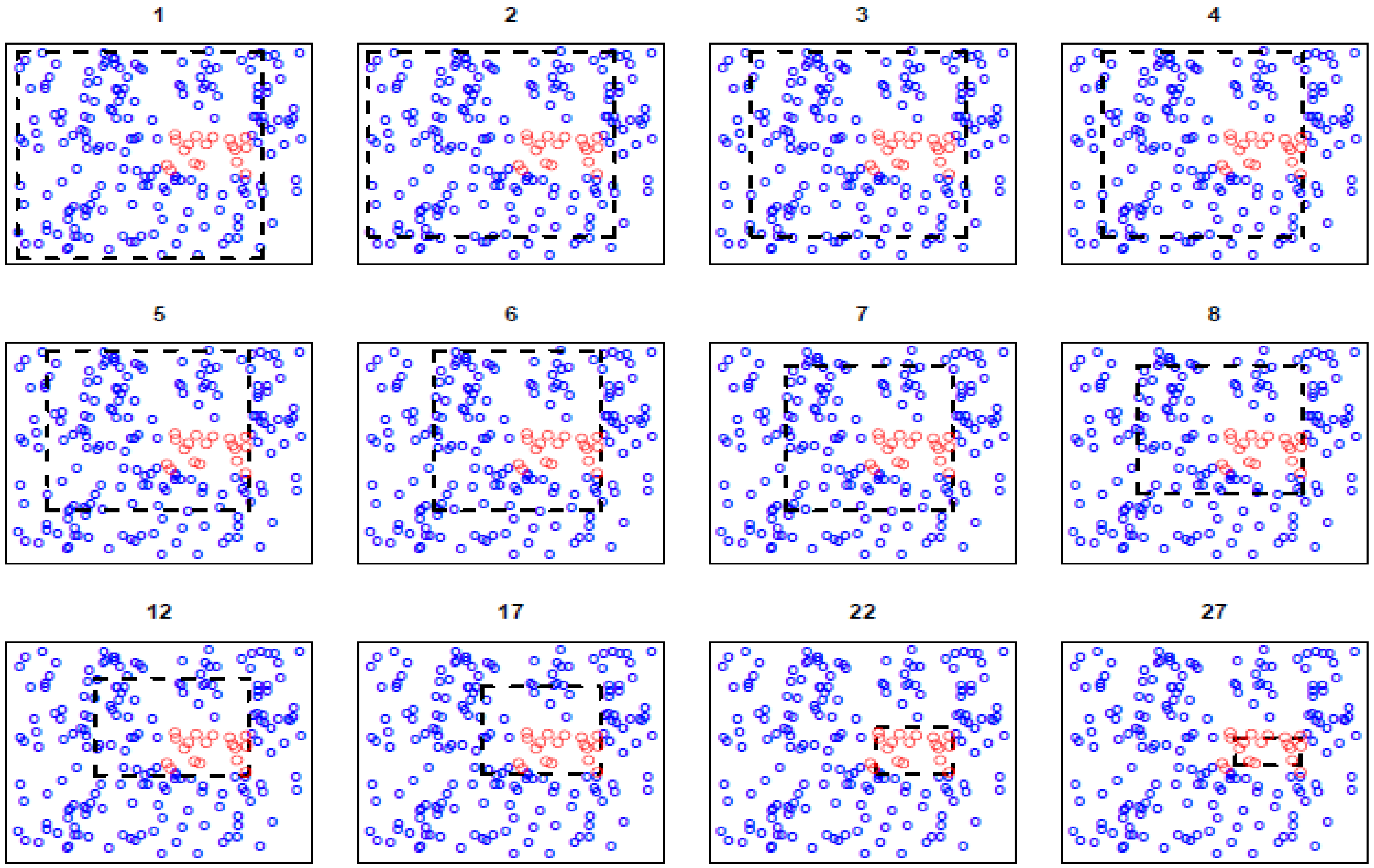
Specificity: Probability of predicting email given true state is email

Introduction

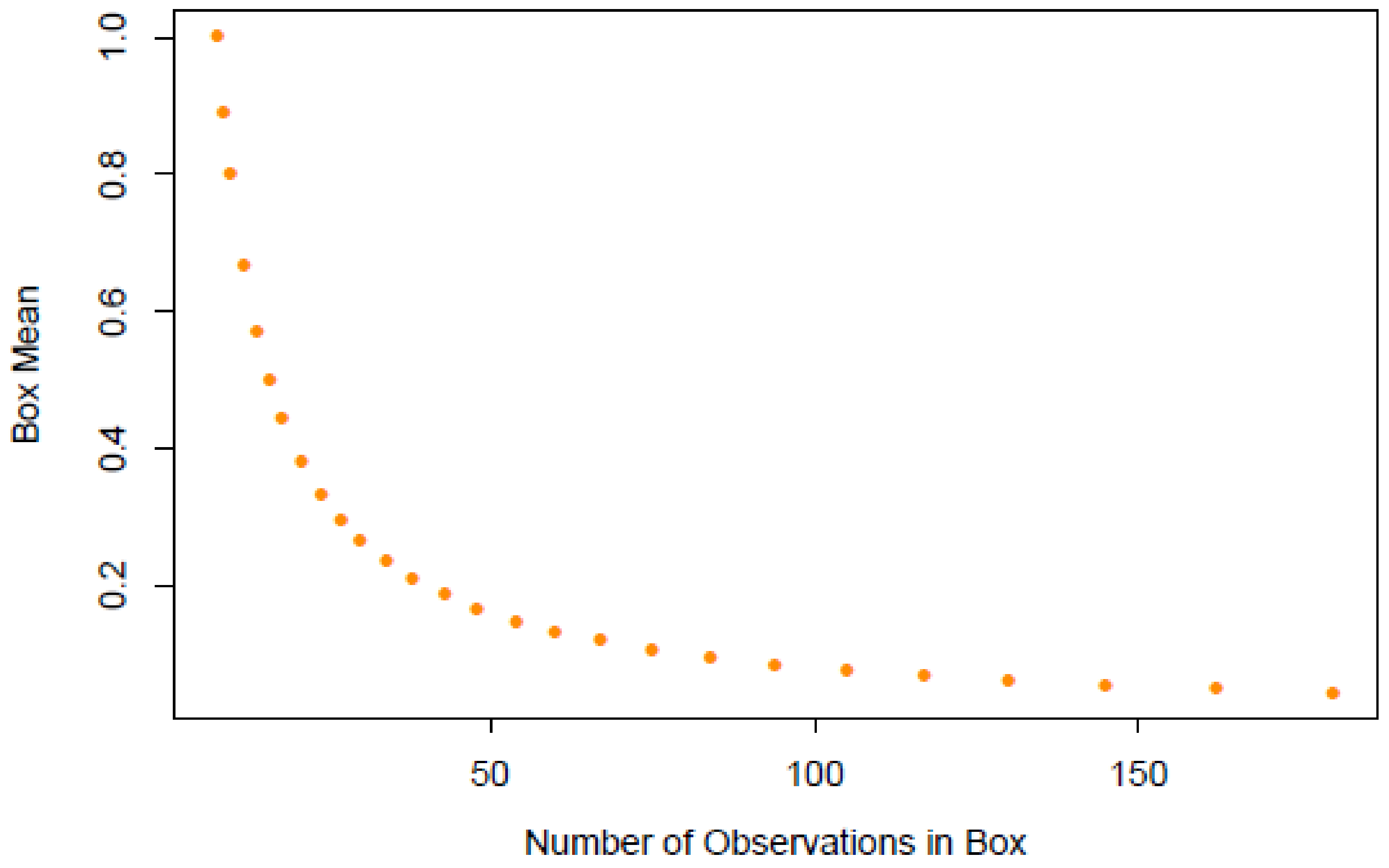
9.3 PRIM: Bump Hunting

- The patient rule induction method(PRIM) finds boxes in the feature space and seeks boxes in which the response average is high. Hence it looks for maxima in the target function, an exercise known as bump hunting.

PRIM(cont.)



PRIM(cont.)



1.0
0.8
0.6
0.4
0.2
Box Mean
50
100
150
Number of Observations in Box

Introduction

MARS: Multivariate Adaptive Regression Splines

- MARS uses expansions in piecewise linear basis functions of the form: (x-t)+ and (t-x)+. We call the two functions a reflected pair.

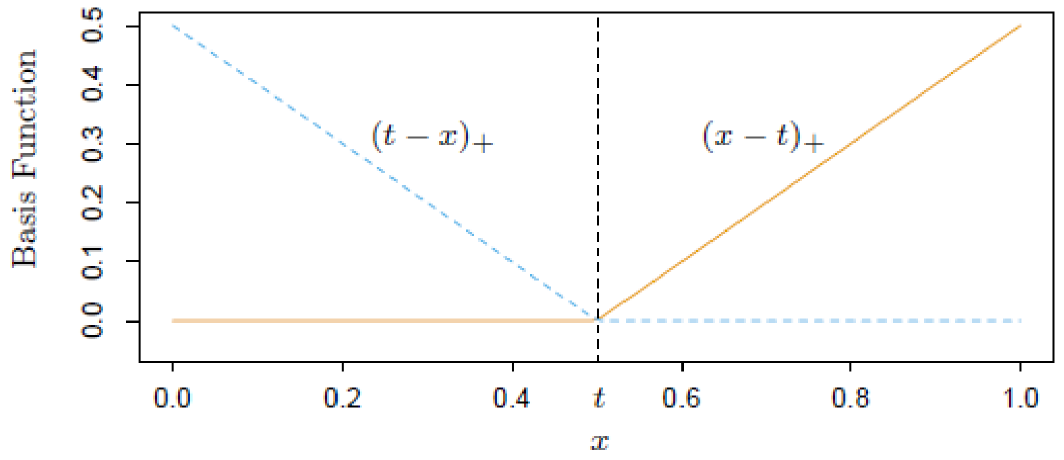


FIGURE 9.9. *The basis functions $(x - t)_+$ (solid orange) and $(t - x)_+$ (broken blue) used by MARS.*

MARS(cont.)

- The idea of MARS is to form reflected pairs for each input X_j with knots at each observed value X_{ij} of that input. Therefore, the collection of basis function is:

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

- The 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 C or a product of two or more such functions.

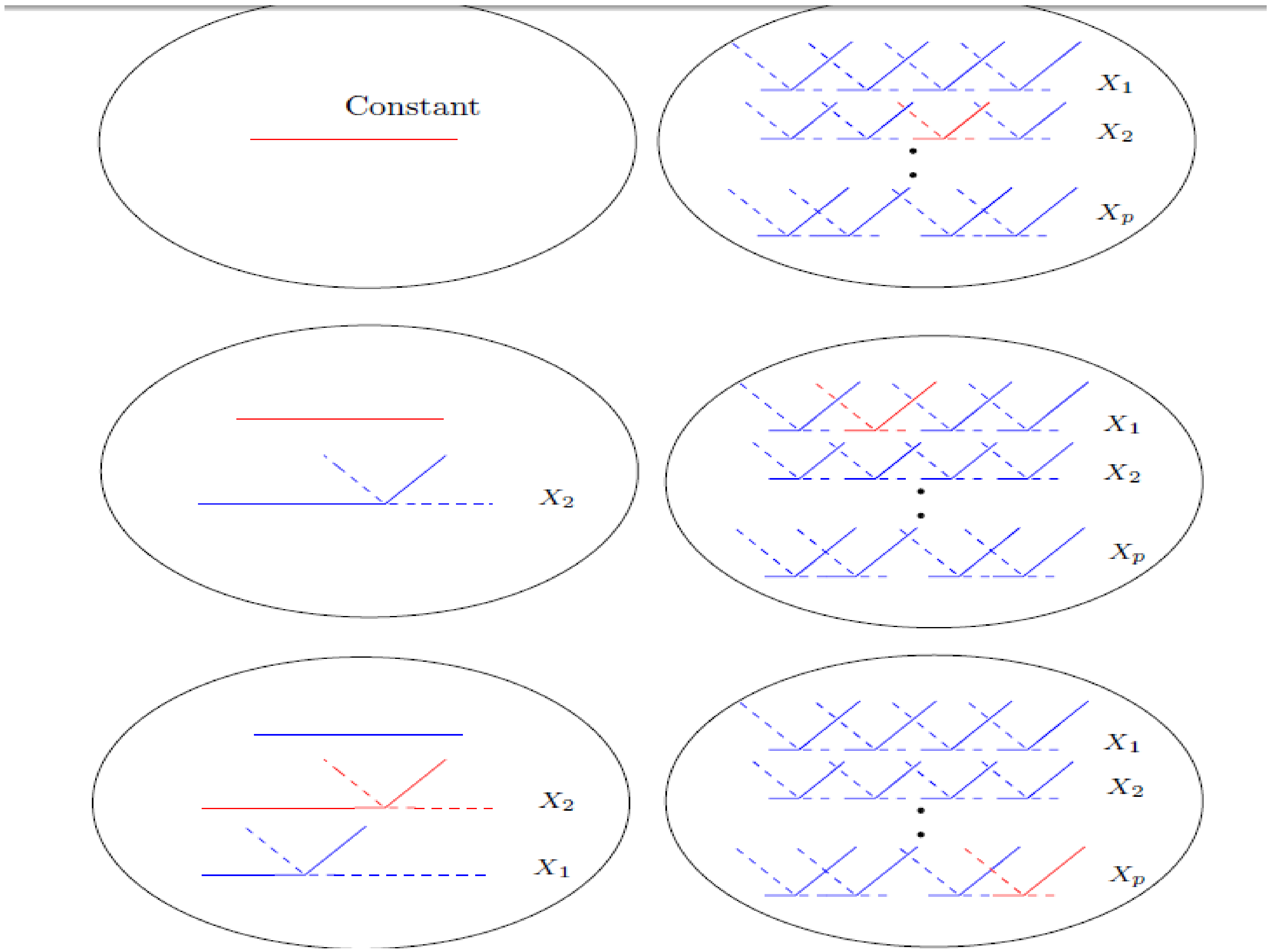
MARS(cont.)

- We start with only the constant function $h_0(x)=1$ in the model set M and all functions in the set C are candidate functions. At each stage we add to the model set M the term of the form:

$$\hat{\beta}_{M+1} h_l(X)(X_j - t)_+ + \hat{\beta}_{M+2} h_l(X)(t - X_j)_+, h_l(X) \in \mathbb{M}$$

that produces the largest decrease in training error. The process is continued until the model set M contains some preset maximum number of terms.

MARS(cont.)



MARS(cont.)

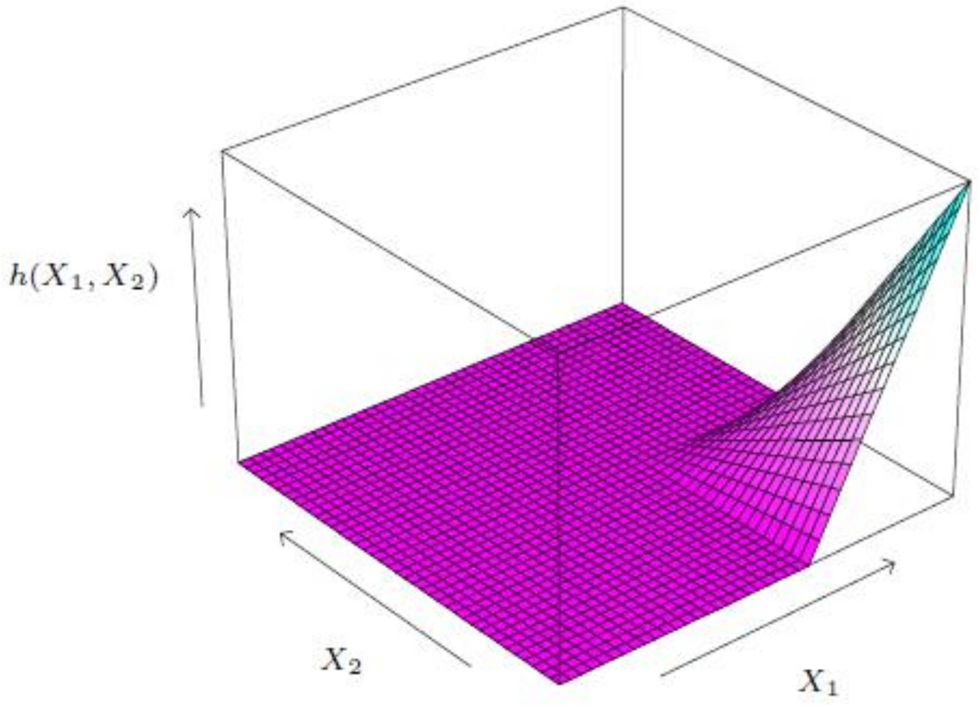


FIGURE 9.11. *The function* $h(X_1, X_2) = (X_1 - x_{51})_+ \cdot (x_{72} - X_2)_+$, *resulting from multiplication of two piecewise linear MARS basis functions.*

MARS(cont.)

- This progress typically overfits the data and so a backward deletion procedure is applied. The term whose removal causes the smallest increase in RSS is deleted from the model at each step, producing an estimated best model $\hat{f}_\lambda$ of each size λ .
- Generalized cross validation is applied to estimate the optimal value of λ

$$\mathrm{GCV}(\lambda) = \frac{\sum_{i=1}^{N} (y_i - \hat{f}_\lambda(x_i))^2}{(1 - M(\lambda)/N)^2} .$$

The value $M(\lambda)$ is the effective number of parameters in the model.

Introduction

Hierarchical Mixtures of Experts

- The HME method can be viewed as a variant of the tree-based methods.

Difference:

- The main difference is that the tree splits are not hard decisions but rather soft probabilistic ones.
- In an HME a linear(or logistic regression) model is fitted in each terminal node, instead of a constant as in the CART.

HME(cont.)

- A simple two-level HME model is shown in Figure. It can be viewed as a tree with soft splits at each non-terminal node.

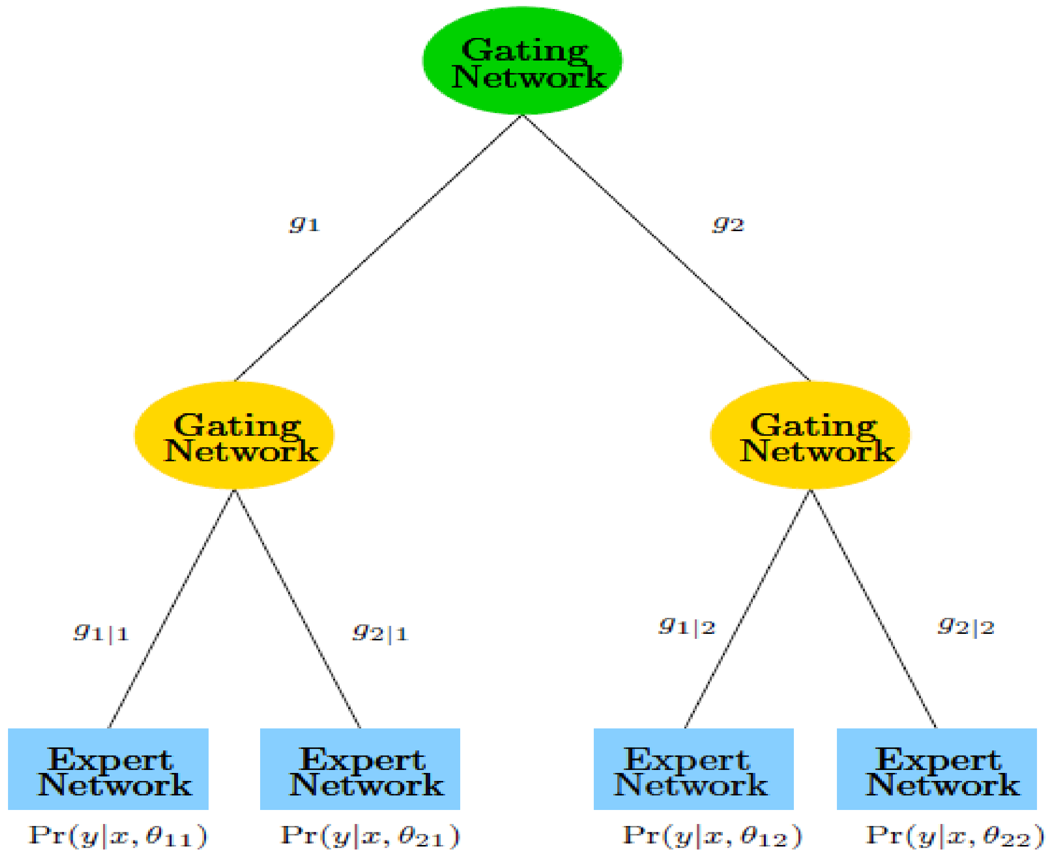


FIGURE 9.13. *A two-level hierarchical mixture of experts (HME) model.*

HME(cont.)

- The terminal node is called expert and the non-terminal node is called gating networks.
  - each expert provides a prediction about the response Y,
  - these are combined together by the gating networks.
- The top gating network has the output:

$$g_j(x, \gamma_j) = \frac{e^{\gamma_j^T x}}{\sum_{k=1}^{K} e^{\gamma_k^T x}}, \ j = 1, 2, \ldots, K, \qquad (9.25)$$

- where each γ_j is a vector of unknown parameters.
- Each $g_j(x, \gamma_j)$ is the probability of assigning an observation with feature vector x to the jth branch.

HME(cont.)

- At the second level, the gating networks have a similar form:

$$g_{\ell|j}(x, \gamma_{j\ell}) = \frac{e^{\gamma_{j\ell}^T x}}{\sum_{k=1}^{K} e^{\gamma_{jk}^T x}}, \ell = 1, 2, \ldots, K.$$

- At the experts, we have a model for the response variable of the form:

$$Y \sim \Pr(y \mid x, \theta_{j\ell}).$$

This differs according to the problem.

HME(cont.)

Regression: The Gaussian linear regression model is used, with

$$\theta_{j\ell} = (\beta_{j\ell}, \sigma_{j\ell}^2):$$

$$Y = \beta_{j\ell}^T x + \varepsilon \text{ and } \varepsilon \sim N(0, \sigma_{j\ell}^2).$$

Classification: The linear logistic regression model is used:

$$\Pr(Y = 1 | x, \theta_{j\ell}) = \frac{1}{1 + e^{-\theta_{j\ell}^T x}}.$$

Denoting the collection of all parameters by $\Psi = \{\gamma_j, \gamma_{j\ell}, \theta_{j\ell}\}$, the total probability that $Y = y$ is

$$\Pr(y | x, \Psi) = \sum_{j=1}^{K} g_j(x, \gamma_j) \sum_{\ell=1}^{K} g_{\ell|j}(x, \gamma_{j\ell}) \Pr(y | x, \theta_{j\ell}). \qquad (9.30)$$

This is a mixture model, with the mixture probabilities determined by the gating network models.

SJTU

THE END of Talk