# Tree-Based Model

Siyu Wang, Tingyin Wang

School of Management
University of Science and Technology of China

2021.12.29

# Table of Contents

1. Generalized Additive Models
   - Univariate Local Smoother
   - Multiple Covariates
   - Projection Pursuit Regression
2. Introduction to tree methods
3. A Practical Guide to Tree Construction
4. A Review of Tree Methods
5. Introduction to MARS

# Table of Contents

1. Generalized Additive Models
    - Univariate Local Smoother
    - Multiple Covariates
    - Projection Pursuit Regression

2. Introduction to tree methods

3. A Practical Guide to Tree Construction

4. A Review of Tree Methods

5. Introduction to MARS

# Scatterplot Smoothers for Linear Model

- The case of a single predictor is

$$E(Y \mid X) = s(X)$$

- To estimate $s(x)$ from data, we can use any reasonable estimate of $E(Y \mid X = x)$. One class of estimates are the **local average estimates**:

$$\hat{s}\left(x_i\right) = \mathrm{Ave}_{j \in N_i}\left\{y_j\right\},$$

  where Ave represents some averaging operator like the mean and $N_i$ is a neighborhood of $x_i$. Here what we consider are **symmetric nearest neighborhoods**.

# Local Linear smoother

- If Ave stands for arithmetic mean, then $\hat{s}(\cdot)$ is the **running mean**, a very simple scatterplot smoother. The running mean is not a satisfactory smoother because it creates large biases at the end points and doesn't generally reproduce straight lines.
- A slight refinement of the running average, the **running lines smoother** alleviates these problems. The running lines estimate is defined by

$$\hat{s}\left(x_i\right) = \hat{\beta}_{0i} + \hat{\beta}_{1i} x_i .$$

# Local Likelihood

- To estimate the model $\eta = s(X)$, the local likelihood procedure generalizes this by assuming that locally $s(x)$ is linear and fits a line in a neighborhood around each $X$ value.
- In the exponential family with canonical link, the local likelihood estimate of $s\left(x_i\right)$ is defined as

$$\hat{s}\left(x_i\right) = \hat{\beta}_{0i} + \hat{\beta}_{1i} x_i,$$

where $\hat{\beta}_{0i}$ and $\hat{\beta}_{1i}$ maximize the local log likelihood

$$\log L_i = \sum_j K_\lambda(i,j)\left\{\frac{y_j\theta_{ij} - b\left(\theta_{ij}\right)}{a(\phi)} + c\left(y_j, \phi\right)\right\},$$

and $\theta_{ij} = \beta_{0i} + \beta_{1i} x_j$, $\lambda$ is the bandwidth parameter.

# Local Scoring

- Recall: Generalized linear model uses an algorithm called **iteratively reweighted least square** (IRLS). Given $\hat{\eta}$ (a current fitting value of the linear predictor) and $\mu$ (a current estimate of the mean for $y$), we form the adjusted dependent variable

  $$z = \hat{\eta} + (y - \hat{\mu})(\frac{d\eta}{d\mu}).$$

  Define weights $W$ by

  $$(W)^{-1} = \left(\frac{d\eta}{d\mu}\right)^2 V,$$

  where $V$ is the variance of $y$ at $\mu = \hat{\mu}$.
- The regression algorithm proceeds by regressing $z$ on $1, x_1, \cdots, x_p$ with weights $W$ to obtain an estimate $\hat{\beta}$.
- Given $\hat{\beta}$, a new $\hat{\mu}$ and $\hat{\eta}$ are computed.

# Local Scoring

- We can develop a smoothing step within IRLS:

  $$\eta^1(x) = \text{smooth}[\eta(x) + (y - \mu)(d\eta/d\mu)]$$

  with weights $(d\mu/d\eta)^2 V^{-1}$ and smooth represents the weighted line smoother regression on the samples that are closed to $x$.
- This is exactly a smooth of the adjusted dependent variable.

# Multivariate nonlinear model

- We assume
$$Y = \eta(\mathbf{X}) + \varepsilon$$
where $\eta(\mathbf{X}) = E(Y \mid \mathbf{X}), \mathrm{Var}(Y \mid \mathbf{X}) = \sigma^2$, and the errors $\varepsilon$ are independent of $\mathbf{X}$.
- The goal is to estimate $\eta(\mathbf{X})$. If we use the least squares criterion $E(Y - \eta(\mathbf{X}))^2$, the best choice for $\eta(\mathbf{X})$ is $E(Y \mid \mathbf{X})$.
- In the case of a single covariate, we estimated $E(Y \mid X)$ by a scatterplot smoother which in its crudest form is the average of those $y_i$ in the sample for which $x_i$ is close to $x$.

# Multivariate nonlinear model

- This is the chief motivation for the additive model

$$\eta(\mathbf{X}) = s_0 + \sum_{j=1}^{p} s_j\left(X_j\right)$$

- Each function is estimated by smoothing on a single co-ordinate; we can thus include sufficient points in the neighborhoods to keep the variance of the estimates down and remain local in each co-ordinate.
- The additive model itself may be a biased estimate of the true regression surface, but hopefully this bias is much lower than that produced by high dimensional smoothers.

# Estimation-The Additive Regression Model

- We now turn to the estimation of $s_0, s_1(\cdot), \cdots, s_p(\cdot)$ in the additive regression model

$$E(Y \mid \mathbf{X}) = s_0 + \sum_{j=1}^{p} s_j\left(X_j\right)$$

where $Es_j\left(X_j\right) = 0$ for every $j$.

- In order to motivate the algorithm, suppose the model

$$Y = s_0 + \sum_{j=1}^{p} s_j\left(X_j\right) + \varepsilon$$

is in fact correct, and we define the partial residual:

$$R_j = Y - s_0 - \sum_{k \neq j} s_k\left(X_k\right).$$

# Estimation-The Additive Regression Model

**Backfitting Algorithm**

- Initialization:

$$s_0 = E(Y), s_1^1(\cdot) \equiv s_2^1(\cdot) \equiv \cdots \quad \equiv s_p^1(\cdot) \equiv 0, \quad m = 0.$$

Iterate: $m = m + 1$

$$R_j = Y - s_0 - \sum_{k=1}^{j-1} s_k^m(X_k) - \sum_{k=j+1}^{p} s_k^m(X_k)$$

$$s_j^m(X_j) = E(R_j \mid X_j).$$

Until: RSS $= E\left(Y - s_0 - \sum_{j=1}^{p} s_j^m(X_j)\right)^2$ fails to decrease.

# Backfitting in the Local Scoring Algorithm

- For multiple covariates the local scoring update is given by (the **exponential family** case)

$$\begin{aligned}\eta^1(\mathbf{x}) &= E[\eta(\mathbf{x}) + (Y - \mu)(\partial\eta/\partial\mu) \mid \mathbf{x}] \\ &= E(Z \mid \mathbf{x})\end{aligned}$$

  where $g(\mu) = \eta$ and $Z$ is the adjusted dependent variable.

# Backfitting in the Local Scoring Algorithm

- For the reasons described in the previous section, we will restrict attention to an additive model:

$$\eta(\mathbf{X}) = s_0 + \sum_{j=1}^{p} s_j\left(X_j\right)$$

- To estimate the $s_j(\cdot)$ 's, we fit an additive regression model to $Z$, treating it as the response variable $Y$ before. The sum of the fitted functions is $\eta^0$ of the next iteration.
- This is the motivation for the general local scoring algorithm which we give for the exponential family case.

# Backfitting in the Local Scoring Algorithm

**General Local Scoring Algorithm**

- Initialization:

$$s_0 = g(E(y)), \quad s_1^0(\cdot) \equiv s_2^0(\cdot) \equiv \cdots \equiv s_p^0(\cdot) \equiv 0, \quad m = 0.$$

Iterate: $m = m + 1$

1. Form the adjusted dependent variable

$$Z = \eta^{m-1} + \left(Y - \mu^{m-1}\right)\left(\partial\eta/\partial\mu^{m-1}\right),$$

where

$$\eta^{m-1} = s_0 + \sum_{j=1}^{p} s_j^{m-1}(X_j) \quad \text{and}$$

$$\eta^{m-1} = g\left(\mu^{m-1}\right)$$

2. Form the weights $W = \left(\partial\mu/\partial\eta^{m-1}\right)^2 V^{-1}$.

3. Fit an additive model to $Z$ using the backfitting algorithm with weights $W$, we get estimated functions $s_j^m(\cdot)$ and model $\eta^m$.

Until: $E\,\text{dev}\,(Y, \mu^m)$ fails to decrease.

# Projection Pursuit Regression

## Definition

- As in our generic supervised learning problem, assume we have an input vector $X$ with $p$ components, and a target $Y$. Let $\omega_m, m = 1, 2, \ldots, M$, be unit $p$-vectors of unknown parameters. The projection pursuit regression (PPR) model has the form

$$f(X) = \sum_{m=1}^{M} g_m \left( \omega_m^T X \right)$$

- This is an **additive model**, but in the derived features $V_m = \omega_m^T X$ rather than the inputs themselves. The functions $g_m$ are unspecified and are estimated along with the directions $w_m$ using some flexible smoothing method.

# Projection Pursuit Regression

**How do we fit a PPR model ?**

- Given training data $(x_i, y_i)\,, i = 1, 2, \ldots, N$? We seek the approximate minimizers of the error function

$$\sum_{i=1}^{N}\left[y_i - \sum_{m=1}^{M} g_m\left(\omega_m^T x_i\right)\right]^2$$

over functions $g_m$ and direction vectors $w_m, m = 1, 2, ..., M$

# Projection Pursuit Regression

**How do we fit a PPR model ?**

- **First Step: Given $\omega$, estimate $g$**
  Consider just one term ($M = 1$). Given the direction vector $\omega$, we form the derived variables $v_i = \omega^T x_i$. Then we have a one-dimensional smoothing problem, and we can apply any scatterplot smoother, such as a smoothing spline, to obtain an estimate of $g$.

# Projection Pursuit Regression

**How do we fit a PPR model ?**

- **Second Step: Given $g$, estimate $\omega$**
  A Gauss-Newton search is convenient for this task. Let $\omega_{\text{old}}$ be the current estimate for $\omega$. We write

$$g\left(\omega^T x_i\right) \approx g\left(\omega_{\text{old}}^T x_i\right) + g'\left(\omega_{\text{old}}^T x_i\right)\left(\omega - \omega_{\text{old}}\right)^T x_i$$

to give

$$\sum_{i=1}^{N}\left[y_i - g\left(\omega^T x_i\right)\right]^2 \approx$$

$$\sum_{i=1}^{N} g'\left(\omega_{\text{old}}^T x_i\right)^2\left[\left(\omega_{\text{old}}^T x_i + \frac{y_i - g\left(\omega_{\text{old}}^T x_i\right)}{g'\left(\omega_{\text{old}}^T x_i\right)}\right) - \omega^T x_i\right]^2$$

# Projection Pursuit Regression

**How do we fit a PPR model ?**

- To minimize the right-hand side, we carry out a least squares regression with target $\omega_{\text{old}}^{T} x_i + \left(y_i - g\left(\omega_{\text{old}}^{T} x_i\right)\right) / g'\left(\omega_{\text{old}}^{T} x_i\right)$ on the input $x_i$, with weights $g'\left(\omega_{\text{old}}^{T} x_i\right)^2$ and no intercept (bias) term. This produces the updated coefficient vector $\omega_{\text{new}}$ .
- These two steps, estimation of $g$ and $\omega$, are iterated until convergence. With more than one term in the PPR model, the model is built in a forward stage-wise manner, adding a pair $(\omega_m, g_m)$ at each stage.

# Neural Networks

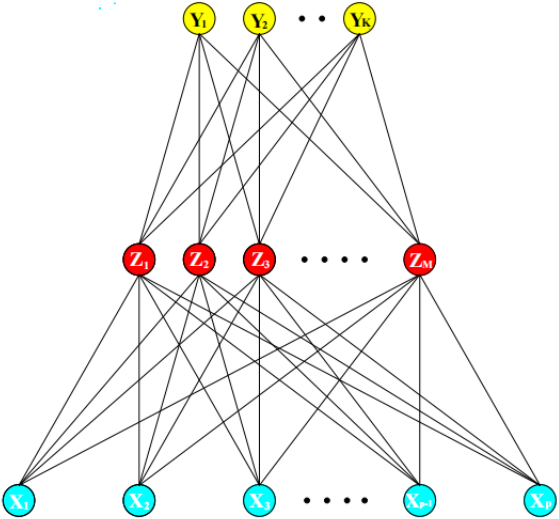
### Definition

- Derived features $Z_m$ are created from linear combinations of the inputs, and then the target $Y_k$ is modeled as a function of linear combinations of the $Z_m$,

$$Z_m = \sigma\left(\alpha_{0m} + \alpha_m^T X\right), m = 1, \ldots, M,$$
$$T_k = \beta_{0k} + \beta_k^T Z, k = 1, \ldots, K,$$
$$f_k(X) = g_k(T), k = 1, \ldots, K,$$

where $Z = (Z_1, Z_2, \ldots, Z_M)$, and $T = (T_1, T_2, \ldots, T_K)$.

# Neural Networks



# Neural Networks

- The activation function $\sigma(v)$ is usually chosen to be the sigmoid $\sigma(v) = 1/\left(1 + e^{-v}\right)$.
- The output function $g_k(T)$ can choose the identity function $g_k(T) = T_k$ or the softmax function

$$g_k(T) = \frac{e^{T_k}}{\sum_{\ell=1}^{K} e^{T_\ell}}$$

- We can think of the $Z_m$ as a basis expansion of the original inputs $X$; the neural network is then a standard linear model, or linear multilogit model.

# Comparison of PPR model and Neural Networks

- The neural network model with one hidden layer has exactly the same form as the projection pursuit model.
- The difference is that the PPR model uses nonparametric functions $g_m(v)$, while the neural network uses a far simpler function based on $\sigma(v)$, with three free parameters in its argument.
- In detail, viewing the neural network model as a PPR model, we identify

$$\begin{aligned} g_m\left(\omega_m^T X\right) &= \beta_m \sigma\left(\alpha_{0m} + \alpha_m^T X\right) \\ &= \beta_m \sigma\left(\alpha_{0m} + \|\alpha_m\| \left(\omega_m^T X\right)\right) \end{aligned}$$

  where $\omega_m = \alpha_m / \|\alpha_m\|$ is the $m$ th unit-vector.

# Table of Contents

1. Generalized Additive Models
    - Univariate Local Smoother
    - Multiple Covariates
    - Projection Pursuit Regression
2. Introduction to tree methods
3. A Practical Guide to Tree Construction
4. A Review of Tree Methods
5. Introduction to MARS

# Regression Model

Many scientific problems reduce to model the relationship between two sets of variables. Regression methodology is designed to quantify these relationships. [16]

- Linear regression for continuous data.
- Logistic regression for binary data.
- Proportional hazard regression for censored survival data.
- Mixed-effect regression for longitudinal data.

These parametric (or semi-parametric) regression methods may not lead to faithful data descriptions when the underlying assumption is not satisfied.

# Recursive Partitioning Based Model

Nonparametric regression has evolved to relax or remove the restrictive assumptions. One type of these methods is recursive partitioning, which provides a useful alternative to the parametric regression methods.

- Classification and Regression Trees (CART).
- Multivariate Adaptive Regression Splines (MARS).
- Forest.
- Survival Trees.

In this lecture we focus our attention on CART and MARS.

# Areas of Applications

- Financial firms:
  - Banking crises [7].
  - Credit cards [12].
  - Investments [3].
- Manufacturing and marketing companies [13].
- Pharmaceutical industries[4].
- Engineering research.
  - Natural language speech recognition [2].
  - Musical sounds [15].
  - Text recognition [5].

# Biomedical Application

Yet the best documented, and arguably most popular uses of tree-based methods are in biomedical research for which classification is a central issue.

- Chest Pain
  - Goldman [9] builds an expert computer system that could assist physician in emergency room to classify patients with chest pain into homogeneous groups within a couple of ours.
  - In this example 10682 patients are included in the train set and 4676 patients are included in the test set.
- Coma
  - Predict the outcome from coma caused by cerebral hypoxia-ischemia.
  - They studied 210 patients with cerebral hypoxia-ischemia and considered 13 factors including age, sex, verbal and motor responses, and eye opening movement.

# Gene related Application

- Gene Expression
  - Zhang [11] analyzed a data set from the expression profiles.
  - 2,000 genes in 22 normal and 40 colon cancer tissues [1].
- Associate Study
  - Zhang analyzed 360 highly polymorphic markers on six chromosomes, each has 3-9 alleles of 200 nuclear families with at least 1 affected child and 100 control families with no affected members, resulting in a total of 1484 individuals.

# Table of Contents

1. Generalized Additive Models
   - Univariate Local Smoother
   - Multiple Covariates
   - Projection Pursuit Regression
2. Introduction to tree methods
3. A Practical Guide to Tree Construction
4. A Review of Tree Methods
5. Introduction to MARS

# The Elements of Tree

- Root node: the circle on the top, contains all samples that are used to build tree.
- Split: the process of separating one node into two.
- Terminal node: also know as 'leaf node', the nodes that make no further split.
- Internal node: nodes that are neither root node or terminal node.
- Left and right daughter nodes: the two offspring nodes generated from a split.

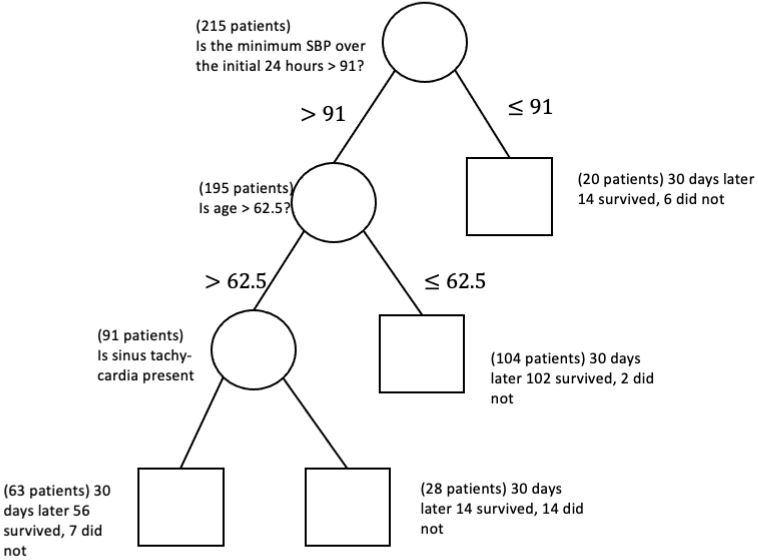
# Questions to Answer

- What is the purpose of growing a tree?
- How is a parent node split into two daughter nodes?
- When do we declare a terminal node?
- How do we make predict?

# An Example

- In CART, Breiman provides an example of 216 heart attack patients with 19 covariates, including blood pressure, age and 17 other ordered and binary variables measured in the first 24 hours, summerizing the medical symptoms considered as an important indicators of the patient's condition.

# An Example


(215 patients)
Is the minimum SBP over the initial 24 hours > 91?
> 91
≤ 91
(195 patients)
Is age > 62.5?
(20 patients) 30 days later 14 survived, 6 did not
> 62.5
≤ 62.5
(91 patients)
Is sinus tachycardia present
(104 patients) 30 days later 102 survived, 2 did not
(63 patients) 30 days later 56 survived, 7 did not
(28 patients) 30 days later 14 survived, 14 did not

# Aim of Recursive Partitioning

Using the information from covariates of a sample to partition the sample space and produce homogeneous sub-space, so that we can make prediction more easily in each subspace.

# Allowable split

- Ordinal predictor
  - Consider variable $x_1$ (age), 32 distinct age values in the range of 13 to 46, giving 31 allowable splits.
  - For an ordinal predictor, the number of allowable splits is one fewer than the number of its distinctly observed values.
- Nominal predictor
  - Consider race that has 5 levels, in total $2^{5-1} - 1 = 15$ allowable splits.
  - Any nominal variable that has $k$ levels contributes $2^{k-1} - 1$ allowable splits (the subsets of all level are assigned to one node and the rest to another).

# Goodness of Splits

The goodness of a split must quantify the homogeneities (or impurities) of two child nodes. To define the goodness of split, we usually define the node impurity as $i(\tau)$ first for node $\tau$, and the goodness of split can thus be defined as:

$$\Delta i(\tau, s) = i(\tau) - P\left(\tau_{l}\right) i\left(\tau_{l}\right) - P\left(\tau_{r}\right) i\left(\tau_{r}\right),$$

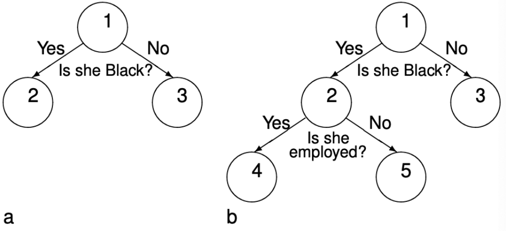
where $P\left(\tau_{l}\right)$ is the proportion of nodes on the left, and similarly for $P\left(\tau_{r}\right)$, which means the decrease of node impurity after the split. The split $s$ is chosen to maximize $\Delta i(\tau, s)$.

# Node Impurity

The node impurity that we often use in practice:

- Node impurity ordinal predictor
  - Variance $\frac{1}{K}\sum_{j=1}^{K}(Y_j - \bar{Y})$.
- Node impurity of nominal predictor
  - Entropy: $\sum_{j=1}^{K} p_j \log \frac{1}{p_j}$.
  - Misclassification rate: $1 - max_j p_j$.
  - Gini index: $\sum_{j=1}^{K} p_j\left(1 - p_j\right) = 1 - \sum_{j=1}^{K} p_j^2$.

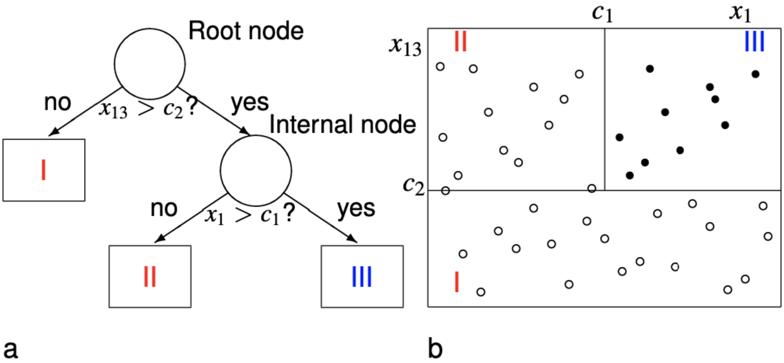
# Recursive Partitioning



- We chose one of the predictor to proceed with the node split (usually the variable maximizing the goodness of splits).
- After splitting the root node (a), we continue to divide its into two daughter nodes (b) by selecting the predictor that again maximizing the goodness of splits.
- We do this iteratively until some stopping criteria are met.

# Regression Tree

The following figure how we partition sample space (b) by generating a tree (a).



# Terminal Nodes

Without a stopping rule, the recursive partitioning process may proceed until the tree is saturated in the sense that the offspring nodes subject to further division cannot be split. (e.g. there is only one subject in a node.) The saturated tree is usually too large and useless.

- The terminal nodes are so small that we cannot make sensible statistical inference.
- This level of detail is rarely scientifically interpretable.

# Stopping Rules and Tree Pruning

To solve the above issue, there are usually two methods to be adopted.

- Adopting threshold: Breiman et al. argued that depending on the stopping threshold, such as that the decrease of node impurity is larger than 1% or that the minimum sample size within a leaf is larger than 5. However, the partitioning tends to end too soon or too late when using such rule.
- Pruning: Find a subtree of the saturated tree that is most "predictive" of the outcome and least vulnerable to the noise in the data.

# Cost complexity

- For a given tree $\tilde{\mathcal{T}} \subset \mathcal{T}$ and $\alpha$, we define cost-complexity as follows:

$$R_\alpha(\tilde{\mathcal{T}}) = \sum_{m=1}^{|\tilde{\mathcal{T}}|} N_m i(r_m) + \alpha|\tilde{\mathcal{T}}|,$$

  where $\alpha(\geq 0)$ is the cost complexity parameter, $|\tilde{\mathcal{T}}|$ is the number of terminal nodes in $\tilde{\mathcal{T}}$, $r_m$ is the terminal node of $\tilde{\mathcal{T}}$, and $N_m$ is the number of samples in terminal node $r_m$.

- For a given $\alpha$, we choose the tree to be 'optimal under given $\alpha$' if it minimizes $R_\alpha(\tilde{\mathcal{T}})$. Increasing $\alpha$ continuously allows us to construct a sequence of nested 'optimal' subtrees from any given tree $\mathcal{T}$ so that we can examine the properties of these subtrees and make a selection from them.

# Cost-Complexity

- For a given tree $\mathcal{T}$ and $\alpha$, is there only one subtree of $\mathcal{T}_{\alpha}$ that has the 'smallest' cost-complexity?

**Theorem**

*For any value of the complexity parameter $\alpha$, there is a unique smallest subtree of that minimizes the cost-complexity.*

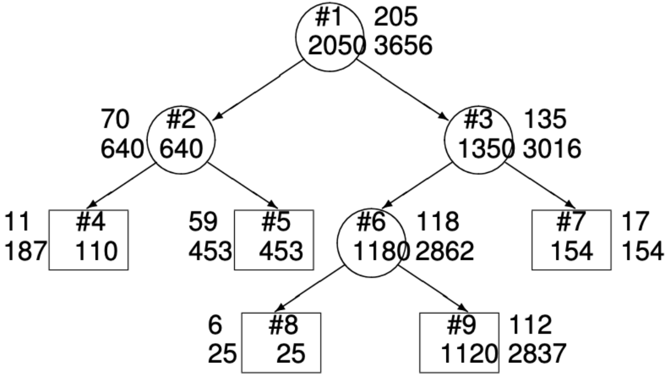
- We cannot have two subtrees of the smallest size and of the same cost-complexity.
- This smallest subtree is referred to as the optimal subtree with respect to the complexity parameter.
- When $\alpha = 0$, the optimal subtree is $\mathcal{T}$ itself.

# Properties of Pruned Tree

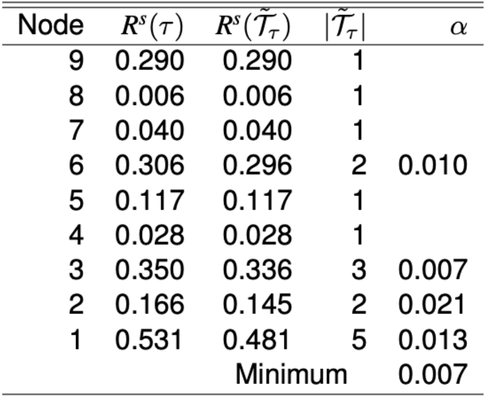
- Not all subtrees are optimal with respect to a complexity parameter.
- Although the complexity parameter takes a continuous range of values, we have only a finite number of subtrees.
- An optimal subtree is optimal for an interval range of the complexity parameter, and the number of such intervals has to be finite.
- We now show that with the continuously increase of $\alpha$, the corresponding optimal subtrees form a nested sequence.

# Nested Optimal Subtrees

Denote $\tilde{\mathcal{T}}_\tau$ as the subtree of $\mathcal{T}$ which has $\tau$ as its root node.



# Nested Optimal Subtrees



| Node | $R^s(\tau)$ | $R^s(\tilde{T}_\tau)$ | $\|\tilde{T}_\tau\|$ | $\alpha$ |
|---|---|---|---|---|
| 9 | 0.290 | 0.290 | 1 | |
| 8 | 0.006 | 0.006 | 1 | |
| 7 | 0.040 | 0.040 | 1 | |
| 6 | 0.306 | 0.296 | 2 | 0.010 |
| 5 | 0.117 | 0.117 | 1 | |
| 4 | 0.028 | 0.028 | 1 | |
| 3 | 0.350 | 0.336 | 3 | 0.007 |
| 2 | 0.166 | 0.145 | 2 | 0.021 |
| 1 | 0.531 | 0.481 | 5 | 0.013 |
| | | Minimum | | 0.007 |

# Nested Optimal Subtrees

We can draw several conclusions from the above table.

- The right column of the above table gives the minimum $\alpha$ needed to replace the subtree from that node to that single node.
- We gradually cut all the subtrees satisfying

  $$R^s(\tau)+\alpha_1 \leq R^s\left(\tilde{\mathcal{T}}_\tau\right)+\alpha_1\left|\tilde{\mathcal{T}}_\tau\right|$$

  .
- We can now see why certain subtree cannot be optimal, and that we only have finite subtrees.

# Nested Optimal Subtrees

- In general, suppose that we end up with m thresholds, $0 < \alpha_1 < \alpha_2 < \cdots < \alpha_m$, than the corresponding optimal subtrees has the relationship that $\mathcal{T}_{\alpha_0} \supset \mathcal{T}_{\alpha_1} \supset \mathcal{T}_{\alpha_2} \supset \cdots \supset \mathcal{T}_{\alpha_m}$, where $\mathcal{T}_{\alpha_1} \supset \mathcal{T}_{\alpha_2}$ means that $\mathcal{T}_{\alpha_2}$ is the subtree of $\mathcal{T}_{\alpha_1}$.

**Theorem**

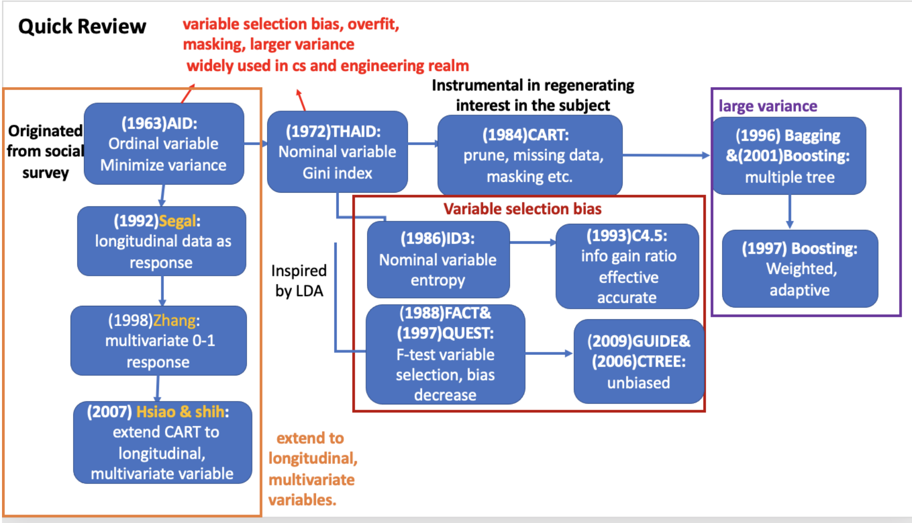
*If $\alpha_1 > \alpha_2$ the optimal subtree corresponding to $\alpha_1$ is a subtree of the optimal subtree corresponding to $\alpha_2$.*

We then use cross-validation to make selection from the optimal tree.

# Table of Contents

1. Generalized Additive Models
   - Univariate Local Smoother
   - Multiple Covariates
   - Projection Pursuit Regression
2. Introduction to tree methods
3. A Practical Guide to Tree Construction
4. A Review of Tree Methods
5. Introduction to MARS

# A Review of Tree Method



Quick Review

variable selection bias, overfit, masking, larger variance

widely used in cs and engineering realm

Instrumental in regenerating interest in the subject

Originated from social survey

(1963)AID: Ordinal variable Minimize variance

(1972)THAID: Nominal variable Gini index

(1984)CART: prune, missing data, masking etc.

large variance

(1996) Bagging &(2001)Boosting: multiple tree

(1997) Boosting: Weighted, adaptive

(1992)Segal: longitudinal data as response

(1998)Zhang: multivariate 0-1 response

(2007) Hsiao & shih: extend CART to longitudinal, multivariate variable

Variable selection bias

(1986)ID3: Nominal variable entropy

(1993)C4.5: info gain ratio effective accurate

(1988)FACT& (1997)QUEST: F-test variable selection, bias decrease

(2009)GUIDE& (2006)CTREE: unbiased

Inspired by LDA

extend to longitudinal, multivariate variables.

# A Review of Tree Method

We can see from the above diagram that tree methods develop in the following several ways.[14]

- Develop methods to decrease variable selection bias.
- Develop methods to reduce variable(usually ensemble learning).
- Invent new impurity function so that we can predict longitudinal, multivariate variables.
- Now efforts have been made to develop new tree framework that can be used to do statistical inference.

# Table of Contents

1. Generalized Additive Models
   - Univariate Local Smoother
   - Multiple Covariates
   - Projection Pursuit Regression
2. Introduction to tree methods
3. A Practical Guide to Tree Construction
4. A Review of Tree Methods
5. Introduction to MARS

# An Introduction to MARS

- MARS model can be reorganized to have the form of

$$\beta_0 + \sum \beta_{ij} \left(x_i - \tau_j\right)^* + \sum_{i \neq k} \beta_{ijkl} \left(x_i - \tau_j\right)^* \left(x_k - \tau_l\right)^* + \cdots$$

  where $(x_i - \tau_j)^*$ is either the negatively truncated function $(x_i - \tau_j)^+$ or the positively truncated one $(x_i - \tau_j)^-$.
- In one-dimensional case, model becomes $\beta_0 + \sum_{k=1}^{M} \beta_k \left(x - \tau_k\right)^*$
- It is noteworthy that model above would be equivalent to a regression tree model if the truncated function $(x - \tau_j)^*$ were replaced with an indicator function $I\left(x > \tau_k\right)$. Thus MARS can be viewed as a generalization of CART.

# An example of MARS model

We can see from the figure below that the predictor space of $(x_1, x_2)$ is partitioned into several rectangles. and MARS consists of six connected planes in total.

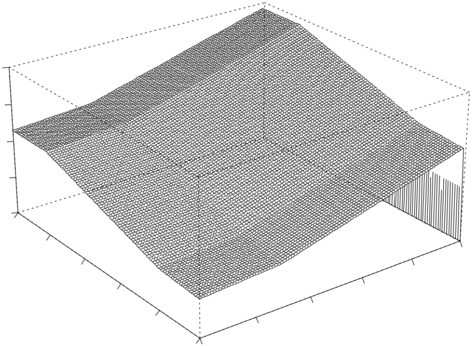


FIGURE 10.7. MARS model: $2.5 + 4(x_1 - 0.3)^+ - (x_1 - 0.3)^- + 4(x_2 - 0.2)^+ - (x_2 - 0.2)^- - 4(x_2 - 0.8)^+$

# Another example of MARS model

We can see from the figure below that the predictor space of $(x_1, x_2)$ is partitioned into several rectangles. and MARS consists of both simple planes and 'twisted' surfaces in total.

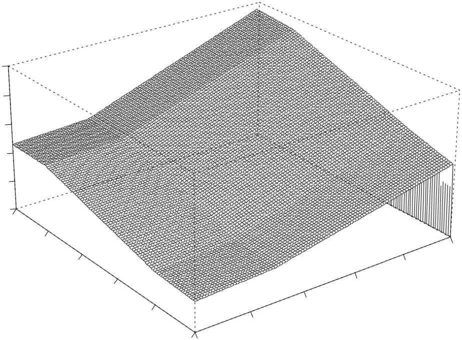


FIGURE 10.8. MARS model: $2.5 + 5(x_1 - 0.3)^+ - (x_1 - 0.3)^- + 4(x_2 - 0.2)^+ - (x_2 - 0.2)^- - 4(x_2 - 0.8)^+ + 2(x_1 - 0.3)^+(x_2 - 0.2)^+ - 5(x_1 - 0.3)^+(x_2 - 0.2)^-$

# Difference between MARS and LR

- In linear model, we decide a priori how many and what terms to be entered into the model. However, we do not know how many terms to include in a MARS prior to data modeling, only a limit of maximum would be assign.
- Every term in a linear model is fully determined, while it is partially specified in the MARS model. In particular, the location of the knot, $\tau_k$, in $(x - \tau_k)^*$ needs to be determined from the data.

# MARS Forward Procedure

- Enter the intercept term, $\beta_0$. Namely, include a constant, 1, as the first basis function.
- Find the combination of predictor $x_i$ and knot $\tau_1$ that gives the best fit to the data when the pair of basis functions: $(x_i - \tau_1)^+$ and $(x_i - \tau_1)^-$ is added to the model.
- If $K$ basis functions have been entered, find the combination of predictor $x_k$, knot $\tau_l$, and an existing term, denoted by $s$, that yields the best fit to the data when $s\,(x_k - \tau_l)^+$ and $s\,(x_k - \tau_l)^-$.
- Repeat step 2 until the maximum number of basis functions have been collected.

# MARS Forward Procedure

- For step 1, to add a pair of basis function $(x_i - \tau_1)^+$ and $(x_i - \tau_1)^-$ that has the constant only, is solving the parameters in $\beta_0 + \beta_1 (x_i - \tau)^+ + \beta_2 (x_i - \tau)^-$, which is equivalent to $\beta_0 + \beta_1 x_i + \beta_2 (x_i - \tau)^+$.
- The process of solving the best node is tedious and for further details, refer to section 10.6, chapter 10.
- After solving for the node, the remaining process is equivalent to solving parameters in linear regression.
- We repeat the above process until K basis function are added to the model.

# MARS Forward Procedure

- In step two, when there is more than one base function in the model, for simplicity, we start with $\beta_0 + \beta_1 x_2 + \beta_2 (x_2 - \tau)^+$, we want to consider multiplicative basic functions of $x_i, (x_i - \tau)^+$, we consider two models as following($x_i$ can either be merged with $x_1$ or $x_1 + \tau$, and we add a term as its 'pair term'.):

$$\beta_0 + \beta_1 x_1 + \beta_2 (x_1 - \tau_1)^+ + \beta_3 x_1 x_i + \beta_4 x_1 (x_i - \tau)^+ ,$$

$$\beta_0 + \beta_1 x_1 + \beta_2 (x_1 - \tau_1)^+ + \beta_3 (x_1 - \tau_1)^+ x_i + \beta_4 (x_1 - \tau_1)^+ (x_i - \tau)^+$$

# An Example of MARS Forward Procedure

| Step | Fitted Model |
|---|---|
| 0 | $-0.71$ |
| 1 | $0.68\underline{-2.18x_2 - 7.1(x_2 - 0.72)^+}$ |
| 2(a) | $-0.41 - 2.18x_2 - 7.92(x_2 - 0.72)^+\underline{+1.28x_1 + 3.59(x_1 - 0.55)^+}$ |
| 2(b) | $-0.4 - 3.37x_2 - 8.21(x_2 - 0.72)^+ + 1.36x_1 + 3.05(x_1 - 0.55)^+$<br>$\underline{+2.65x_2x_3 - 30.4x_2(x_3 - 0.94)^+}$ |
| 2(c) | $-0.39 - 3.17x_2 - 8.24(x_2 - 0.72)^+ + 1.32x_1 + 3.09(x_1 - 0.55)^+$<br>$+2.56x_2x_3 - 37x_2(x_3 - 0.94)^+\underline{-0.4x_2x_4 - 0.81x_2(x_4 - 0.84)^+}$ |
| $\vdots$ | $\vdots$ |

# MARS Backward-Deletion Step

We stop the forward process when the iteration times reach the maximum, which we choose from experience.

- Begin with the MARS model that contains all, say M, basis functions generated from the forward algorithm.
- Delete the existing nonconstant basis function that makes the least contribution to the model according to the least squares criterion.
- Repeat step 1 until only the constant basis remains in the model.

# MARS Backward-Deletion Step

Suppose that we start with a five-basis-function model

$$f_1(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 (x_1 - \tau_1)^+ + \beta_3 (x_1 - \tau_1)^+ x_2 + \beta_4 (x_1 - \tau_1)^+ (x_2 - \tau)^+$$

in step 0. One of the following four nonconstant basis functions, $x_1, (x_1 - \tau_1)^+, (x_1 - \tau_1)^+ x_2$ and $(x_1 - \tau_1)^+ (x_2 - \tau)^+$ can be removed in step1. If we remove $x_1$, the new model is

$$\beta_0 + \beta_1 (x_1 - \tau_1)^+ + \beta_2 (x_1 - \tau_1)^+ x_2 + \beta_3 (x_1 - \tau_1)^+ (x_2 - \tau)^+ .$$

We proceed the above procedure until all terms are removed.

# Criterion for Model Selection

Silverman[6], Friedman [8] and Zhang used a modified version of the generalized cross-validation criterion originally proposed by Craven and Wahba [10]:

$$GCV(k) = \frac{\sum_{i=1}^{N}\left(Y_i - \hat{f}_k\left(\mathbf{x}_i\right)\right)^2}{N[1-(C(k)/N)]^2}$$

For all candidates produced in MARS backward-deletion step, we select the one that minimize the GCV statistic.
The reason for using GCV is that it adjusts the training RSS to take into account the flexibility of the model. We penalize flexibility because models that are too flexible will model the specific realization of noise in the data instead of just the systematic structure of the data.

# Reference I

[1] Uri Alon et al. “Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays”. In: *Proceedings of the National Academy of Sciences* 96.12 (1999), pp. 6745–6750.

[2] Lalit R Bahl et al. “A tree-based statistical language model for natural language speech recognition”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37.7 (1989), pp. 1001–1008.

[3] Susan E Brennan. “Centering attention in discourse”. In: *Language and Cognitive processes* 10.2 (1995), pp. 137–167.

[4] Ju Chen, Steven W Kubalak, and Kenneth R Chien. “Ventricular muscle-restricted targeting of the RXRalpha gene reveals a non-cell-autonomous requirement in cardiac chamber morphogenesis”. In: *Development* 125.10 (1998), pp. 1943–1949.

# Reference II

[5] Ginige L De Silva and Jonathan J Hull. “Proper noun detection in document images”. In: *Pattern Recognition* 27.2 (1994), pp. 311–320.

[6] Peter J Diggle and Michael F Hutchinson. “On spline smoothing with autocorrelated errors”. In: *Australian Journal of Statistics* 31.1 (1989), pp. 166–182.

[7] Rupa Duttagupta and Paul Cashin. “The anatomy of banking crises”. In: *IMF Working Papers* 2008.093 (2008).

[8] Jerome H Friedman. “Multivariate adaptive regression splines”. In: *The annals of statistics* (1991), pp. 1–67.

# Reference III

[9] Patricia S Goldman-Rakic. “The prefrontal landscape: implications of functional architecture for understanding human mentation and the central executive”. In: *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 351.1346 (1996), pp. 1445–1453.

[10] Gene H Golub, Michael Heath, and Grace Wahba. “Generalized cross-validation as a method for choosing a good ridge parameter”. In: *Technometrics* 21.2 (1979), pp. 215–223.

[11] S Gugercin, AC Antoulas, and HP Zhang. “Analysis of the issue of consistency in identification for robust control”. In: (2001).

[12] Jiawei Han et al. “Emerging scientific applications in data mining”. In: *Communications of the ACM* 45.8 (2002), pp. 54–58.

# Reference IV

[13] Nissan Levin, Jacob Zahavi, and Morris Olitsky. “AMOS—A probability-driven, customer-oriented decision support system for target marketing of solo mailings”. In: *European Journal of Operational Research* 87.3 (1995), pp. 708–721.

[14] Wei-Yin Loh. “Fifty years of classification and regression trees”. In: *International Statistical Review* 82.3 (2014), pp. 329–348.

[15] Alicja Wieczorkowska. “Rough sets as a tool for audio signal classification”. In: *International Symposium on Methodologies for Intelligent Systems*. Springer. 1999, pp. 367–375.

[16] Heping Zhang and Burton H Singer. *Recursive partitioning and applications*. Springer Science & Business Media, 2010.