**Decision Tree Induction**

(most popular prediction method in DM)

Continually reinvented idea ~ 30 years

**Others:**
- principal components → Latent Semantic Indexing
- KL expansion
- Ex proj pursuit → ICA, BSS
- PRR → neural networks

**Statistics:**

AID → THAID → CHAID → CART

Morgen, Sonquist, Messenger
(lead rep. in Statistics)
Social Science data

**Machine Learning:** (took seriously)

ID3 → C4.5 → C5.0

no good

like CART

Quinlan
EE - pattern recognition (in early 70's)

References:

Breiman, Friedman, Olshen & Stone (1983)
"Classification and Regression Trees"
chapman & Hall

Quinlan (1993)
"C4.5: Programs for Machine Learning"
Morgan Kaufmann

Main competitors today:
CART, CHAID, C4.5/5.0

We will study CART
- still state of the art
- point out differences as we go along
**Regression Trees**
\( y \in \mathbb{R} \)

**Structural model:** \( \mathcal{F} = \text{class of functions} \)

\[
F(x) = \sum_{m=1}^{M} c_m I(x \in R_m)
\]

\( \mathcal{R}_M \) = subregions of input variable space

\( S^I \) = set of all possible joint values

\( \mathcal{R}_m \subseteq S^I \)

**Examples:**

\( x_3 \leq 14.2 \) & \( 1.7 \leq x_{253} \leq 644 \) & \( \text{color} = \text{red} \)

\( 1.3 x_4 + 7.2 x_{11} - 15 \leq 8.2 \)

\( x \in \mathcal{R}_m \Rightarrow \text{restriction on its joint values} \)
Score criterion - least squares

\[
\{ \hat{c}_m, \hat{R}_m \}_i^M = \arg \min_{\{c_m, R_m\}_i} \sum_{i=1}^N \left[ y_i - \sum_{m=1}^M c_m I(x_i \in R_m) \right]^2
\]

\[\hat{F}(x) = \sum_{m=1}^M \hat{c}_m I(x \in \hat{R}_m) \text{ function est.}\]

Search strategy:

Given \( \{ R_m \}_i^M \), \( \{ \hat{c}_m \}_i^M \) = linear least-squares fit of \( y_i \) on \( \{ I(x_i \in R_m) \}_i^M \)

But optimization WRT \( \{ R_m \}_i^M \) very difficult!

Feasible computation \( \Rightarrow \) restrictions on \( \{ R_m \}_i^M \)
Restrictions:

1. \( \{R_m\}_{m=1}^M \) disjoint

   \[ \Rightarrow R_m \cap R_{m'} = \emptyset, \ m \neq m' \]

   want to predict for all \( x \in S \)

   \[ \Rightarrow \{R_m\}_{m=1}^M \text{ cover input space} \]

   \( S' \subseteq \bigcup_{m=1}^M R_m \Rightarrow \text{partition of } S' \)

2. \[ F(x) = \sum_{m=1}^M c_m I(x \in R_m) \]

   if \( x \in R_m \) then \( F(x) = c_m \)

   piecewise constant approx

Given \( \{R_m\}_{m=1}^M \) \[ c_m = \overline{y}_m \]

\[ \overline{y}_m = \text{mean } \{ y_i | x_i \in R_m \} \]

\[ = \frac{1}{N_m} \sum_{x_i \in R_m} y_i \]

\( N_m = \# \{ x_i \in R_m \} \)
Problem still to difficult

(2) "Simple" regions

$S^l_j = \text{set of all possible values of } x_j$

$S^l = S^l_1 \times S^l_2 \times \cdots \times S^l_m$ (outer product)

$S^l_j$ depends on type of variable

$A_j \subseteq S^l_j = \text{subset of values for } x_j$

$R = \bigcap_{j=1}^{m} A_j \Rightarrow I(x \in R) = \prod_{j=1}^{m} I(x_j \in A_j)$
Structural model:

$$F(x) = \sum_{m=1}^{M} c_m \prod_{j=1}^{m} I(x_j \in A_{jm})$$

subset of values of $x_j$
in $m$th region

if $$\bigcap_{j=1}^{m} (x_j \in A_{jm}) \Rightarrow F(x) = c_m$$

conjunctive "rule" in ML

Note: $A_{jm} = S_j' \Rightarrow$ rule true for all values of $x_j$

$$\prod_{j}^{m} I(x_j \in A_{jm}) = I(x \in A_{jm})$$

$A_{jm} \notin S_j'$

$$\{ j \mid A_{jm} \subset S_j' \} = \text{Vars that define}$
$$\text{mth rule or region}$$

Also note: constraint

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

regions must cover $S'$

$$\sum_{m=1}^{M} \prod_{j=1}^{m} I(x_j \in A_{jm}) = 1$$

limits $\{ A_{jm} \}_{j=1}^{m}$ $m=1$
Def. of $S_j$ depends on type of $X_j$

(1) $X_j =$ numeric (orderable)

$I (X_j \in S_j) = I (a_j \leq X_j < b_j)$

$a_j \in \mathbb{R}_j$ & $b_j \in \mathbb{R}_j$ real numbers

Note: $a_j = -\infty$ , $b_j = \infty$
possible values

(2) $X_j =$ categorical (nominal - unordered)

$S_j$ explicitly delineated

Examples:

$X_j =$ occupation

$S_j =$ \{manager, student, retired\}

$X_j =$ gender

$S_j =$ \{female\}
FIGURE 9.2. Partitions and CART. Top right panel shows a partition of a two-dimensional feature space by recursive binary splitting, as used in CART, applied to some fake data. Top left panel shows a general partition that cannot be obtained from recursive binary splitting. Bottom left panel shows the tree corresponding to the partition in the top right panel, and a perspective plot of the prediction surface appears in the bottom right panel.
Simplified problem:

\[ \hat{\mathcal{S}}_{jm} \overset{\mathcal{J}_i}{\underset{j=1}{\overset{M}{\cap}}} \mathcal{M} = \arg \min_{\hat{\mathcal{S}}_{jm} \overset{\mathcal{J}_i}{\underset{j=1}{\overset{M}{\cap}}} \mathcal{M}} \sum_{i=1}^{N} \left[ y_{i} - \sum_{m=1}^{M} \tilde{y}_{m} \prod_{j=1}^{M} I(x_{ij} \in \hat{\mathcal{S}}_{jm}) \right]^{2} \]

\[ \hat{F}(x) = \sum_{m=1}^{M} \tilde{y}_{m} \prod_{j=1}^{M} I(x_{ij} \in \hat{\mathcal{S}}_{jm}) \]

depends on \[ \hat{\mathcal{S}}_{jm} \overset{\mathcal{J}_i}{\underset{j=1}{\overset{M}{\cap}}} \mathcal{M} \]

\[ \hat{F}(x) \]

Still too difficult to solve exactly

(except for \( M \leq 6 \): Bar Jon Jost)

Approximation: Recursive partitioning

Search strategy = greedy steepest descent
**Iterative algorithm**

**Initializes**: \( R_1 = S; \quad F_1(x) = \overline{y} \)

\[ F_M(x) = \sum_{m=1}^{M} \overline{y}_m I(x \in R_m) \]

**At m-th iteration**: 

choose one of the regions \( m^* \)

partition \( R_{m^*} \rightarrow R_{m^*}^{(1)} \cup R_{m^*}^{(2)} \)

\( R_{m^*}^{(1)} \cap R_{m^*}^{(2)} = \emptyset \) empty set

\( R_{m^*} = \text{"parent" region} \)

\( R_{m^*}^{(1)} = \text{"left daughter" region} \)

\( R_{m^*}^{(2)} = \text{"right daughter" region} \)
Replace $R_{m*} \leftarrow R_{m*}^{(c1)}$

Add $R_{M+1} \leftarrow R_{m*}^{(c2)}$

$$F_{M+1} (x) = \sum_{m=1}^{M+1} \gamma_m I(x \in R_m)$$

$M \leftarrow M + 1$ for next iteration

Questions:

(a) Which region to split ($m*$)

(b) How to split it

$$R_{m*} \rightarrow R_{m*}^{(c1)} \cup R_{m*}^{(c2)}$$
(a) Which region? 

\[
m^* = \arg \min_{1 \leq m \leq M} \sum_{i=1}^{N} \left[ y_i - \mu^{(c)}_m I(x_i \in R^{(c)}_m) - \frac{c}{m} I(x_i \in R^{(c)}_m) \right]^2
\]

\[
\sum_{i=1}^{N} = \sum_{m=1}^{M} \sum_{x_i \in R^{(c)}_m} (\text{disjoint})
\]

\[
m^* = \arg \min_{1 \leq m \leq M} \sum_{i=1}^{N} \left[ y_i - \mu^{(c)}_m I(x_i \in R^{(c)}_m) - \frac{c}{m} I(x_i \in R^{(c)}_m) \right]^2
\]

= region that most improves fit

= region most improved by split (need only consider \( x_i \in R^{(c)}_m \))

\[
m^* = \arg \max_{1 \leq m \leq M} \frac{N^{(c)}_m N^{(c)}_m}{N_m} \left( \frac{c^{(c)}_m - \mu^{(c)}_m}{c^{(c)}_m - \mu^{(c)}_m} \right)^2
\]
(b) How to split?

\[ I(X \in R_m) = \prod_{j=1}^{m} I(X_j \in Q_{jm}) \]

\[ X_j = \text{one of the predictor var's} \]

\[ t_{jm} \subset Q_{jm}, \quad \bar{t}_{jm} = Q_{jm} - t_{jm} \]

\[ \text{Eligible splits:} \]

\[ I(X \in R^{(k)}_m) = I(X_j \in t_{jm}) \prod_{j' \neq j} I(X_{j'} \in \bar{Q}_{jm}) \]

\[ I(X \in R^{(k)}_m) = I(X_j \in \bar{t}_{jm}) \prod_{j' \neq j} I(X_{j'} \in Q_{jm}) \]

\[ I(X_j \in Q_{jm}) \quad \quad X_j \]

\[ I(X_j \in t_{jm}) \quad I(X_j \in \bar{t}_{jm}) \quad 1 - I(X_j \in t_{jm}) \]

\[ \text{can always split again} \]
Jointly choose $j^*$ & $t^*_m$ to give max improvement to fit (for each $m$)

therefore, entire problem becomes:

$$(m^*, j^*, t^*_m) = \arg \min_{1 \leq m \leq M} \min_{1 \leq j \leq m} \sum_{x_i \in R_m} \left\{ \left[ y_i - \bar{y}_e I(x_{ij} \in t^*_m) - \bar{y}_r I(x_{ij} \in \bar{E}_m) \right]^2 \\
- \left[ y_i - \bar{y}_m \right]^2 \right\}$$

$$\bar{y}_e = \sum_{x_i \in R_m} y_i I(x_{ij} \in t^*_m) / N_e$$

$$N_e = \sum_{x_i \in R_m} I(x_{ij} \in t^*_m)$$

$$\bar{y}_r = (N_m \bar{y}_m - N_e \bar{y}_e) / (N_m - N_e)$$

$$\bar{y}_m = \sum_{x_i \in R_m} y_i / N_m$$

$$N_m = \sum_{x_i \in R_m} 1 \# of obs. \in R_m$$

feasible problem!
Characterization of "split" depends on type of variable $x_j$

(1) $x_j$ = numeric (orderable)

$I(x_j \in t_{jm}) = I(x_j \leq t), t \in \text{split point} \subseteq \text{singular number}$

if $\Delta_{jm} =$ finite # of values

then opt. over $t_{jm} =$ opt. on a single number $\Rightarrow$ exhaustive search.

sorting + updating formulae for $\bar{y}_e$ & $\text{mse} \Rightarrow O(\# \Delta_{jm})$

what about $x_j =$ interval - scale

$\Rightarrow$ $\infty \text{ to } \#(\Delta_{jm})$
Trick: need only consider 

\[ t \in \{ x_{ij} \mid x \in \mathbb{R}^m \} \]

since criterion only depends on which obs. have \( x_{ij} \leq t \), not on explicit value of \( t \).

\[ t \uparrow \text{ not identifiable} \]

(same value for criterion)

Choose midpoint (arbitrary)

Do need to consider at most \( \#(R_m) \) candidate splits \( \Rightarrow \) exhaustive search

\( \sim O(\#R_m) \)

(2) \( x_{ij} = \text{categorical (nominal)} \)

no order relation

\( \#(t_{jm}) = 2^{\#(S_{jm})} - 1 - 1 \)

exponential in \( \#(S_{jm}) \)
therefore, exhaustive search \( \Omega(2^{\#(Q_{jm})}) \)

problem for \( \#(Q_{jm}) \geq 10 \cdot 12 \)

Happens often in Data Mining.

**Trick (approximation):**

\[ \tilde{r}_{ej} = \text{mean} ( y_i \mid x_j = l \text{ th value } \in Q_{jm} ) \]

consider \( x_j \) as ordinal

with order relation defined by \( \tilde{r}_{ej} \)

\[ \Rightarrow \text{comp } \approx O(N_m) \]

\[ \leq \#(R_m) \]

not exponential \( \sim 2^{\#(Q_{jm})} \)
Recursive Partitioning Algorithm
(search strategy)

\( R_1 = \mathcal{X} \) entire input space

\begin{algorithm}
\begin{algorithmic}
\While {\( M \leq M_{\text{max}} \)}
\State \((m^*, j^*, t_{jm}^*) = \arg \min_{1 \leq m \leq M, 1 \leq j \leq n} \sum_{x \in R_m} \left[ y_i - \overline{g}_m I(x_j \in t_{jm}) - \overline{g}_m I(x_j \notin t_{jm}) \right]^2 - \left[ y_i - \overline{g}_m \right]^2 \)
\State \( I(x \in R_{m^*}) \leq I(x \in R_{m^*}) I(x_j \in t_{jm}^*) I(x_j \notin t_{jm}^*) \)
\State \( I(x \in R_{m+1}) = I(x \in R_{m^*}) I(x_j \in t_{jm}^*) I(x_j \notin t_{jm}^*) \)
\State \( M \leftarrow M + 1 \)
\EndWhile
\end{algorithmic}
\end{algorithm}
Notes:

$R_5 - R_{10}$ not optimal/partitioning for $M = 6$.

Optimization performed over a very restricted subset of all possible $M = 6$ partitionings. (nested hierarchical)

Trade-off is computational feasibility. (search strategy)
Connection between Recursive Partitioning and Binary Trees
Why Trees?

Internal nodes represent splits
Terminal nodes represent final regions defining model

\[ l_i = \bar{y}_i = \bar{y}_c \]
**Decision Tree**

\[
\hat{F}(x) = \sum_{m=1}^{\mathcal{E}} y_m \mathbb{I}(x \in R_m)
\]

**Why so popular:**

1. **visualize n-dim model (2-dim graphic)**
2. **see what vars are important**
3. **traverse tree to evaluate \( \hat{F}(x) \)**
   - (find out which \( x \in R_m \))
   - if \( x \in R_m \) \( \Rightarrow \hat{F}(x) = y_m \)
   - \( \text{comp} \propto \log_2(\mathcal{M}) \), indep of \( n \) & \( N \)
**Missing Predictor Values:**

Often there are missing values among the predictor variables.

Sometime (for large $p$) there are no complete obs.

**Questions:**

(a) given the tree, how to classify test obs with missing predictor values

(b) how to build the tree in the presence of missing data
Suppose \( \mathbf{x} = (x_1, x_2, \ldots, x_{k-1}, \ldots, x_n) \) 

\[ \text{missing value} \]

Split of \( R_m \): 

if \( I(x_{j(m)} \in S_m) = 1 \) go left 
= 0 go right 

if \( l \neq j(m) \) no problem 
what if \( l = j(m) = \text{missing} \)

Training: for each \( x_j \) use non missing values 

\[ \{ y_i, x_{ij} | x_i \in R_m \text{ & } x_{ij} \neq \text{missing} \} \]

to estimate \( S_m \).

What to do with training obs. \( \in R_m \) missing \( x_{j(m)} \)?

Prediction: given \( \mathbf{x} \), traverse tree to terminal node 

what if at node \( m \), \( x_{j(m)} = \text{missing} \)?
Both problems solved by

**Sumogate splits**

**Training:**

optimal split of \( R_m = (j(m), \delta_m) \)

split variable \( j(m) \)

subset of values \( \delta_m \)

**Trick:** use other vars. to predict split.

consider \( \tilde{S}_j = \{ \tilde{y}_i, X_i \mid X_i(j(m) \neq \text{missing}) \} \)

where \( \tilde{y}_i = I(X_i(j(m) \in \delta_m)\)

\( \uparrow \) "sumogate" response

\[ \begin{align*}
X_{j(m)} & \quad \delta_m \\
\tilde{y} & \quad 1 \\
0 & \quad \cdots \cdots \\
\end{align*} \]
$$c_{m}^{l} = \max \left\{ \frac{\text{corr} \left[ Y, I(x_{i} \in s) \right]}{\lambda} \right\}$$

$$\lambda_{m} = \text{maximizing value}$$

order $\{ k \neq j(m) \}$ on decreasing $c_{k}$

Note: $c_{j(m)} = 1 = \max \text{ over all vars}$

let $j(l, m) = l \text{ th largest at node m}$

with $j(1, m) = j(m)$

$$\sum_{k=2}^{m} j(l, m) = \text{ordered surrogate splits for } (j(m), o_{m})$$

$$\sum_{k=2}^{m} o_{k} = \text{surrogate split subsets}$$
Augmented splitting rule at each mode $m$ (non terminal)

Store $L$ best surrogates $\{j_{l}(e,m), s_{l}(e,m)\}_{l=1}^{L}$

$l = 1$

Loop
  
  if $x_{j_{l}(e,m)} \neq \text{missing}$ then exit loop

  $l \leftarrow l + 1$; if $l > L$ then exit loop

end loop

if $l \leq L$ then use $(j_{l}(e,m), s_{l}(e,m))$
  to define split

else send left / right with respective probabilities $N_{L} / N_{R}$

$\{j_{l}(e,m), s_{l}(e,m)\}_{l=2}^{L} = \text{“surrogate” splits at mode } m$

$(\hat{j}(1,m), \hat{s}(1,m)) = (j(m), s(m))$

= “primary” split
Basic motivation:

If $X_{j(m)}$ is missing, use non missing split locally most highly correlated with it to define split.

Exploit linear & nonlinear relationships among the input variables

When (usually) there is a high degree of association among the predictors missing values cause little loss of information $\Rightarrow$ performance

Redundant var\'s can be helpful