Decision & Regression Trees

CSci 5525: Machine Learning

Instructor: Arindam Banerjee

October 7, 2013

Attribute-based representations

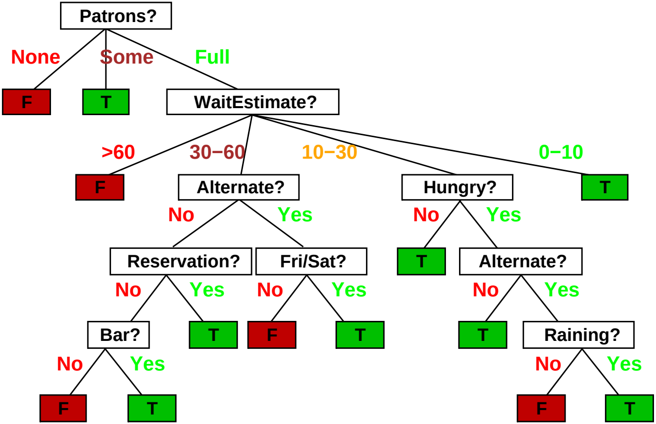
Examples described by attribute values (Boolean, discrete, continuous, etc.)

Example	*Attributes*										*Target*
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	WillWait
X_1	T	F	F	T	Some	\$\$\$	F	T	French	0–10	T
X_2	T	F	F	T	Full	\$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	\$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	\$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
X_6	F	T	F	T	Some	\$\$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	\$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	\$\$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	\$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	\$\$\$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	\$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	\$	F	F	Burger	30–60	T

Classification of examples is positive (T) or negative (F)

Decision Tree Example (Restaurant)

One possible representation for hypotheses

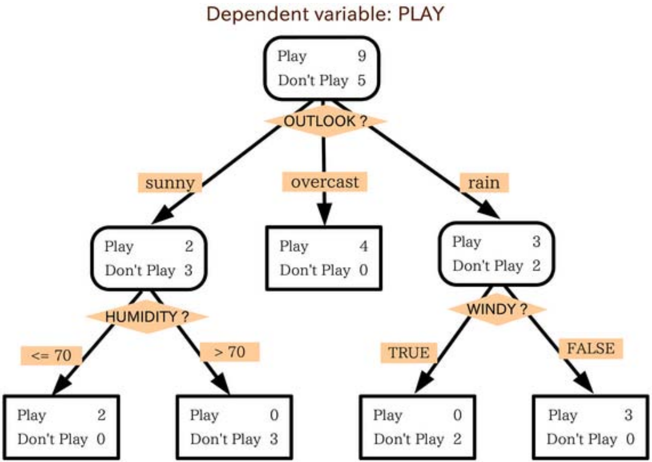


Example: Playing Golf

Play golf dataset

Independent variables				Dep. var
OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY
sunny	85	85	FALSE	Don't Play
sunny	80	90	TRUE	Don't Play
overcast	83	78	FALSE	Play
rain	70	96	FALSE	Play
rain	68	80	FALSE	Play
rain	65	70	TRUE	Don't Play
overcast	64	65	TRUE	Play
sunny	72	95	FALSE	Don't Play
sunny	69	70	FALSE	Play
rain	75	80	FALSE	Play
sunny	75	70	TRUE	Play
overcast	72	90	TRUE	Play
overcast	81	75	FALSE	Play
rain	71	80	TRUE	Don't Play

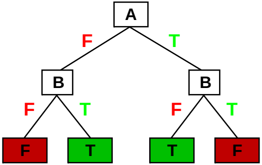
Decision Tree Example (Golf)



Expressiveness

- Decision trees can express any function of the input attributes

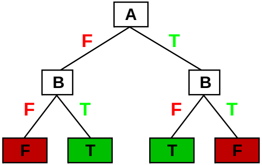
A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



Expressiveness

- Decision trees can express any function of the input attributes
  - for Boolean functions, truth table row → path to leaf

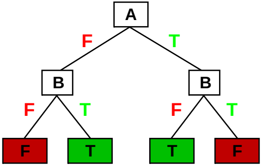
A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



Expressiveness

- Decision trees can express any function of the input attributes
  - for Boolean functions, truth table row $\rightarrow$ path to leaf

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

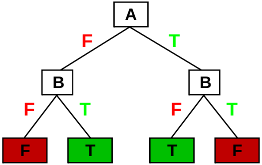


- The basic trade-off

Expressiveness

- Decision trees can express any function of the input attributes
  - for Boolean functions, truth table row $\rightarrow$ path to leaf

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

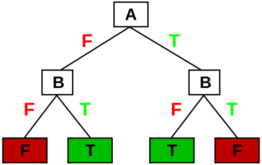


- The basic trade-off
  - There is a consistent decision tree for any training set

Expressiveness

- Decision trees can express any function of the input attributes
  - for Boolean functions, truth table row $\rightarrow$ path to leaf

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

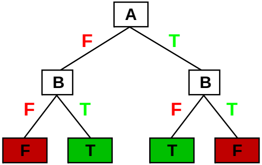


- The basic trade-off
  - There is a consistent decision tree for any training set
    - Unless f is nondeterministic in x

Expressiveness

- Decision trees can express any function of the input attributes
  - for Boolean functions, truth table row → path to leaf

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

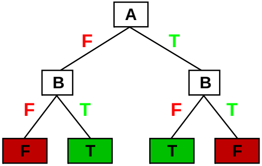


- The basic trade-off
  - There is a consistent decision tree for any training set
    - Unless f is nondeterministic in x
  - One path to leaf for each example

Expressiveness

- Decision trees can express any function of the input attributes
  - for Boolean functions, truth table row $\rightarrow$ path to leaf

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

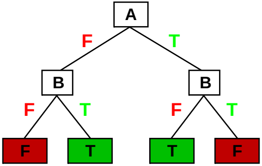


- The basic trade-off
  - There is a consistent decision tree for any training set
    - Unless f is nondeterministic in x
  - One path to leaf for each example
  - But it probably won't generalize to new examples

Expressiveness

- Decision trees can express any function of the input attributes
  - for Boolean functions, truth table row $\rightarrow$ path to leaf

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



- The basic trade-off
  - There is a consistent decision tree for any training set
    - Unless f is nondeterministic in x
  - One path to leaf for each example
  - But it probably won't generalize to new examples
- Prefer to find more compact decision trees

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes
  - Number of boolean functions

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes
  - Number of boolean functions
  - Number of distinct truth tables with 2^n rows $= 2^{2^n}$

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes
  - Number of boolean functions
  - Number of distinct truth tables with 2^n rows $= 2^{2^n}$
  - 6 Boolean attributes give 18,446,744,073,709,551,616 trees

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes
  - Number of boolean functions
  - Number of distinct truth tables with 2^n rows $= 2^{2^n}$
  - 6 Boolean attributes give 18,446,744,073,709,551,616 trees
- How many purely conjunctive hypotheses ($Hungry \cap \neg Rain$)

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes
  - Number of boolean functions
  - Number of distinct truth tables with 2^n rows $= 2^{2^n}$
  - 6 Boolean attributes give 18,446,744,073,709,551,616 trees
- How many purely conjunctive hypotheses $(Hungry \cap \neg Rain)$
  - Each attribute can be in (positive), in (negative), or out $\implies$ 3^n distinct conjunctive hypotheses

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes
  - Number of boolean functions
  - Number of distinct truth tables with 2^n rows $= 2^{2^n}$
  - 6 Boolean attributes give 18,446,744,073,709,551,616 trees
- How many purely conjunctive hypotheses ($Hungry \cap \neg Rain$)
  - Each attribute can be in (positive), in (negative), or out
    $\implies 3^n$ distinct conjunctive hypotheses
- More expressive hypothesis space

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes
  - Number of boolean functions
  - Number of distinct truth tables with 2^n rows $= 2^{2^n}$
  - 6 Boolean attributes give 18,446,744,073,709,551,616 trees
- How many purely conjunctive hypotheses ($Hungry \cap \neg Rain$)
  - Each attribute can be in (positive), in (negative), or out $\implies 3^n$ distinct conjunctive hypotheses
- More expressive hypothesis space
  - Increases chance that target function can be expressed

Hypothesis Spaces

- How many distinct decision trees with n Boolean attributes
  - Number of boolean functions
  - Number of distinct truth tables with 2^n rows $= 2^{2^n}$
  - 6 Boolean attributes give 18,446,744,073,709,551,616 trees
- How many purely conjunctive hypotheses $(Hungry \cap \neg Rain)$
  - Each attribute can be in (positive), in (negative), or out
    $\Longrightarrow$ 3^n distinct conjunctive hypotheses
- More expressive hypothesis space
  - Increases chance that target function can be expressed
  - Increases number of hypotheses consistent w/ training set
    $\Longrightarrow$ may get worse predictions

Decision Tree Learning

Aim: Find a small tree consistent with the training examples

Decision Tree Learning

Aim: Find a small tree consistent with the training examples

Recursively choose "most significant" attribute as root of (sub)tree

Decision Tree Learning

Aim: Find a small tree consistent with the training examples

Recursively choose "most significant" attribute as root of (sub)tree

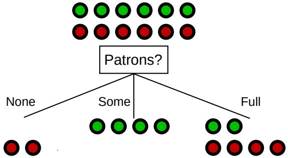
Good attribute splits the examples (ideally) into "pure" subsets

Decision Tree Learning

Aim: Find a small tree consistent with the training examples

Recursively choose "most significant" attribute as root of (sub)tree

Good attribute splits the examples (ideally) into "pure" subsets

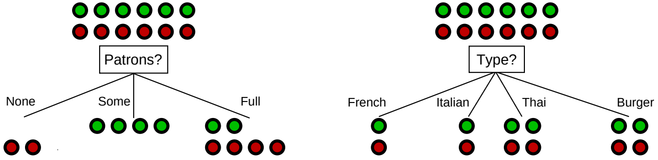


Decision Tree Learning

Aim: Find a small tree consistent with the training examples

Recursively choose "most significant" attribute as root of (sub)tree

Good attribute splits the examples (ideally) into "pure" subsets



Patrons? is a better choice

—gives *information* about the classification

Information Gain

- The entropy of a r.v. X with distribution $(p(x_1), \ldots, p(x_n))$

$$H(X) = \sum_{i=1}^{n} -p(x_i) \log_2 p(x_i)$$

Information Gain

- The entropy of a r.v. X with distribution $(p(x_1), \ldots, p(x_n))$

$$H(X) = \sum_{i=1}^{n} -p(x_i) \log_2 p(x_i)$$

- The conditional entropy

$$H(X|Y) = \sum_{j=1}^{m} p(y_j) H(X|y_j)$$

Information Gain

- The entropy of a r.v. X with distribution $(p(x_1), \ldots, p(x_n))$

$$H(X) = \sum_{i=1}^{n} -p(x_i) \log_2 p(x_i)$$

- The conditional entropy

$$H(X|Y) = \sum_{j=1}^{m} p(y_j) H(X|y_j)$$

- Information Gain

$$IG(X|Y) = H(X) - H(X|Y)$$

Information Gain

- The entropy of a r.v. X with distribution $(p(x_1), \ldots, p(x_n))$

$$H(X) = \sum_{i=1}^{n} -p(x_i) \log_2 p(x_i)$$

- The conditional entropy

$$H(X|Y) = \sum_{j=1}^{m} p(y_j) H(X|y_j)$$

- Information Gain

$$IG(X|Y) = H(X) - H(X|Y)$$

  - How informative is Y regarding X

Information Gain

- The entropy of a r.v. X with distribution $(p(x_1), \ldots, p(x_n))$

$$H(X) = \sum_{i=1}^{n} -p(x_i) \log_2 p(x_i)$$

- The conditional entropy

$$H(X|Y) = \sum_{j=1}^{m} p(y_j) H(X|y_j)$$

- Information Gain

$$IG(X|Y) = H(X) - H(X|Y)$$

  - How informative is Y regarding X
- How informative is an attribute for classification

Information Gain

- The entropy of a r.v. X with distribution $(p(x_1), \ldots, p(x_n))$

$$H(X) = \sum_{i=1}^{n} -p(x_i) \log_2 p(x_i)$$

- The conditional entropy

$$H(X|Y) = \sum_{j=1}^{m} p(y_j) H(X|y_j)$$

- Information Gain

$$IG(X|Y) = H(X) - H(X|Y)$$

  - How informative is Y regarding X
- How informative is an attribute for classification
- Example: p positive and n negative examples at the root

Information Gain

- The entropy of a r.v. X with distribution $(p(x_1), \ldots, p(x_n))$

$$H(X) = \sum_{i=1}^{n} -p(x_i) \log_2 p(x_i)$$

- The conditional entropy

$$H(X|Y) = \sum_{j=1}^{m} p(y_j) H(X|y_j)$$

- Information Gain

$$IG(X|Y) = H(X) - H(X|Y)$$

  - How informative is Y regarding X
- How informative is an attribute for classification
- Example: p positive and n negative examples at the root
  - $H(\langle p/(p+n), n/(p+n)\rangle)$ bits needed to classify

Information Gain

- The entropy of a r.v. X with distribution $(p(x_1), \ldots, p(x_n))$

$$H(X) = \sum_{i=1}^{n} -p(x_i) \log_2 p(x_i)$$

- The conditional entropy

$$H(X|Y) = \sum_{j=1}^{m} p(y_j) H(X|y_j)$$

- Information Gain

$$IG(X|Y) = H(X) - H(X|Y)$$

  - How informative is Y regarding X
- How informative is an attribute for classification
- Example: p positive and n negative examples at the root
  - $H(\langle p/(p+n), n/(p+n) \rangle)$ bits needed to classify
  - For 12 restaurant examples, $p = n = 6$, so we need 1 bit

Information Gain (Contd.)

- An attribute splits the examples E into subsets E_i

Information Gain (Contd.)

- An attribute splits the examples E into subsets E_i
  - Each E_i should need less information to classify

Information Gain (Contd.)

- An attribute splits the examples E into subsets E_i
  - Each E_i should need less information to classify
- Let E_i have p_i positive and n_i negative examples

Information Gain (Contd.)

- An attribute splits the examples E into subsets E_i
  - Each E_i should need less information to classify
- Let E_i have p_i positive and n_i negative examples
  - $H(\langle p_i/(p_i + n_i), n_i/(p_i + n_i)\rangle)$ bits needed to classify

Information Gain (Contd.)

- An attribute splits the examples E into subsets E_i
  - Each E_i should need less information to classify
- Let E_i have p_i positive and n_i negative examples
  - $H(\langle p_i/(p_i + n_i), n_i/(p_i + n_i)\rangle)$ bits needed to classify
  - Expected number of bits per example over all branches is

$$H(X|Y) = \sum_i \frac{p_i + n_i}{p + n} \; H(\langle p_i/(p_i + n_i), n_i/(p_i + n_i)\rangle)$$

Information Gain (Contd.)

- An attribute splits the examples E into subsets E_i
  - Each E_i should need less information to classify
- Let E_i have p_i positive and n_i negative examples
  - $H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i)\rangle)$ bits needed to classify
  - Expected number of bits per example over all branches is

$$H(X|Y) = \sum_i \frac{p_i+n_i}{p+n} \; H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i)\rangle)$$

- For '*Patrons*', this is 0.459 bits, for '*Type*' this is (still) 1 bit

Information Gain (Contd.)

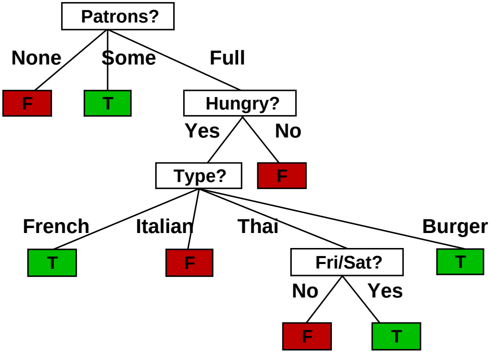
- An attribute splits the examples E into subsets E_i
  - Each E_i should need less information to classify
- Let E_i have p_i positive and n_i negative examples
  - $H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i)\rangle)$ bits needed to classify
  - Expected number of bits per example over all branches is

$$H(X|Y) = \sum_i \frac{p_i+n_i}{p+n} H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i)\rangle)$$

- For '*Patrons*', this is 0.459 bits, for '*Type*' this is (still) 1 bit
- Choose the attribute that maximizes information gain

A Simple Decision Tree

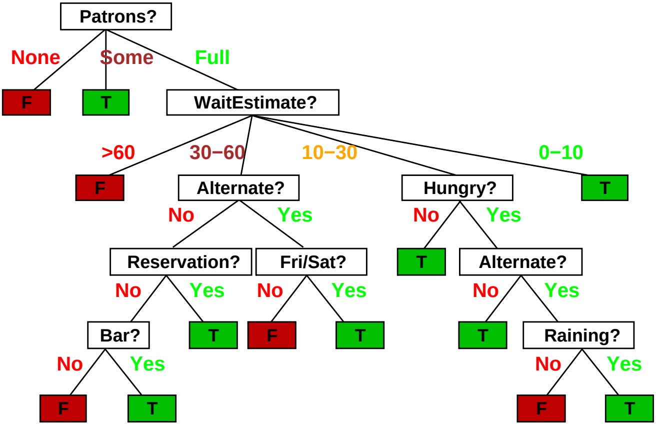
- Decision tree learned from the 12 examples:



- Substantially simpler than “true” tree
  - More complex hypothesis is not justified by a small dataset

The Original Tree

The original (complex) tree with same performance on the training set



Another Example: Edibility

Skin	Color	Thorny?	Flowering?	Edible?
smooth	pink	no	yes	yes
smooth	pink	no	no	yes
scaly	pink	no	yes	no
rough	purple	no	yes	no
rough	orange	yes	yes	no
scaly	orange	yes	no	no
smooth	purple	no	yes	yes
smooth	orange	yes	yes	no
rough	purple	yes	yes	no
smooth	purple	yes	no	no
scaly	purple	no	no	no
scaly	pink	yes	yes	no
rough	purple	no	no	yes
rough	orange	yes	no	yes

Edibility Example (Contd.)

- Entropy $H(Edible) = 0.9403$

Edibility Example (Contd.)

- Entropy $H(Edible) = 0.9403$
- Information Gain for each attribute

Edibility Example (Contd.)

- Entropy $H(Edible) = 0.9403$
- Information Gain for each attribute
  - $IG(Edible|Skin) = 0.2467$

Edibility Example (Contd.)

- Entropy $H(Edible) = 0.9403$
- Information Gain for each attribute
  - $IG(Edible|Skin) = 0.2467$
  - $IG(Edible|Color) = 0.0292$

Edibility Example (Contd.)

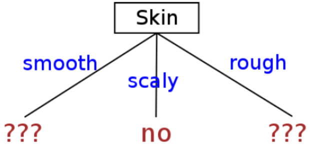
- Entropy $H(Edible) = 0.9403$
- Information Gain for each attribute
    - $IG(Edible|Skin) = 0.2467$
    - $IG(Edible|Color) = 0.0292$
    - $IG(Edible|Thorny) = 0.1519$

Edibility Example (Contd.)

- Entropy $H(Edible) = 0.9403$
- Information Gain for each attribute
  - $IG(Edible|Skin) = 0.2467$
  - $IG(Edible|Color) = 0.0292$
  - $IG(Edible|Thorny) = 0.1519$
  - $IG(Edible|Flowering) = 0.0481$

Edibility Example (Contd.)

- Entropy $H(Edible) = 0.9403$
- Information Gain for each attribute
  - $IG(Edible|Skin) = 0.2467$
  - $IG(Edible|Color) = 0.0292$
  - $IG(Edible|Thorny) = 0.1519$
  - $IG(Edible|Flowering) = 0.0481$



Edibility Example (Contd.)

- Consider $Skin = Smooth$

Edibility Example (Contd.)

- Consider $Skin = Smooth$
  - Entropy $H(Edible|Skin = smooth) = 0.9710$

Edibility Example (Contd.)

- Consider $Skin = Smooth$
  - Entropy $H(Edible|Skin = smooth) = 0.9710$
  - $IG(Edible|Color, Skin = smooth) = 0.4000$

Edibility Example (Contd.)

- Consider $Skin = Smooth$
  - Entropy $H(Edible|Skin = smooth) = 0.9710$
  - $IG(Edible|Color, Skin = smooth) = 0.4000$
  - $IG(Edible|Thorny, Skin = smooth) = 0.9710$

Edibility Example (Contd.)

- Consider $Skin = Smooth$
  - Entropy $H(Edible|Skin = smooth) = 0.9710$
  - $IG(Edible|Color, Skin = smooth) = 0.4000$
  - $IG(Edible|Thorny, Skin = smooth) = 0.9710$
  - $IG(Edible|Flowering, Skin = smooth) = 0.0200$

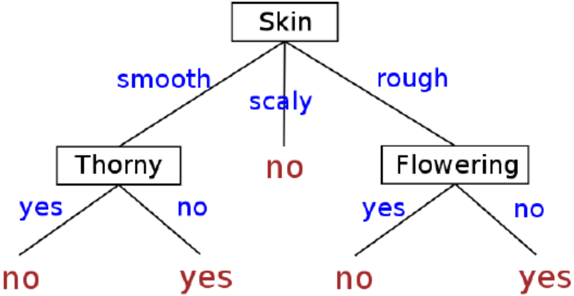
Edibility Example (Contd.)

- Consider $Skin = Smooth$
  - Entropy $H(Edible|Skin = smooth) = 0.9710$
  - $IG(Edible|Color, Skin = smooth) = 0.4000$
  - $IG(Edible|Thorny, Skin = smooth) = 0.9710$
  - $IG(Edible|Flowering, Skin = smooth) = 0.0200$
- Consider $Skin = rough$

Edibility Example (Contd.)

- Consider $Skin = Smooth$
  - Entropy $H(Edible|Skin = smooth) = 0.9710$
  - $IG(Edible|Color, Skin = smooth) = 0.4000$
  - $IG(Edible|Thorny, Skin = smooth) = 0.9710$
  - $IG(Edible|Flowering, Skin = smooth) = 0.0200$
- Consider $Skin = rough$
  - Choose *Flowering*

Edibility Decision Tree



Other Methods for Feature Selection

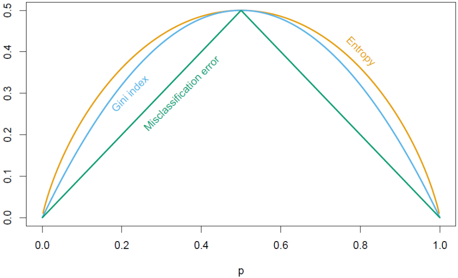
- Issues with Information Gain
- Gain Ratio

$$GainRatio(X|Y) = \frac{IG(X|Y)}{H(Y)}$$

- Gini Index

$$\begin{aligned} Gini(X) &= \sum_{i \neq j} p(i)p(j) \\ Gini(X|Y) &= \sum_{j} p(y_j) Gini(X|y_j) \\ GiniGain(X|Y) &= Gini(X) - Gini(X|Y) \end{aligned}$$

Loss functions for Decision Trees



p = proportion of points in class 2

Entropy has been scaled to pass through $(\frac{1}{2}, \frac{1}{2})$

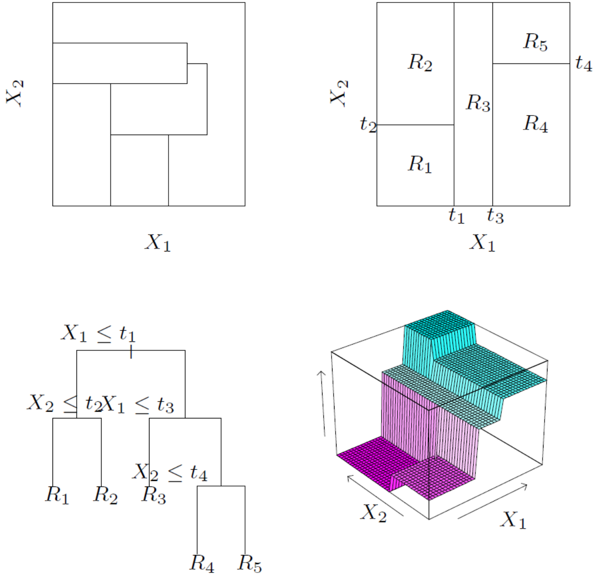
Challenges: Overfitting, Continuous features, Stability

- Overfitting: May generate a large tree
  - Good training set performance, poor test set performance
  - Decision tree pruning: Remove irrelevant attributes
  - Information gain is small after splitting
  - Significance test with null hypothesis of no underlying pattern
- Continuous valued attributes
  - Split into regions, convert to categorical variable
  - Find the split(s) so as to maximize information gain

Temperature	40	48	60	72	80	90
Play Tennis	No	No	Yes	Yes	Yes	No

- Stability
  - Mild change in attributes can change the tree
  - High variance, unstable, but interpretable
  - Ensembles approaches to reduce variance

Regression Trees



Regression Trees

- Assume regions with 'constant' values

Regression Trees

- Assume regions with 'constant' values
- The regression model is given by

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

Regression Trees

- Assume regions with 'constant' values
- The regression model is given by

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

- Using criterion $\sum(y_i - f(\mathbf{x}_i))^2$ we have

$$\hat{c}_m = \text{ave}(y_i | \mathbf{x}_i \in R_m)$$

Regression Trees

- Assume regions with 'constant' values
- The regression model is given by

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

- Using criterion $\sum (y_i - f(\mathbf{x}_i))^2$ we have

$$\hat{c}_m = \text{ave}(y_i | \mathbf{x}_i \in R_m)$$

- Difficult to find general regions (in high-d)

Regression Trees

- Assume regions with 'constant' values
- The regression model is given by

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

- Using criterion $\sum(y_i - f(\mathbf{x}_i))^2$ we have

$$\hat{c}_m = \text{ave}(y_i | \mathbf{x}_i \in R_m)$$

- Difficult to find general regions (in high-d)
- Difficult to even find best binary partition

Regression Trees

- Assume regions with 'constant' values
- The regression model is given by

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

- Using criterion $\sum (y_i - f(\mathbf{x}_i))^2$ we have

$$\hat{c}_m = \text{ave}(y_i | \mathbf{x}_i \in R_m)$$

- Difficult to find general regions (in high-d)
- Difficult to even find best binary partition
- Greedy approach: For each feature j and split point s

$$R_1(j,s) = \{X | X_j \leq s\} \quad R_2(j,s) = \{X | X_j > s\}$$

Regression Trees

- Assume regions with 'constant' values
- The regression model is given by

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

- Using criterion $\sum (y_i - f(\mathbf{x}_i))^2$ we have

$$\hat{c}_m = \text{ave}(y_i | \mathbf{x}_i \in R_m)$$

- Difficult to find general regions (in high-d)
- Difficult to even find best binary partition
- Greedy approach: For each feature j and split point s

$$R_1(j,s) = \{X | X_j \leq s\} \quad R_2(j,s) = \{X | X_j > s\}$$

- The splitting variable and split point is chosen by solving

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

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion
- $T \subset T_0$ is any tree obtained by collapsing internal nodes

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion
- $T \subset T_0$ is any tree obtained by collapsing internal nodes
- The goodness of the tree T

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T|$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} y_i \quad Q_m(T) = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} (y_i - \hat{c}_m)^2$$

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion
- $T \subset T_0$ is any tree obtained by collapsing internal nodes
- The goodness of the tree T

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T|$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} y_i \quad Q_m(T) = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} (y_i - \hat{c}_m)^2$$

- For each α, find a subtree $T_\alpha \subseteq T_0$ to minimize $C_\alpha(T)$

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion
- $T \subset T_0$ is any tree obtained by collapsing internal nodes
- The goodness of the tree T

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T|$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} y_i \quad Q_m(T) = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} (y_i - \hat{c}_m)^2$$

- For each α, find a subtree $T_\alpha \subseteq T_0$ to minimize $C_\alpha(T)$
- There is a unique smallest subtree

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion
- $T \subset T_0$ is any tree obtained by collapsing internal nodes
- The goodness of the tree T

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T|$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} y_i \quad Q_m(T) = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} (y_i - \hat{c}_m)^2$$

- For each α, find a subtree $T_\alpha \subseteq T_0$ to minimize $C_\alpha(T)$
- There is a unique smallest subtree
  - Successively collapse nodes with smallest increase in $\sum_m N_m Q_m(T)$

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion
- $T \subset T_0$ is any tree obtained by collapsing internal nodes
- The goodness of the tree T

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T|$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} y_i \quad Q_m(T) = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} (y_i - \hat{c}_m)^2$$

- For each α, find a subtree $T_\alpha \subseteq T_0$ to minimize $C_\alpha(T)$
- There is a unique smallest subtree
    - Successively collapse nodes with smallest increase in $\sum_m N_m Q_m(T)$
    - Collapse till we have a single node tree

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion
- $T \subset T_0$ is any tree obtained by collapsing internal nodes
- The goodness of the tree T

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T|$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} y_i \quad Q_m(T) = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} (y_i - \hat{c}_m)^2$$

- For each α, find a subtree $T_\alpha \subseteq T_0$ to minimize $C_\alpha(T)$
- There is a unique smallest subtree
  - Successively collapse nodes with smallest increase in $\sum_m N_m Q_m(T)$
  - Collapse till we have a single node tree
  - The sequence of subtrees will contain T_α

Tree Pruning

- Consider a large tree T_0 with some simple stopping criterion
- $T \subset T_0$ is any tree obtained by collapsing internal nodes
- The goodness of the tree T

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T|$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} y_i \quad Q_m(T) = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} (y_i - \hat{c}_m)^2$$

- For each α, find a subtree $T_\alpha \subseteq T_0$ to minimize $C_\alpha(T)$
- There is a unique smallest subtree
  - Successively collapse nodes with smallest increase in $\sum_m N_m Q_m(T)$
  - Collapse till we have a single node tree
  - The sequence of subtrees will contain T_α
- Choose α by cross-validation