

Supervised Learning

- Decision trees
- Artificial neural nets
- K-nearest neighbor
- Support vectors
- Linear regression
- Logistic regression
- ...

Supervised Learning

- $y=F(x)$: true function (usually not known)
- D : training sample drawn from $F(x)$

57,M,195,0,125,95,39,25,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0	0
78,M,160,1,130,100,37,40,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0	1
69,F,180,0,115,85,40,22,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0	0
18,M,165,0,110,80,41,30,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	0
54,F,135,0,115,95,39,35,1,1,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0	1
84,F,210,1,135,105,39,24,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0	0
89,F,135,0,120,95,36,28,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0	1
49,M,195,0,115,85,39,32,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0	0
40,M,205,0,115,90,37,18,0	0
74,M,250,1,130,100,38,26,1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0	0
77,F,140,0,125,100,40,30,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1	1

...

Supervised Learning

Train Set:

57,M,195,0,125,95,39,25,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0	0
78,M,160,1,130,100,37,40,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0	1
69,F,180,0,115,85,40,22,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0	0
18,M,165,0,110,80,41,30,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	0
54,F,135,0,115,95,39,35,1,1,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0	1
84,F,210,1,135,105,39,24,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0	0
89,F,135,0,120,95,36,28,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,1,0,0	1
49,M,195,0,115,85,39,32,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0	0
40,M,205,0,115,90,37,18,0	0
74,M,250,1,130,100,38,26,1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	0
77,F,140,0,125,100,40,30,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1	1

...

Test Set:

71,M,160,1,130,105,38,20,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0	?
--	---

Supervised Learning

- $F(x)$: true function (usually not known)
- D : training sample drawn from $F(x)$

57,M,195,0,125,95,39,25,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0 0

78,M,160,1,130,100,37,40,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0 1

69,F,180,0,115,85,40,22,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0 0


18,M,165,0,110,80,41,30,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 0

54,F,135,0,115,95,39,35,1,1,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0 1

- $G(x)$: model learned from training sample D

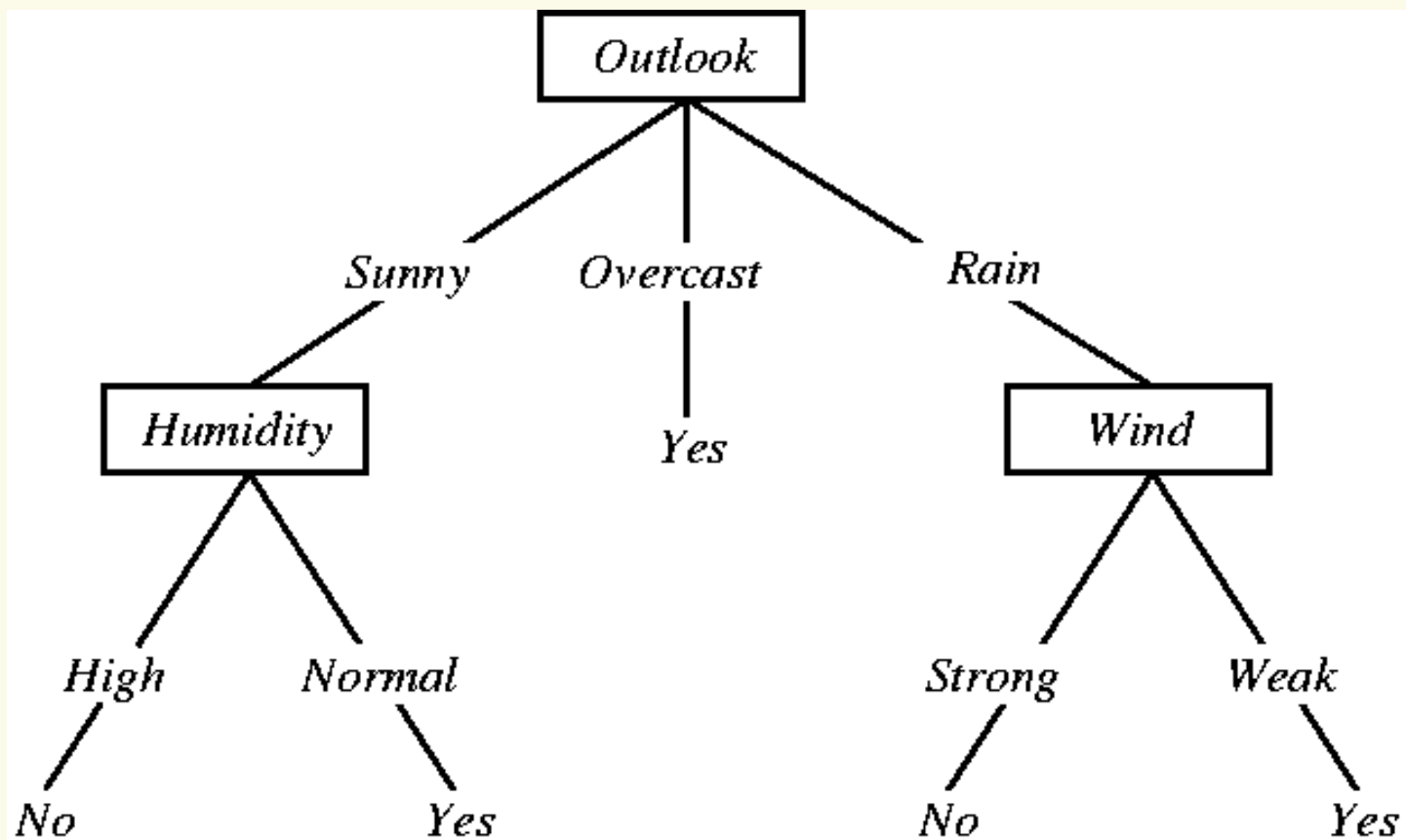
71,M,160,1,130,105,38,20,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0 ?

- Goal: $E\langle (F(x)-G(x))^2 \rangle$ is small (near zero) for future test samples drawn from $F(x)$

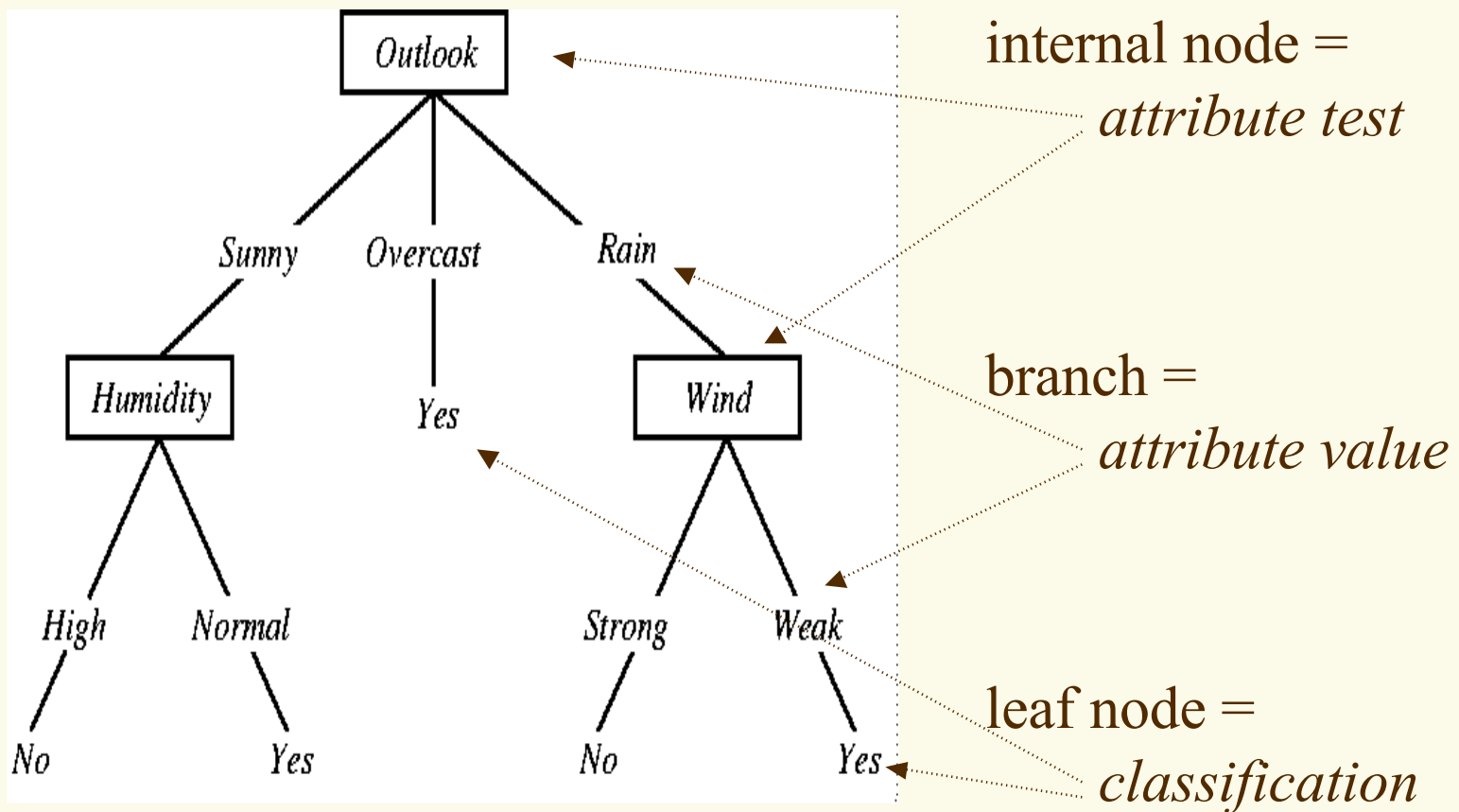
A spiral-bound notebook with a light brown, textured cover and a dark brown border. The notebook is open to a blank page with a light beige, textured background. The words "Decision Trees" are written in a dark brown, serif font in the center of the page. The spiral binding is visible on the left side.

Decision Trees

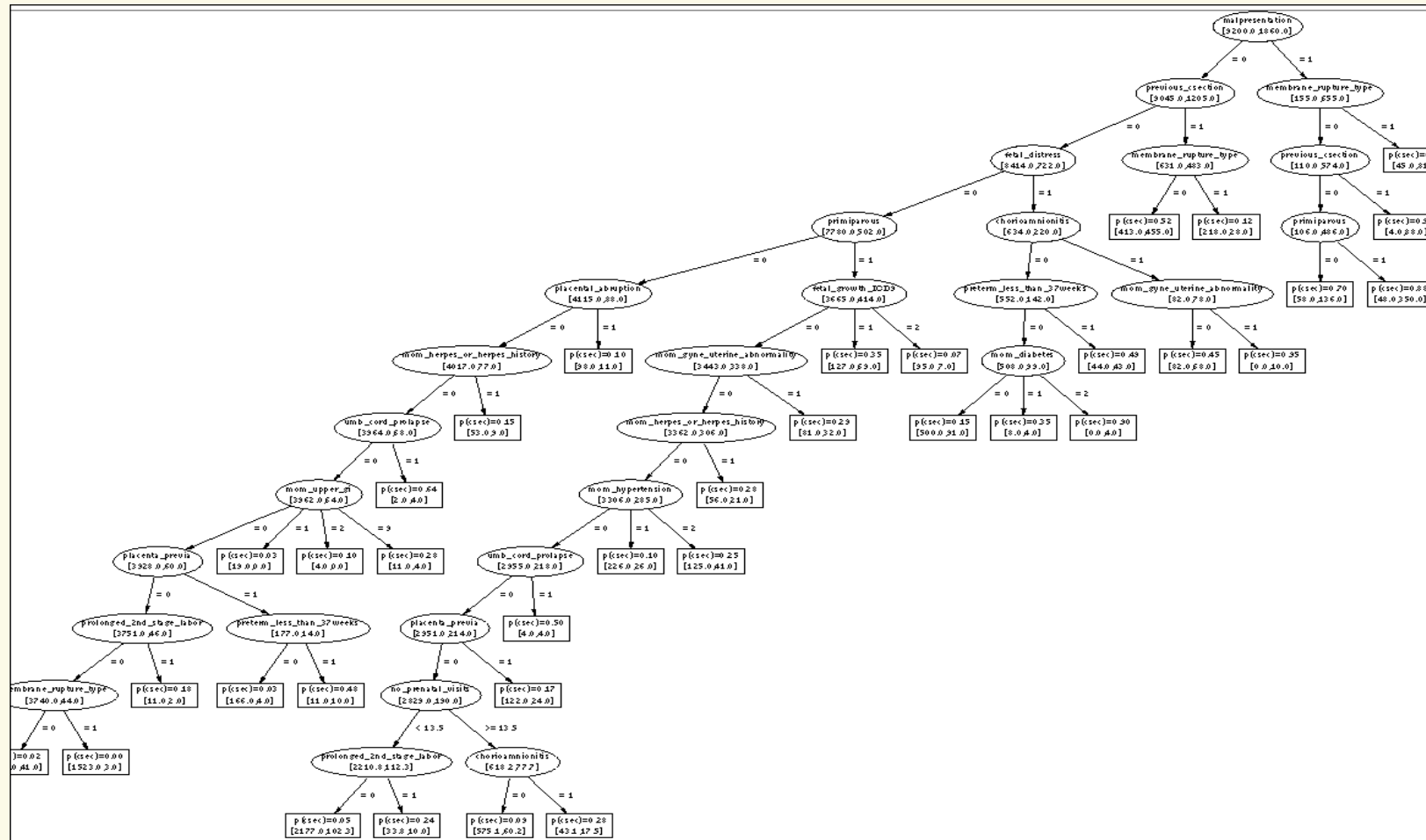
A Simple Decision Tree



Representation



A Real Decision Tree



A Real Decision Tree

Decision Tree Trained on 1000 Patients:

```
+833+167 (tree) 0.8327 0.1673 0
fetal_presentation = 1: +822+116 (tree) 0.8759 0.1241 0
| previous_csection = 0: +767+81 (tree) 0.904 0.096 0
| | primiparous = 0: +399+13 (tree) 0.9673 0.03269 0
| | primiparous = 1: +368+68 (tree) 0.8432 0.1568 0
| | | fetal_distress = 0: +334+47 (tree) 0.8757 0.1243 0
| | | | birth_weight < 3349: +201+10.555 (tree) 0.9482 0.05176 0
| | | | birth_weight >= 3349: +133+36.445 (tree) 0.783 0.217 0
| | | fetal_distress = 1: +34+21 (tree) 0.6161 0.3839 0
| previous_csection = 1: +55+35 (tree) 0.6099 0.3901 0
fetal_presentation = 2: +3+29 (tree) 0.1061 0.8939 1
fetal_presentation = 3: +8+22 (tree) 0.2742 0.7258 1
```

Real Data: C-Section Prediction

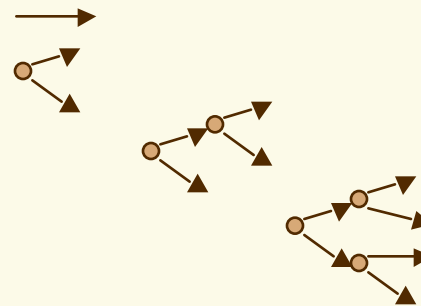
Demo summary:

- Fast
- Reasonably intelligible
- Larger training sample => larger tree
- Different training sample => different tree

collaboration with Magee Hospital, Siemens Research, Tom Mitchell

Search Space

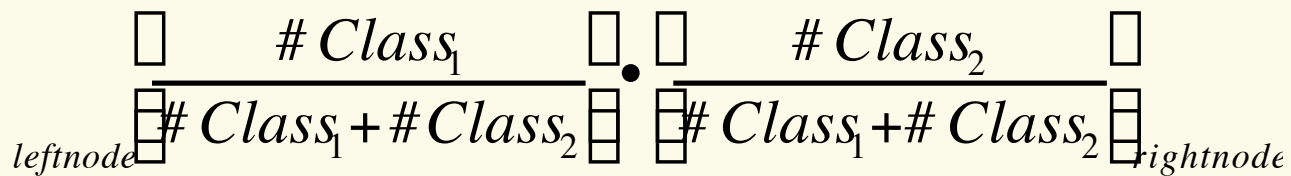
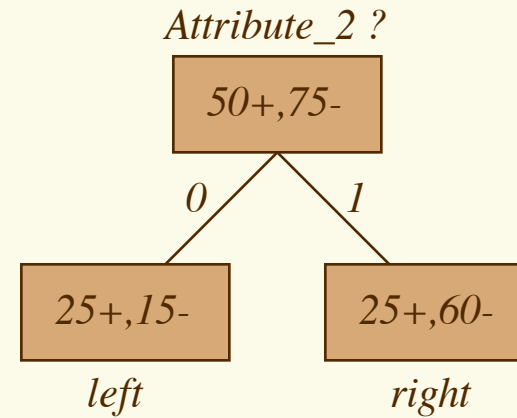
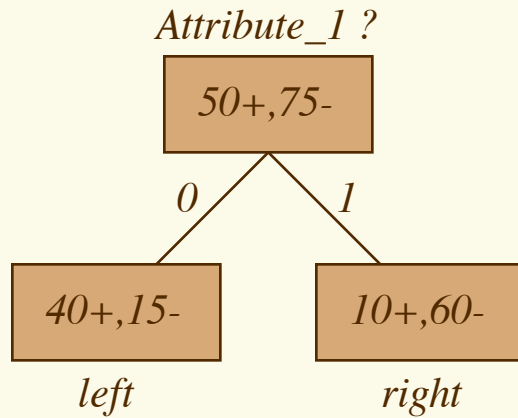
- all possible sequences of all possible tests
- very large search space, e.g., if N binary attributes:
 - 1 null tree
 - N trees with 1 (root) test
 - $N*(N-1)$ trees with 2 tests
 - $N*(N-1)*(N-1)$ trees with 3 tests
 - $\approx N^4$ trees with 4 tests
- size of search space is exponential in number of attributes
 - too big to search exhaustively
 - exhaustive search probably would overfit data (too many models)
 - so what do we do instead?



Top-Down Induction of Decision Trees

- a.k.a. Recursive Partitioning
 - *find “best” attribute test to install at root*
 - *split data on root test*
 - *find “best” attribute tests to install at each new node*
 - *split data on new tests*
 - *repeat until:*
 - all nodes are pure
 - all nodes contain fewer than k cases
 - distributions at nodes indistinguishable from chance
 - tree reaches predetermined max depth
 - no more attributes to test

Find “Best” Split?



0.6234

0.4412

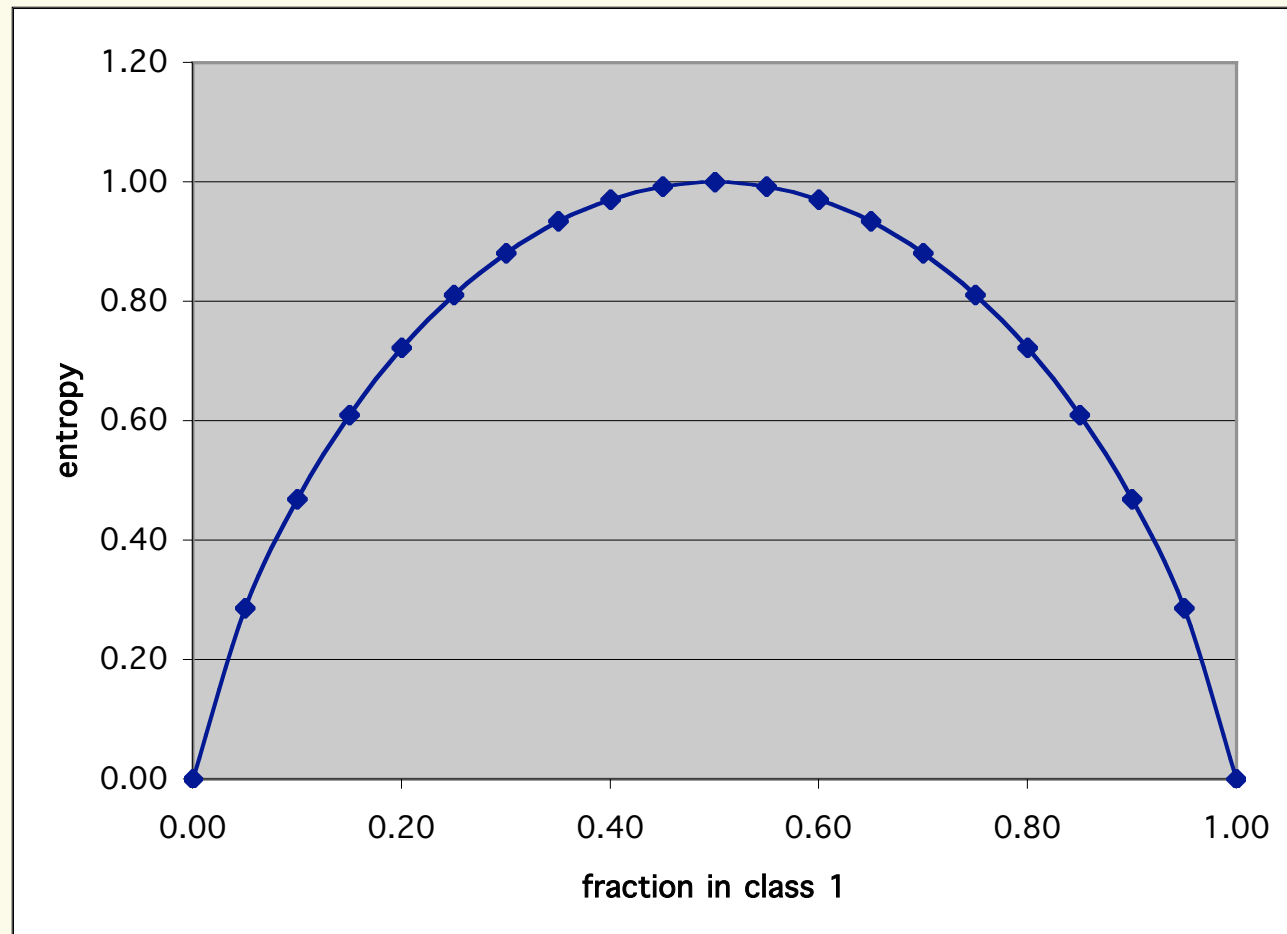
Splitting Rules

- Information Gain = reduction in entropy due to splitting on an attribute
- Entropy = expected number of bits needed to encode the class of a randomly drawn + or – example using the optimal info-theory coding

$$Entropy = -\sum p_+ \log_2 p_+ - \sum p_- \log_2 p_-$$

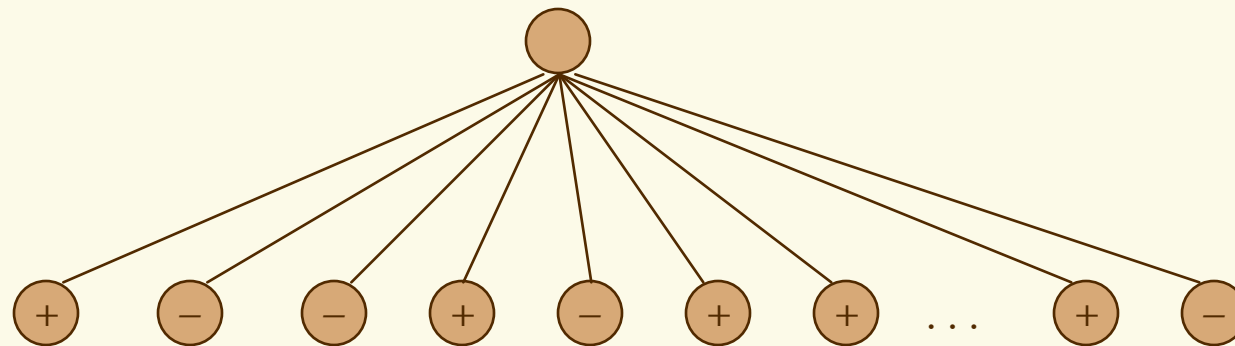
$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Entropy



Splitting Rules

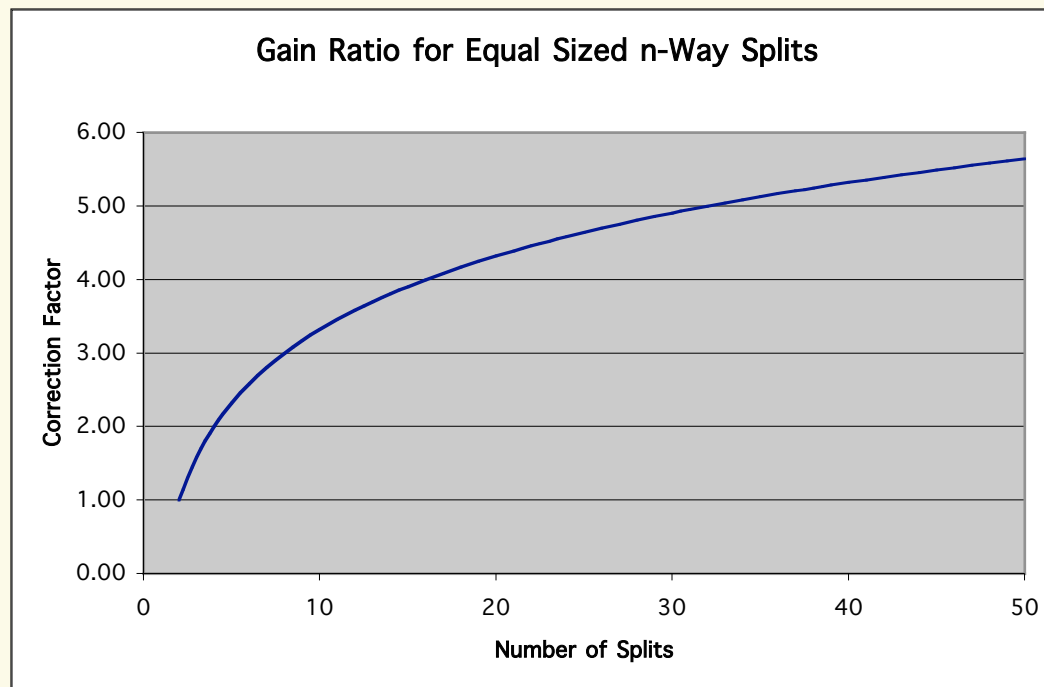
- Problem with Node Purity and Information Gain:
 - prefer attributes with many values
 - extreme cases:
 - Social Security Numbers
 - patient ID's
 - integer/nominal attributes with many values (JulianDay)



Splitting Rules

$$\text{GainRatio}(S, A) = \frac{\text{Entropy}(S) - \sum_{v \in \text{Value}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)}{\sum_{v \in \text{Value}(A)} \frac{|S_v|}{|S|} \log_2 \frac{|S_v|}{|S|}}$$

Gain_Ratio Correction Factor



Splitting Rules

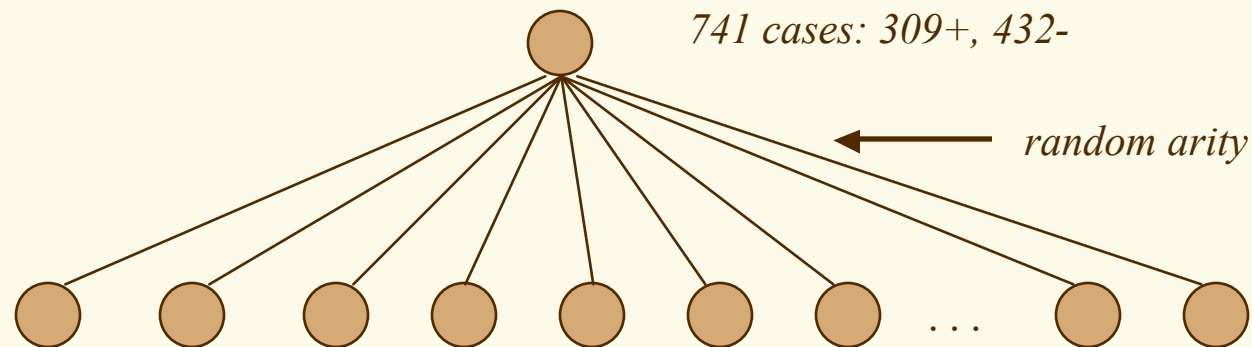
- GINI Index
 - Measure of node impurity

$$GINI_{node}(Node) = 1 - \sum_{c \in \text{classes}} [p_c]^2$$

$$GINI_{split}(A) = \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} GINI(N_v)$$

Experiment

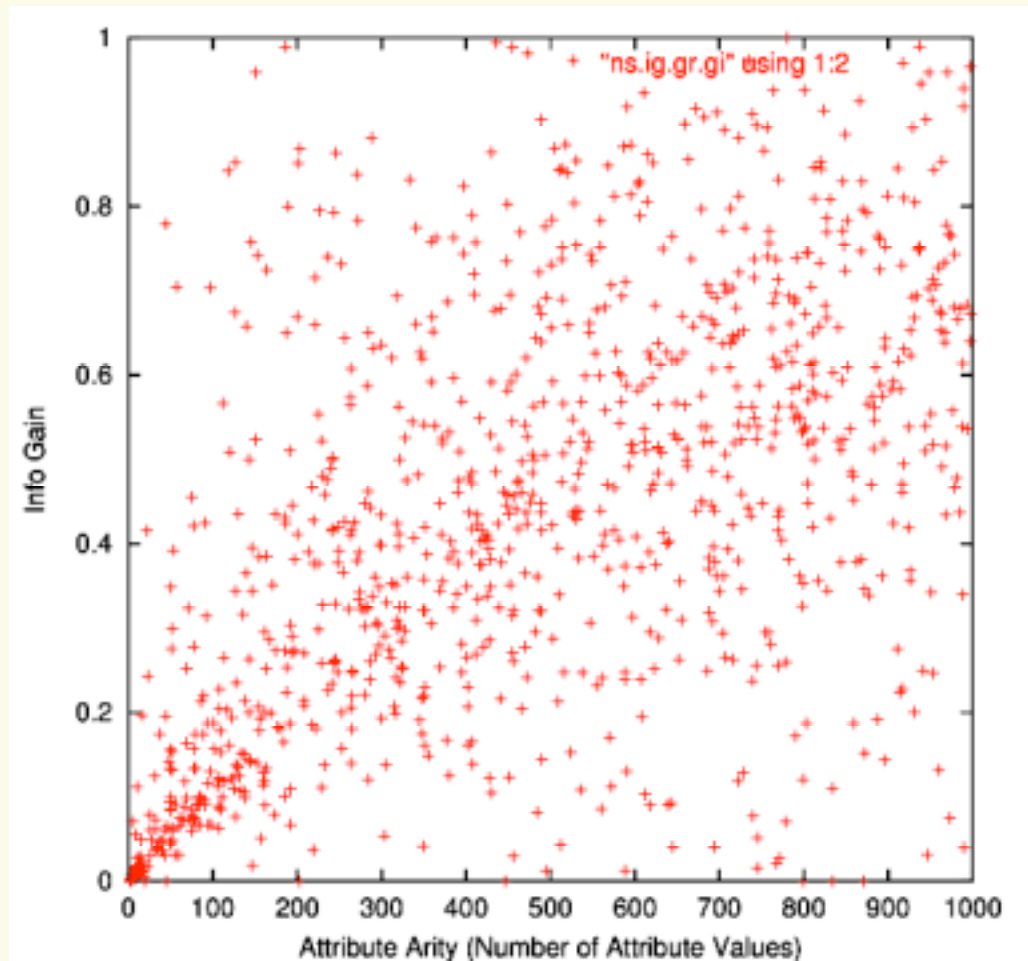
- Randomly select # of cases: 2-1000
- Randomly select fraction of +'s and -'s
- Randomly select attribute arity: 2-1000
- Randomly assign cases to branches!!!!
- Compute IG, GR, GINI



Info_Gain

Good Splits

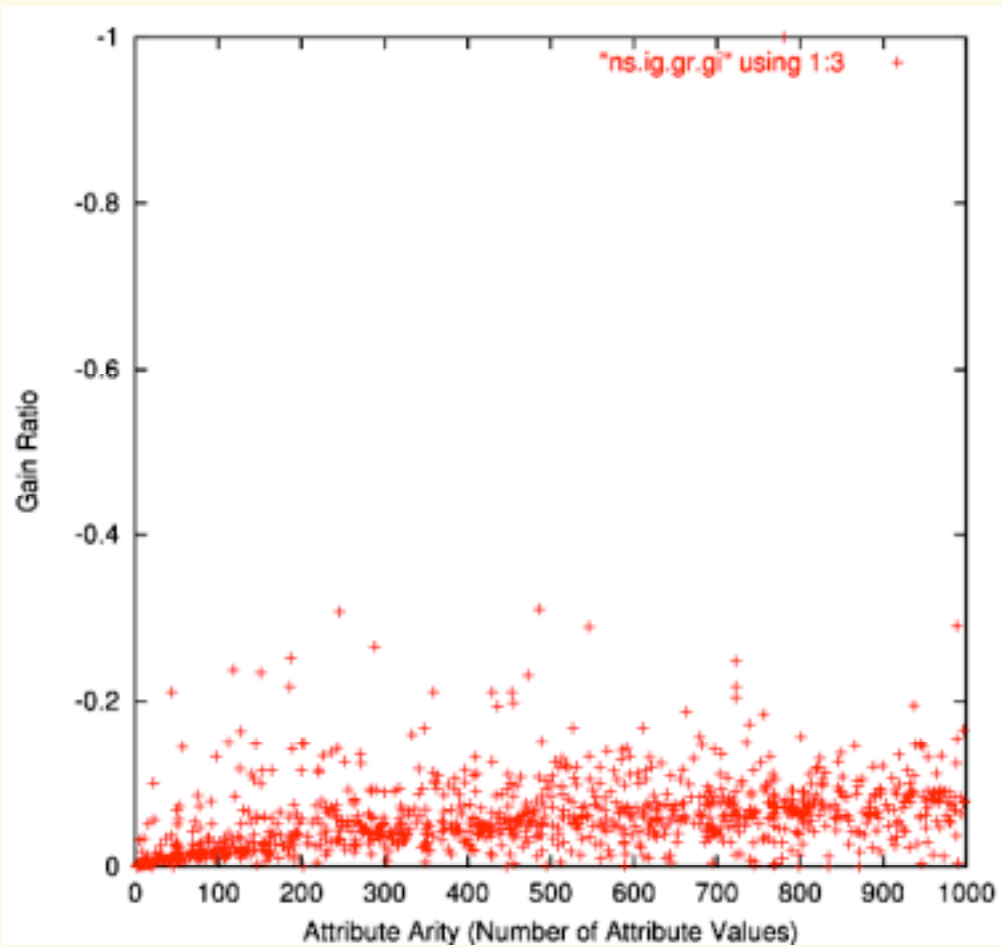
Poor Splits



Gain_Ratio

Good Splits

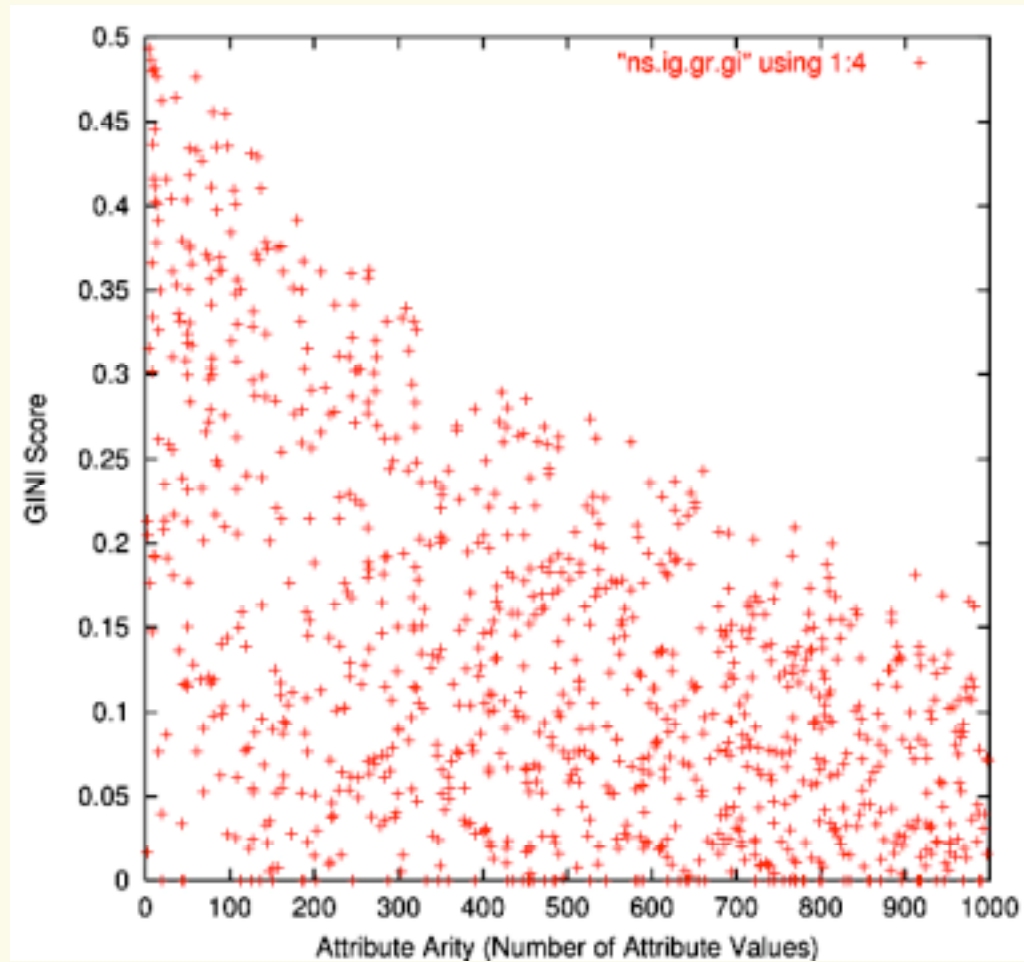
Poor Splits



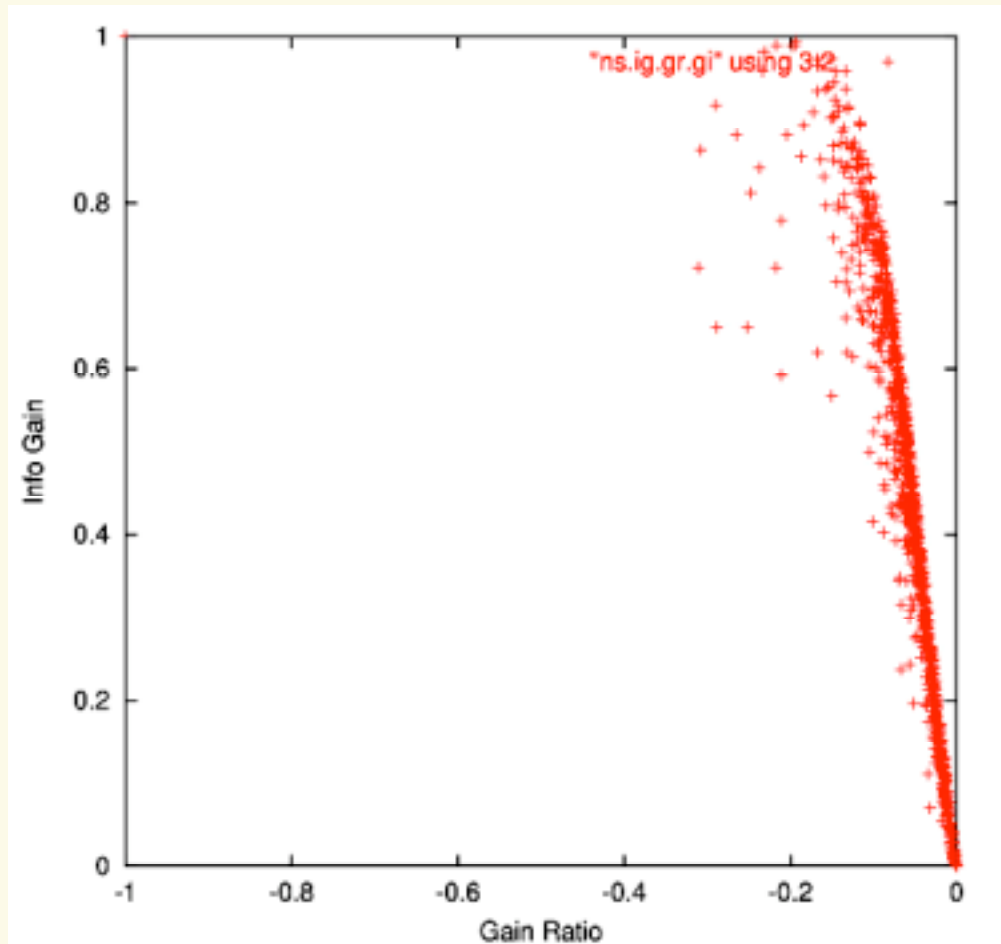
GINI Score

Poor Splits

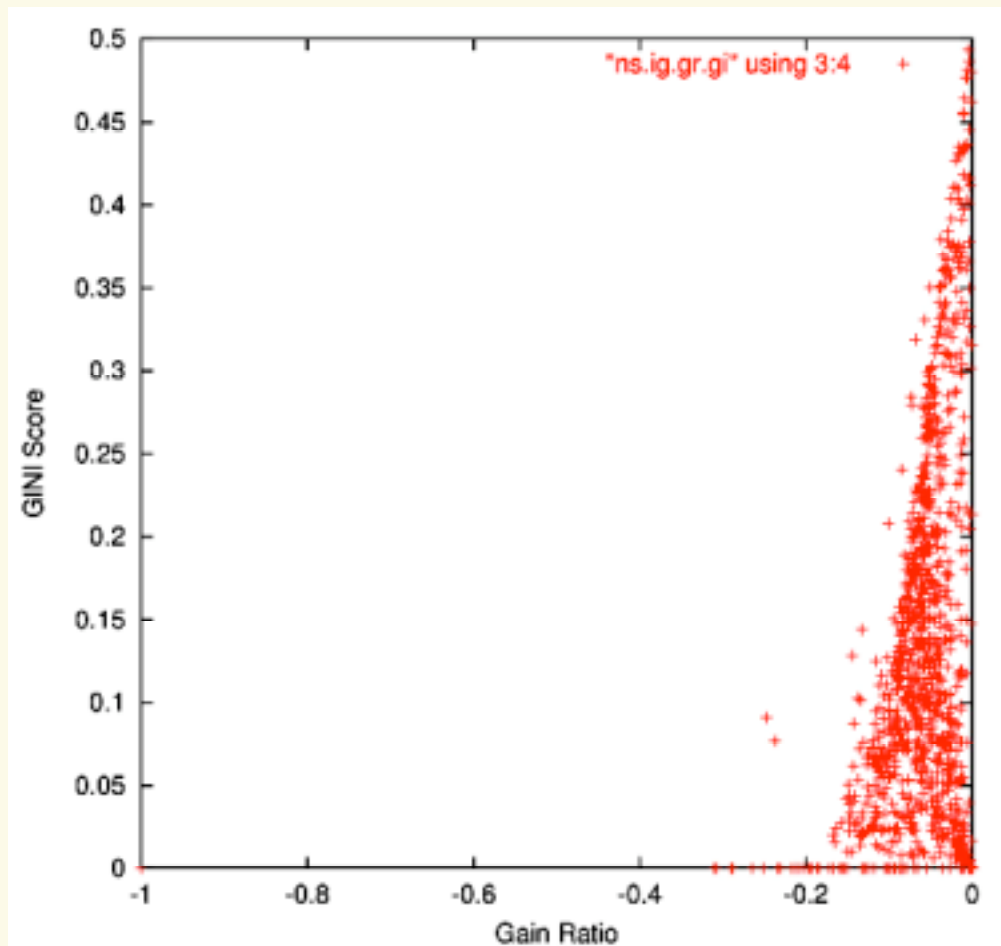
Good Splits



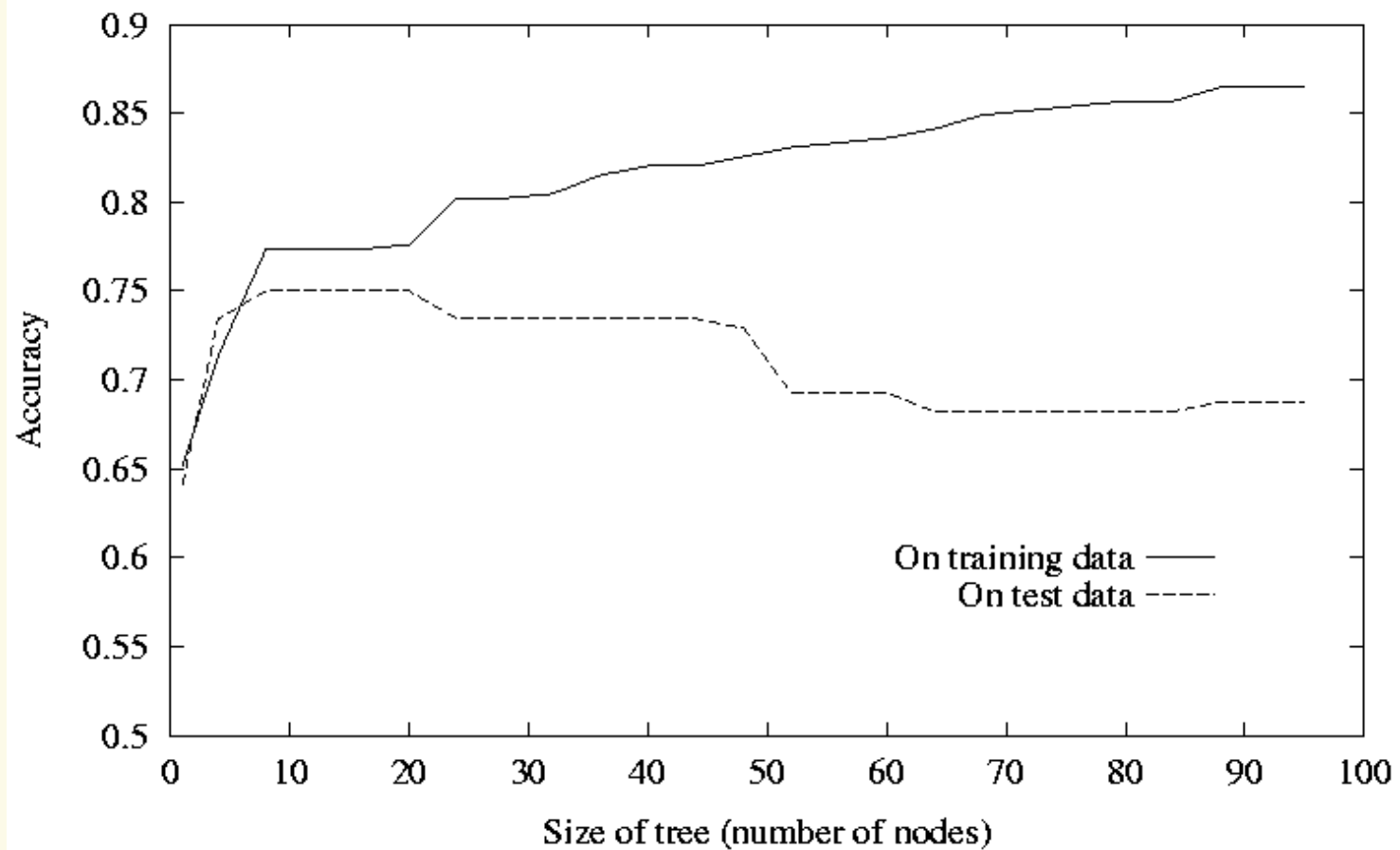
Info_Gain vs. Gain_Ratio



GINI Score vs. Gain_Ratio



Overfitting



©Tom Mitchell, McGraw Hill, 1997

Pre-Pruning (Early Stopping)

- Evaluate splits before installing them:
 - don't install splits that don't look worthwhile
 - when no worthwhile splits to install, done
- Seems right, but:
 - hard to properly evaluate split without seeing what splits would follow it (use lookahead?)
 - some attributes useful only in combination with other attributes
 - suppose no single split looks good at root node?

Post-Pruning

- Grow decision tree to full depth (no pre-pruning)
- Prune-back full tree by eliminating splits that do not appear to be warranted statistically
- Use train set, or an independent prune/test set, to evaluate splits
- Stop pruning when remaining splits all appear to be warranted
- Alternate approach: convert to rules, then prune rules

Greedy vs. Optimal

- Optimal
 - Maximum expected accuracy (test set)
 - Minimum size tree
 - Minimum depth tree
 - Fewest attributes tested
 - Easiest to understand
- Test order not always important for accuracy
- Sometimes random splits perform well

Decision Tree Predictions

- Classification
- Simple probability
- Smoothed probability
- Probability with threshold(s)

Performance Measures

- Accuracy
 - High accuracy doesn't mean good performance
 - Accuracy can be misleading
 - What threshold to use for accuracy?
- Root-Mean-Squared-Error

$$\text{RMSE} = \sqrt{\sum_{i=1}^{\# \text{ test}} (1 - \text{Pred_Prob}_i(\text{True_Class}_i))^2}$$

- Other measures: ROC, Precision/Recall, ...

Attribute Types

- Boolean
- Nominal
- Ordinal
- Integer
- Continuous
 - Sort by value, then find best threshold for binary split
 - Cluster into n intervals and do n-way split

A spiral-bound notebook with a brown cover and a cream-colored page. The spiral binding is on the left side. The page contains the title 'Missing Attribute Values' and a single bullet point.

Missing Attribute Values

- Some data sets have many missing values

Regression Trees vs. Classification

- Split criterion: minimize RMSE at node
- Tree yields discrete set of predictions

$$\text{RMSE} = \sqrt{\sum_{i=1}^{\# \text{ test}} (\text{True}_i - \text{Pred}_i)^2}$$

Converting Decision Trees to Rules

- each path from root to a leaf is a separate rule:

fetal_presentation = 1: +822+116 (tree) 0.8759 0.1241 0

| previous_csection = 0: +767+81 (tree) 0.904 0.096 0

| | primiparous = 1: +368+68 (tree) 0.8432 0.1568 0

| | | fetal_distress = 0: +334+47 (tree) 0.8757 0.1243 0

| | | | birth_weight < 3349: +201+10.555 (tree) 0.9482 0.05176 0

fetal_presentation = 2: +3+29 (tree) 0.1061 0.8939 1

fetal_presentation = 3: +8+22 (tree) 0.2742 0.7258 1

if (fp=1 & ¬pc & primip & ¬fd & bw<3349) => 0,

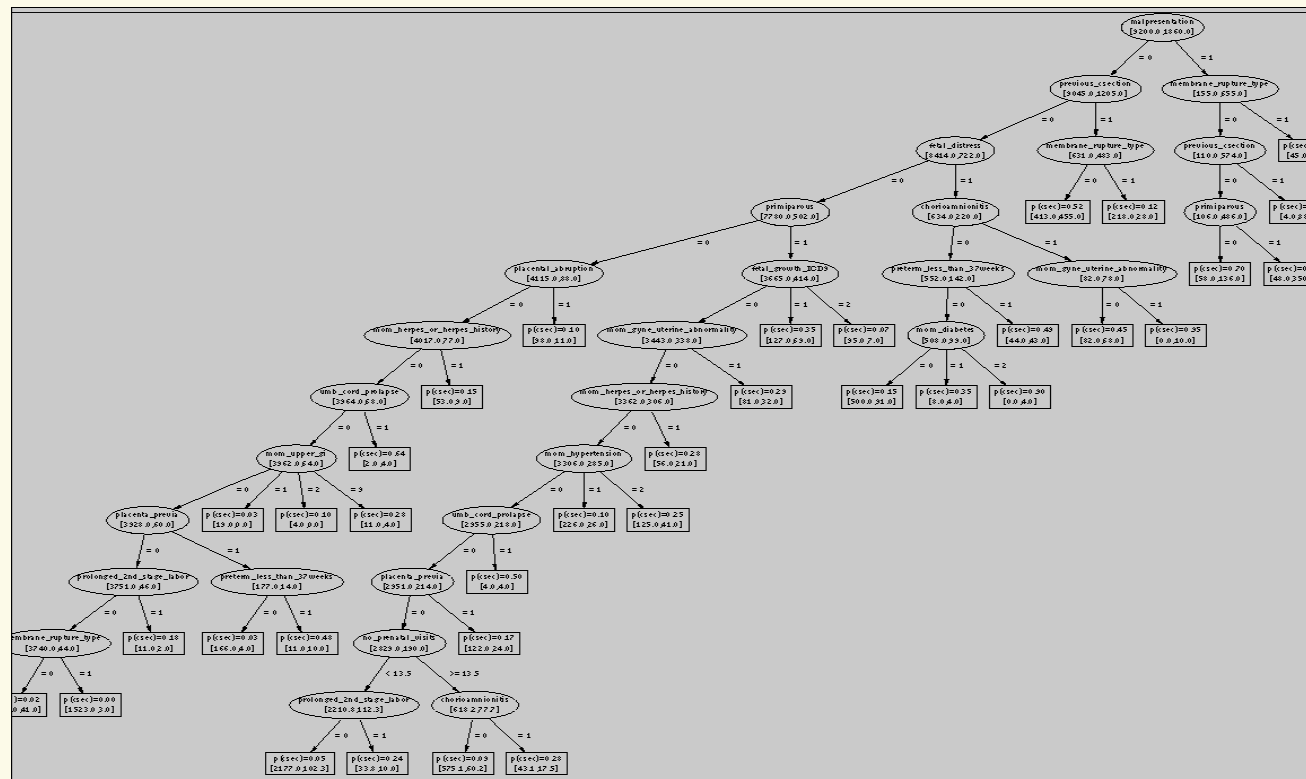
if (fp=2) => 1,

if (fp=3) => 1.

Advantages of Decision Trees

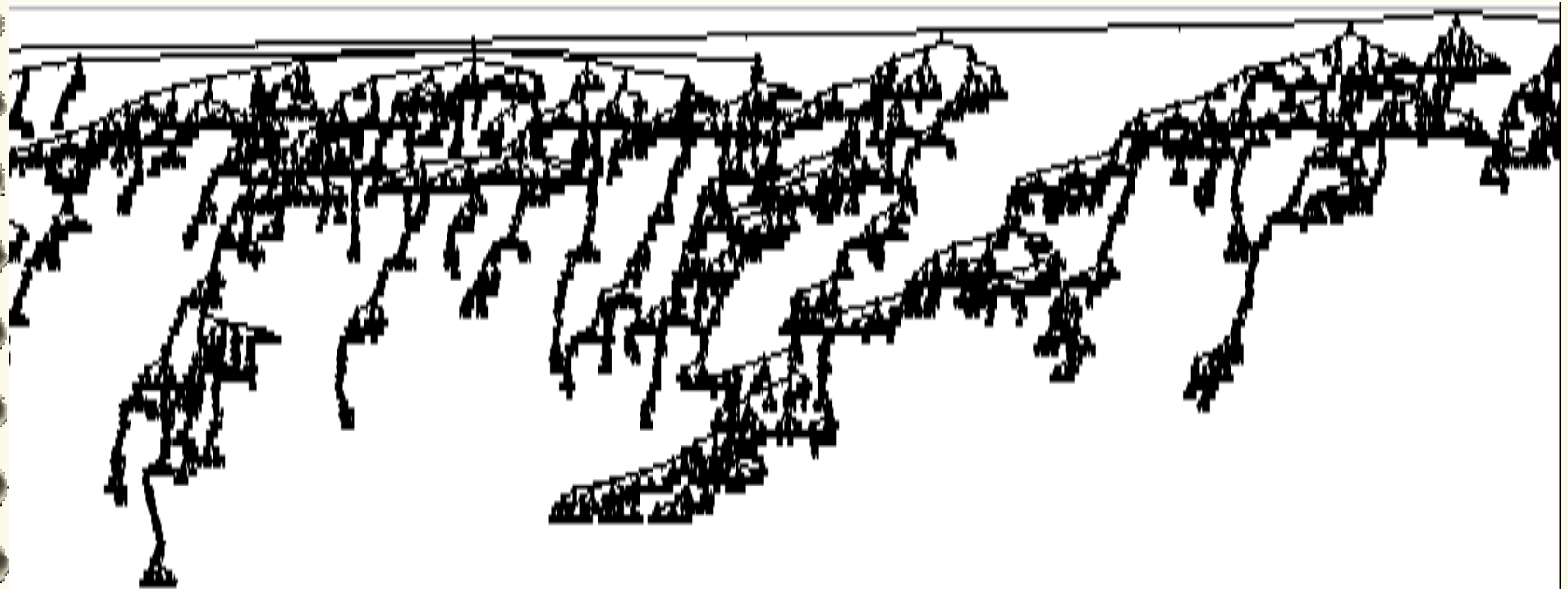
- TDIDT is relatively fast, even with large data sets (10^6) and many attributes (10^3)
 - advantage of recursive partitioning: only process all cases at root
- Small-medium size trees usually intelligible
- Can be converted to rules
- TDIDT does feature selection
- TDIDT often yields compact models (Occam's Razor)
- Decision tree representation is understandable

Decision Trees are Intelligible



Not *ALL* Decision Trees Are Intelligible

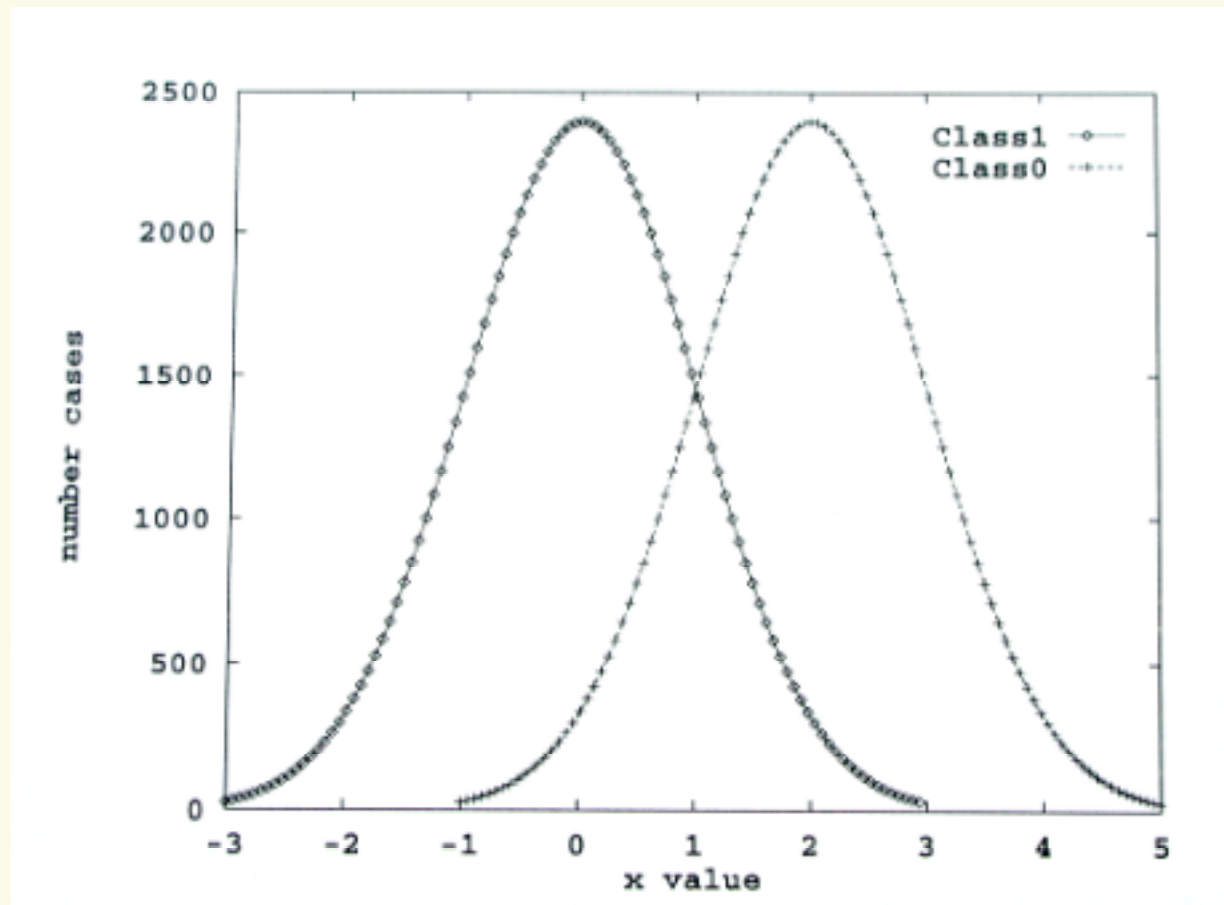
Part of Best Performing C-Section Decision Tree



Predicting Probabilities with Trees

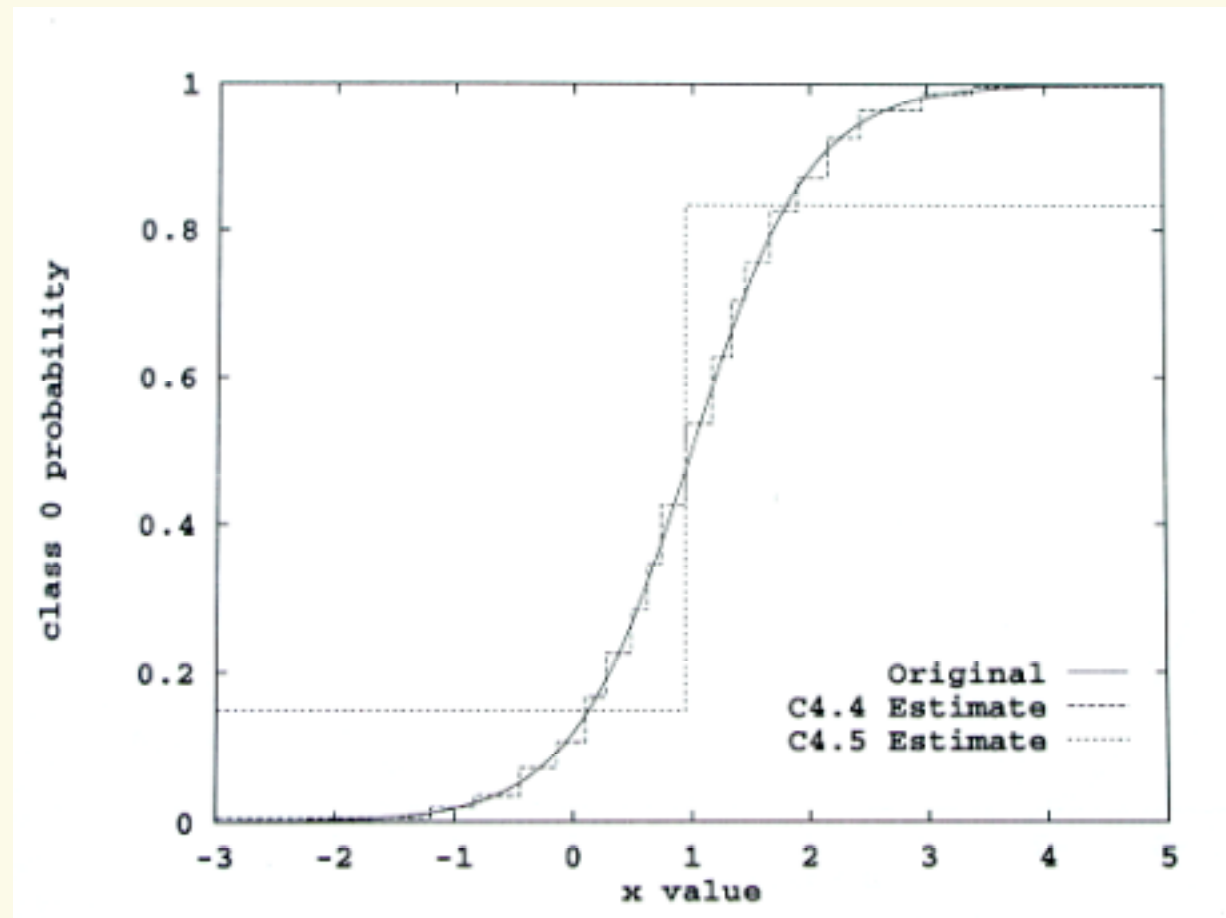
- Small Tree
 - few leafs
 - few discrete probabilities
- Large Tree
 - many leafs
 - few cases per leaf
 - few discrete probabilities
 - probability estimates based on small/noisy samples
- What to do?

A Simple Two-Class Problem



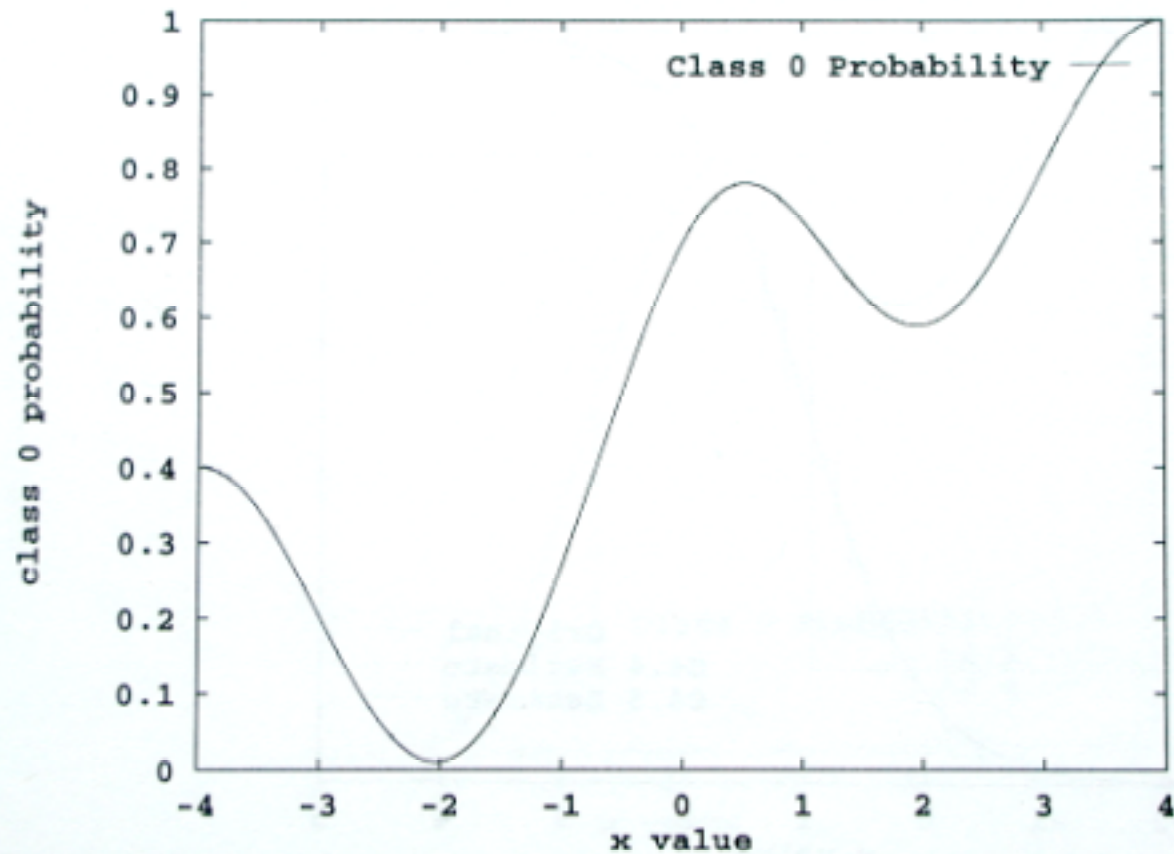
From Provost, Domingos pet-mlj 2002

Classification vs. Predicting Probs



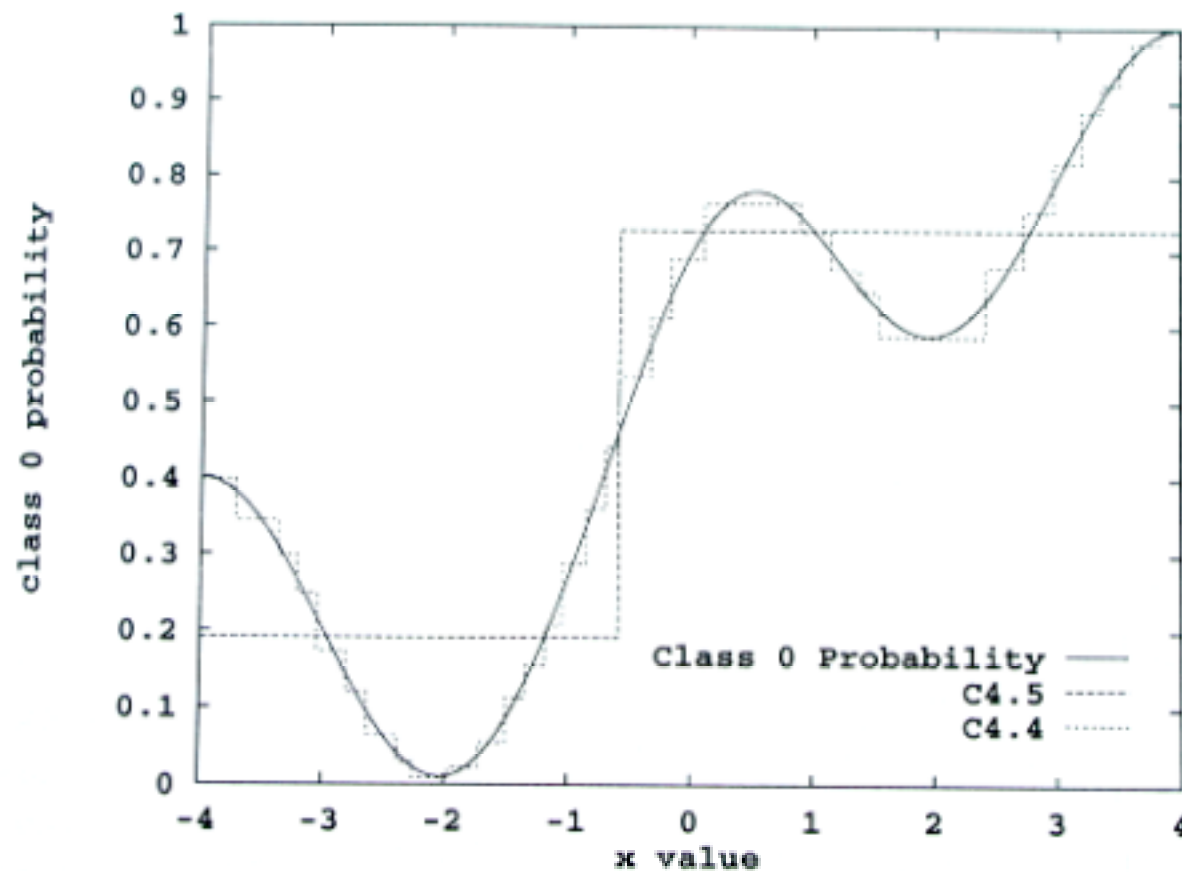
From Provost, Domingos pet-mlj 2002

A Harder Two-Class Problem



From Provost, Domingos pet-mlj 2002

Classification vs. Prob Prediction



From Provost, Domingos pet-mlj 2002

PET: Probability Estimation Trees

- Smooth large trees
 - correct estimates from small samples at leafs
- Average many trees
 - average of many things each with a few discrete values is more continuous
 - averages improve quality of estimates
- Both

Laplacian Smoothing

- Small leaf count: 4+, 1-
- Maximum Likelihood Estimate: k/N
 - $P(+)=4/5=0.8$; $P(-)=1/5=0.2$?
- Could easily be 3+, 2- or even 2+, 3-, or worse
- Laplacian Correction: $(k+1)/(N+C)$
 - $P(+)= (4+1)/(5+2) = 5/7 = 0.7143$
 - $P(-)= (1+1)/(5+2) = 2/7 = 0.2857$
 - If $N=0$, $P(+)=P(-)=1/2$
 - Bias towards $P(\text{class}) = 1/C$

Bagging (Model Averaging)

- Train many trees with different random samples
- Average prediction from each tree

Results

Table II. Summary of experimental results: AUC comparisons.

Systems	Wins-Ties-Losses	Avg. diff. (%)	Sign test	Wilcoxon test
C4.4 vs. C4.5	18 - 1 - 6	2.0	1.0	0.3
C4.4 vs. C4.5-L	13 - 3 - 9	0.2	30.0	30.0
C4.5-L vs. C4.5	21 - 2 - 2	1.7	0.1	0.1
C4.5-B vs. C4.5	24 - 1 - 0	7.3	0.1	0.1
C4.4-B vs. C4.4	23 - 2 - 0	5.3	0.1	0.1
C4.4-B vs. C4.5-B	11 - 5 - 9	-0.1	45.0	50.0

C4.4: no pruning or collapsing
“L”: Laplacian Smoothing
“B”: bagging

Weaknesses of Decision Trees

- Large or complex trees can be just as unintelligible as other models
- Trees don't easily represent some basic concepts such as M-of-N, parity, non-axis-aligned classes...
- Don't handle real-valued parameters as well as Booleans
- If model depends on summing contribution of many different attributes, DTs probably won't do well
- DTs that look very different can be same/similar
- Usually poor for predicting continuous values (regression)
- Propositional (as opposed to 1st order)
- Recursive partitioning: run out of data fast as descend tree

Popular Decision Tree Packages

- ID3 (ID4, ID5, ...) [Quinlan]
 - research code with many variations introduced to test new ideas
- CART: Classification and Regression Trees [Breiman]
 - best known package to people outside machine learning
 - 1st chapter of CART book is a good introduction to basic issues
- C4.5 (C5.0) [Quinlan]
 - most popular package in machine learning community
 - both decision trees and rules
- IND (INDuce) [Buntine]
 - decision trees for Bayesians (good at generating probabilities)
 - available from NASA Ames for use in U.S.

When to Use Decision Trees

- Regression doesn't work
- Model intelligibility is important
- Problem does not depend on many features
 - Modest subset of features contains relevant info
 - not vision
- Speed of learning is important
- Linear combinations of features not critical
- Medium to large training sets

Current Research

- Increasing representational power to include M-of-N splits, non-axis-parallel splits, perceptron-like splits, ...
- Handling real-valued attributes better
- Using DTs to explain other models such as neural nets
- Incorporating background knowledge
- TDIDT on really large datasets
 - $\gg 10^6$ training cases
 - $\gg 10^3$ attributes
- Better feature selection
- Unequal attribute costs
- Decision trees optimized for metrics other than accuracy