BITS Pilani
Pilani Campus

# Machine Learning

## AIML CZG565

## M5 : Decision Tree Classifier

Course Faculty of M.Tech Cluster
BITS – CSIS - WILP

# Machine Learning

## Disclaimer and Acknowledgement


- These content of modules & context under topics are planned by the course owner Dr. Sugata, with grateful acknowledgement to many others who made their course materials freely available online
- We here by acknowledge all the contributors for their material and inputs.
- We have provided source information wherever necessary
- Students are requested to refer to the textbook w.r.t detailed content of the presentation deck shared over canvas
- We have reduced the slides from canvas and modified the content flow to suit the requirements of the course and for ease of class presentation

**Slide Source / Preparation / Review:**
From BITS Pilani WILP: Prof.Sugata, Prof.Chetana, Prof.Rajavadhana, Prof.Monali, Prof.Anita, Prof.Sangeetha, Prof.Swarna, Prof.Srinath
External: CS109 and CS229 Stanford lecture notes, Dr.Andrew NG and many others who made their course materials freely available online

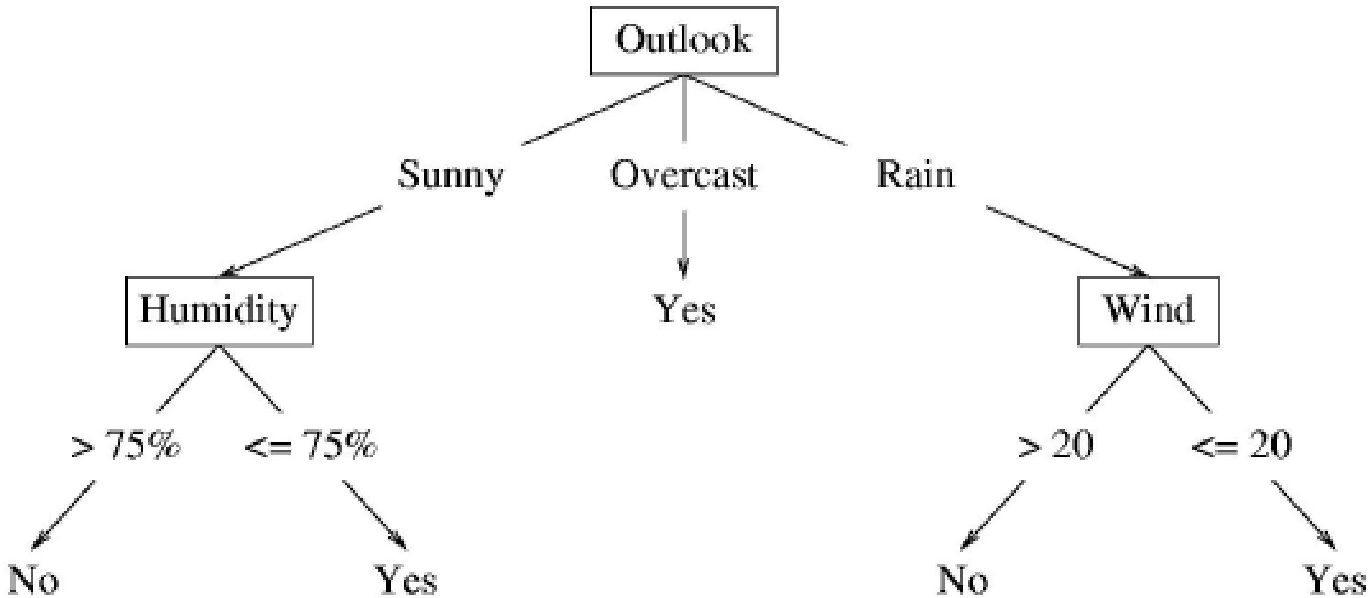
# Course Plan

| | |
|---|---|
| M1 | Introduction |
| M2 | Machine learning Workflow |
| M3 | Linear Models for Regression |
| M4 | Linear Models for Classification |
| M5 | Decision Tree |
| M6 | Instance Based Learning |
| M7 | Support Vector Machine |
| M8 | Bayesian Learning |
| M9 | Ensemble Learning |
| M10 | Unsupervised Learning |
| M11 | Machine Learning Model Evaluation/Comparison |

# Agenda

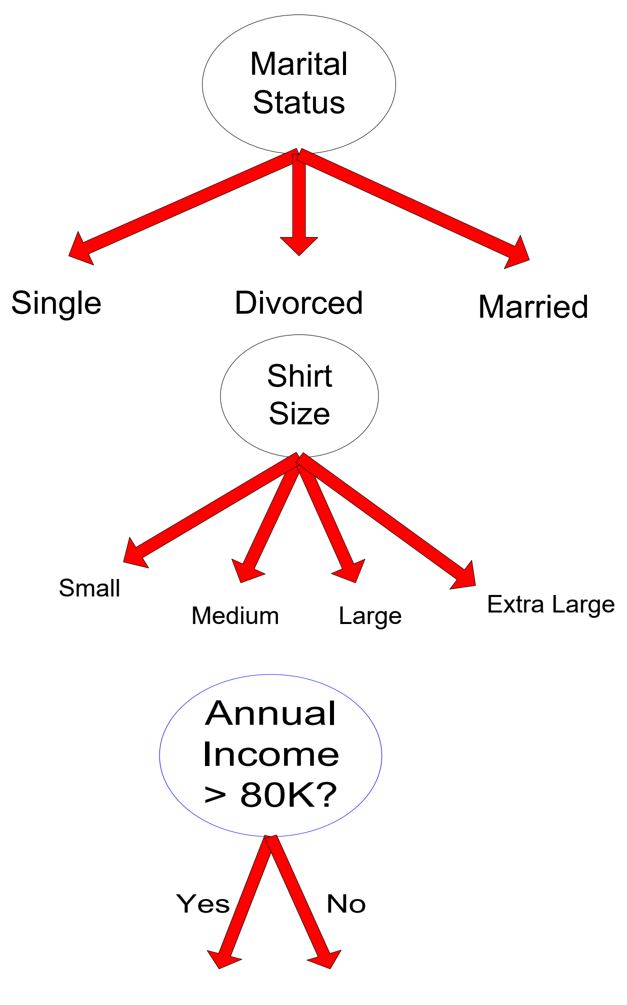
- Information Theory
- Entropy Based Decision Tree Construction
- Avoiding Overfitting
- Minimum Description Length
- Handling Continuous valued attributes

# Decision Trees – Dealing with Continuous Values

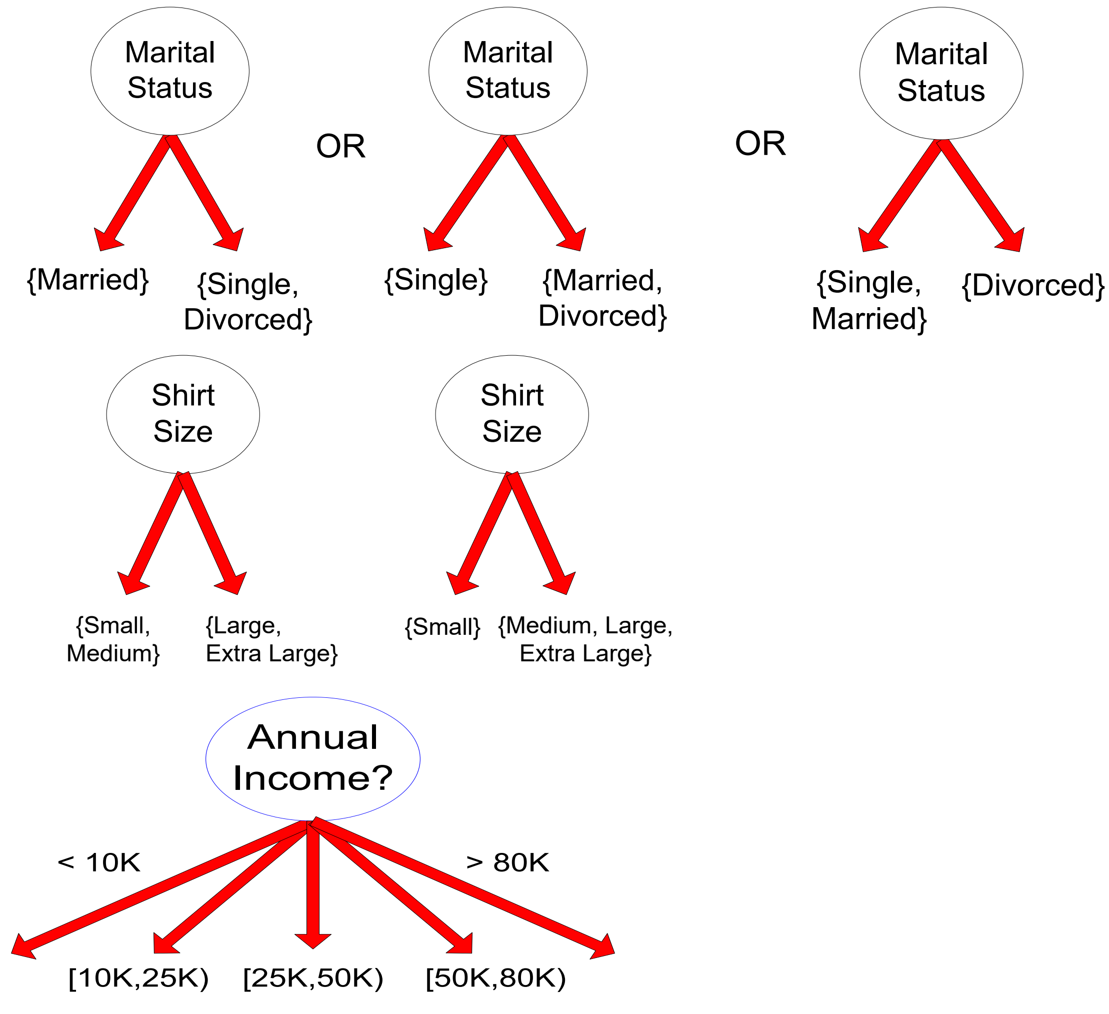


# Summary

## Sample splits of Different Attribute Type




(i) Binary split




(ii) Multi-way split

# Decision Trees – Dealing with Continuous Values – Problem Type 5

- Given a continuous-valued attribute *A*, dynamically create a new attribute $A_c$

$$A_c = \text{True } \textit{if } A < c, \text{ False } \textit{otherwise}$$

- How to determine threshold value *c* ?

**Way 1: Multi-way Split**

- Example. *Temperature* in the *PlayTennis* example
  - Sort the examples according to *Temperature*

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *Temperature* | 40 | 48 | \| | 60 | 72 | 80 | \| | 90 |
| *PlayTennis* | No | No | *54* | Yes | Yes | Yes | *85* | No |

  - Determine candidate thresholds by averaging consecutive values where there is a change in classification: (48+60)/2=54 and (80+90)/2=85

# Decision Trees – Dealing with Continuous Values

- Given a continuous-valued attribute $A$, dynamically create a new attribute $A_c$

$$A_c = \text{True } \textit{if } A < c, \text{ False } \textit{otherwise}$$

- How to determine threshold value $c$ ?

**Way 2: Binary Splits**

- Example. *Annual Income*

Sort the examples according to *Annual Income*

Linearly scan all possible threshold, to determine best split point where the impurity is the least

| ID | Home Owner | Marital Status | Annual Income | Defaulted? |
|---|---|---|---|---|
| 1 | Yes | Single | 125000 | No |
| 2 | No | Married | 100000 | No |
| 3 | No | Single | 70000 | No |
| 4 | Yes | Married | 120000 | No |
| 5 | No | Divorced | 95000 | Yes |
| 6 | No | Single | 60000 | No |
| 7 | Yes | Divorced | 220000 | No |
| 8 | No | Single | 85000 | Yes |
| 9 | No | Married | 75000 | No |
| 10 | No | Single | 90000 | Yes |

# Decision Trees – Dealing with Continuous Values – Problem Type 6

- Given a continuous-valued attribute $A$, dynamically create a new attribute $A_c$

$$A_c = \text{True } \textit{if } A < c, \text{ False } \textit{otherwise}$$

- How to determine threshold value $c$ ?

**Way 2: Binary Splits**

- Example. *Annual Income*

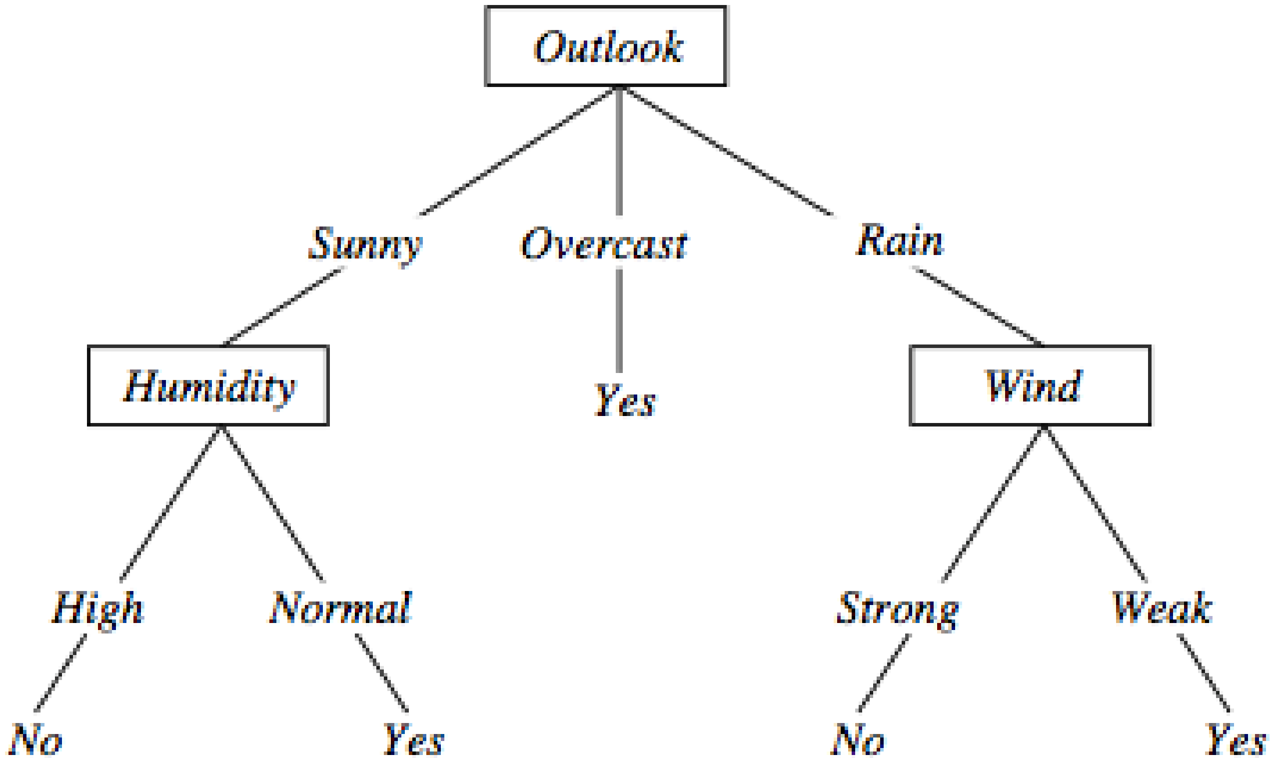
| Class | No | No | No | Yes | Yes | Yes | No | No | No | No |
|---|---|---|---|---|---|---|---|---|---|---|
| | Annual Income (in '000s) | | | | | | | | | |
| Sorted Values → | 60 | 70 | 75 | 85 | 90 | 95 | 100 | 120 | 125 | 220 |

| Split Positions → | 55 | | 65 | | 72.5 | | 80 | | 87.5 | | 92.5 | | 97.5 | | 110 | | 122.5 | | 172.5 | | 230 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > |
| Yes | 0 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 1 | 2 | 2 | 1 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 0 |
| No | 0 | 7 | 1 | 6 | 2 | 5 | 3 | 4 | 3 | 4 | 3 | 4 | 3 | 4 | 4 | 3 | 5 | 2 | 6 | 1 | 7 | 0 |
| Gini | 0.420 | | 0.400 | | 0.375 | | 0.343 | | 0.417 | | 0.400 | | *0.300* | | 0.343 | | 0.375 | | 0.400 | | 0.420 | |

# Computing Information-Gain for Continuous-Valued Attributes - Summary

- Let attribute A be a continuous-valued attribute
- Must determine the *best split point* for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible *split point*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of $a_i$ and $a_{i+1}$
  - The point with the *minimum expected information requirement* for A is selected as the split-point for A
- Split:
  - D1 is the set of tuples in D satisfying A ≤ split-point, and D2 is the set of tuples in D satisfying A > split-point

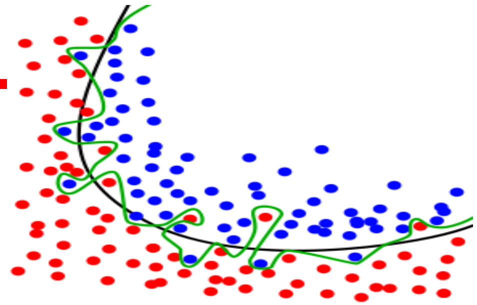
# Overfitting in decision trees




⟨*Outlook=Sunny, Temp=Hot, Humidity=Normal, Wind=Strong, PlayTennis=No* ⟩

New noisy example causes splitting of second leaf node.

# Overfitting and Tree Pruning



- Overfitting: An induced tree may overfit the training data
  - Too many branches, may reflect anomalies due to noise or outliers in the training data
  - Poor accuracy for unseen samples

<u>Approaches to tackle:</u>

- Pruning
- Constraint on the depth of tree
- Constraint on the minimum nodes allowed at leaf


BITS Pilani, Pilani Campus

# Avoid overfitting in Decision Trees

- **Two strategies:**
  1. Stop growing the tree earlier the tree, before perfect classification
  2. Allow the tree to *overfit* the data, and then *post-prune* the tree
- Training and validation set
  - split the training in two parts (training and validation) and use validation to assess the utility of *post-pruning*
    - *Reduced error pruning*
    - *Rule post pruning*

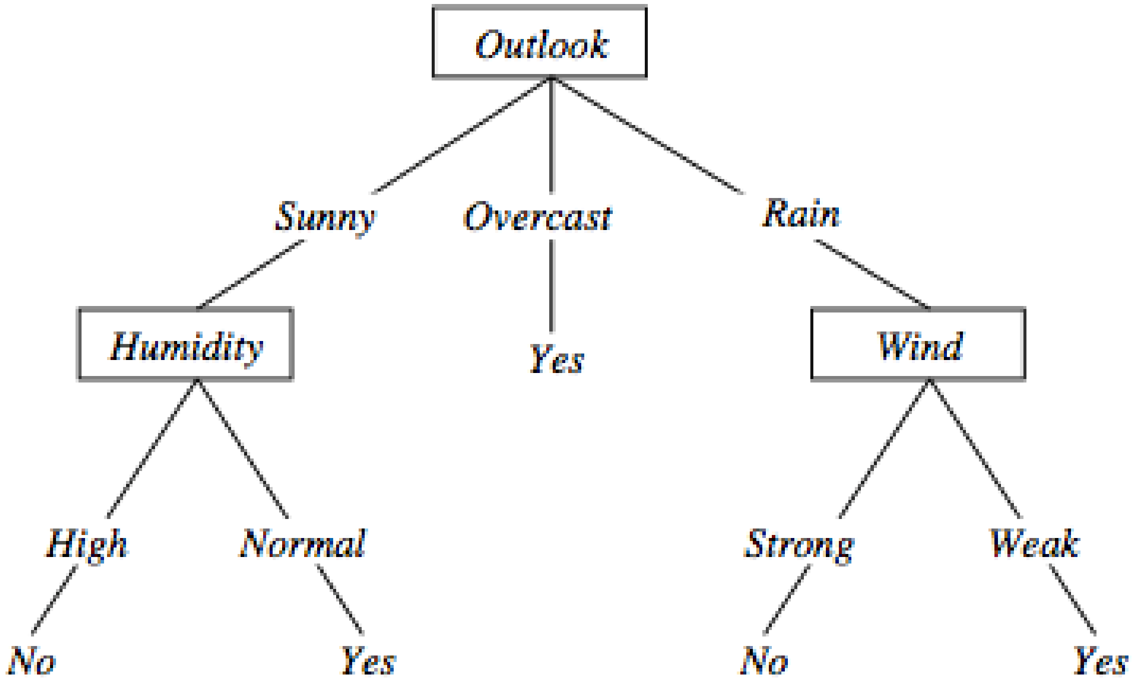
# Reduced-error pruning

- Each node is a candidate for pruning
- *Pruning* consists in removing a sub tree rooted in a node: the node becomes a leaf and is assigned the most common classification
- Nodes are removed only if the resulting tree performs no worse on the validation set.
- Nodes are pruned iteratively: at each iteration the node whose removal most increases accuracy on the validation set is pruned.
- Pruning stops when no pruning increases accuracy

# Rule post-pruning – Additional Read

1. Create the decision tree from the training set
2. Convert the tree into an equivalent set of rules
   - Each path corresponds to a rule
   - Each node along a path corresponds to a pre-condition
   - Each leaf classification to the post-condition
3. Prune (generalize) each rule by removing those preconditions whose removal improves accuracy …
   - … over validation set
4. Sort the rules in estimated order of accuracy, and consider them in sequence when classifying new instances

# Converting to rules




$$(Outlook=Sunny) \wedge (Humidity=High) \Rightarrow (PlayTennis=No)$$

# Rule Post-Pruning – Additional Read

- Convert tree to rules (one for each path from root to a leaf)
- For each antecedent in a rule, remove it if error rate on validation set does not decrease
- Sort final rule set by accuracy

Outlook=sunny ^ humidity=high -> No
Outlook=sunny ^ humidity=normal -> Yes
Outlook=overcast -> Yes
Outlook=rain ^ wind=strong -> No
Outlook=rain ^ wind=weak -> Yes

Compare first rule to:
 Outlook=sunny->No
 Humidity=high->No
Calculate accuracy of 3 rules based on validation set and pick best version.

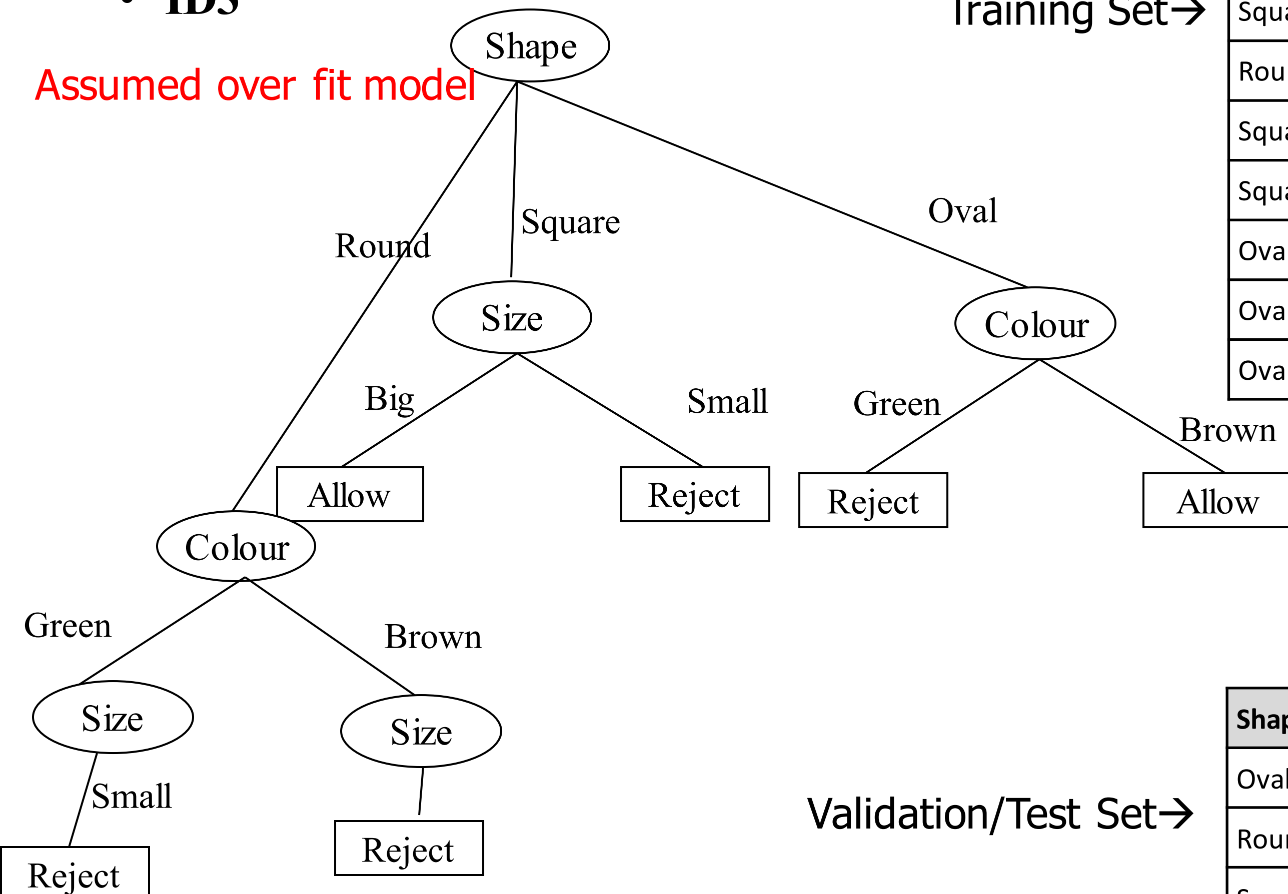
# Why converting to rules?- Additional Read

- Each distinct path produces a different rule: a condition removal may be based on a local (contextual) criterion. Node pruning is global and affects all the rules
- Provides flexibility of not removing entire node
- In rule form, tests are not ordered and there is no book-keeping involved when conditions (nodes) are removed
- Converting to rules improves readability for humans

# Over fitting

- **ID3**

Assumed over fit model




Training Set→

| Shape | Colour | Size | Action |
|---|---|---|---|
| Round | Green | Small | Reject |
| Square | Black | Big | Allow |
| Square | Brown | Big | Allow |
| Round | Brown | Small | Reject |
| Square | Green | Big | Allow |
| Square | Brown | Small | Reject |
| Oval | Green | Big | Reject |
| Oval | Brown | Small | Allow |
| Oval | Green | Small | Reject |

Validation/Test Set→

| Shape | Colour | Size | Action |
|---|---|---|---|
| Oval | Black | Small | Reject |
| Round | Brown | Big | Allow |
| Square | Brown | Big | Allow |
| Oval | Green | Small | Allow |

# How to Address Overfitting

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - General stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - More restrictive conditions (for pre-pruning) :
    - Stop if number of instances is less than some user-specified threshold
    - Stop if class distribution of instances are independent of the available features (e.g., using $\chi^2$ test)
    - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).

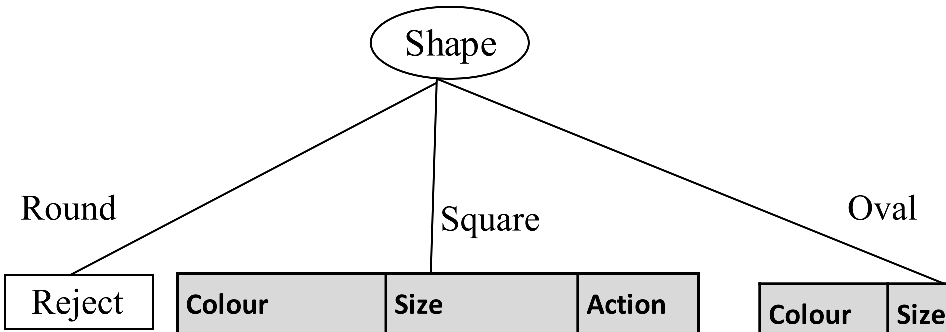
# How to Address Overfitting…

- Post-pruning
  - Grow decision tree to its entirety
  - Trim the nodes of the decision tree in a bottom-up fashion
  - If generalization error(i.e. expected error of the model on previously unseen records) improves after trimming, replace sub-tree by a leaf node.
  - Class label of leaf node is determined from majority class of instances in the sub-tree

# Pre-Pruning – Problem Type 7

| Shape | Colour | Size | Action |
|---|---|---|---|
| Round | Green | Small | Reject |
| Square | Black | Big | Allow |
| Square | Brown | Big | Allow |
| Round | Brown | Small | Reject |
| Square | Green | Big | Allow |
| Square | Brown | Small | Reject |
| Oval | Green | Big | Reject |
| Oval | Brown | Small | Allow |
| Oval | Green | Small | Reject |

- *Threshold: Gain <=0.85*




| Colour | Size | Action |
|---|---|---|
| Black | Big | Allow |
| Brown | Big | Allow |
| Green | Big | Allow |
| Brown | Small | Reject |

Gain =0.8113

| Colour | Size | Action |
|---|---|---|
| Green | Big | Reject |
| Brown | Small | Allow |
| Green | Small | Reject |

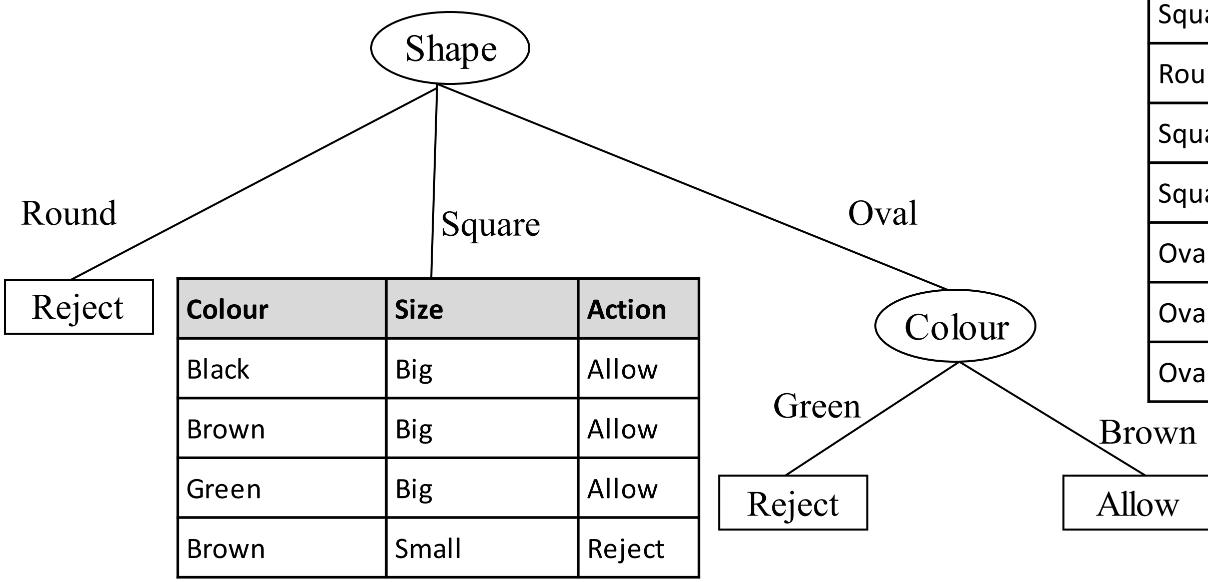
Gain =0.9182

Idea : While construction Prune the nodes whose gain is greater than the predefined threshold

| Shape | Colour | Size | Action |
|---|---|---|---|
| Oval | Black | Small | Reject |
| Round | Brown | Big | Allow |
| Square | Brown | Big | Allow |
| Oval | Green | Small | Allow |

# Pre-Pruning – Problem Type 7

- *Threshold: Gain <=0.85*




| Colour | Size | Action |
|---|---|---|
| Black | Big | Allow |
| Brown | Big | Allow |
| Green | Big | Allow |
| Brown | Small | Reject |

| Shape | Colour | Size | Action |
|---|---|---|---|
| Round | Green | Small | Reject |
| Square | Black | Big | Allow |
| Square | Brown | Big | Allow |
| Round | Brown | Small | Reject |
| Square | Green | Big | Allow |
| Square | Brown | Small | Reject |
| Oval | Green | Big | Reject |
| Oval | Brown | Small | Allow |
| Oval | Green | Small | Reject |

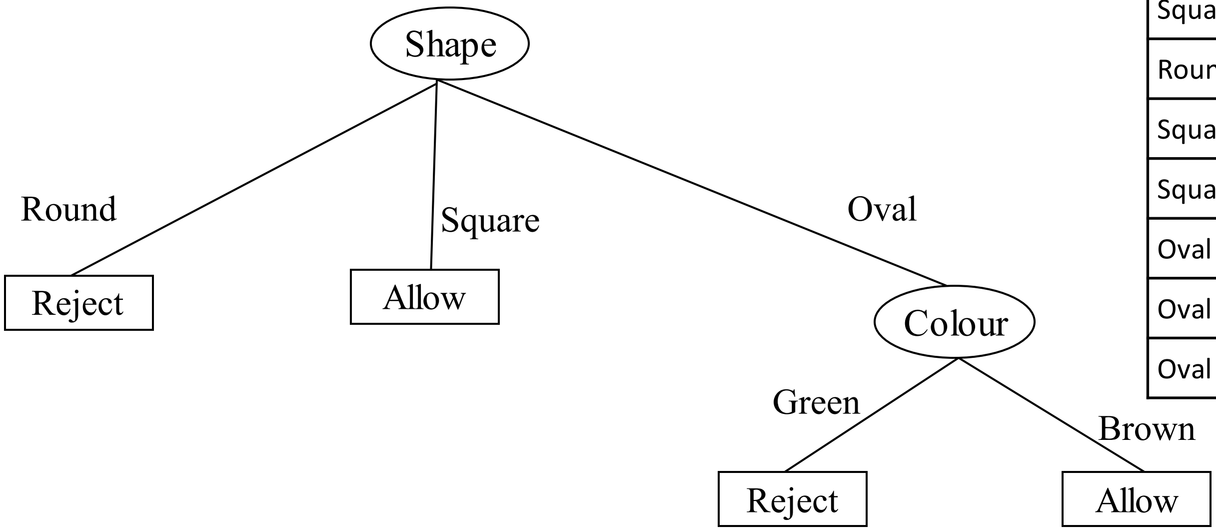
| Shape | Colour | Size | Action |
|---|---|---|---|
| Oval | Black | Small | Reject |
| Round | Brown | Big | Allow |
| Square | Brown | Big | Allow |
| Oval | Green | Small | Allow |

Note:

- DT can estimate the probability of instance's membership to a class

# Pre-Pruning – Problem Type 7

- *Threshold: Gain <=0.85*




| Shape | Colour | Size | Action |
|---|---|---|---|
| Round | Green | Small | Reject |
| Square | Black | Big | Allow |
| Square | Brown | Big | Allow |
| Round | Brown | Small | Reject |
| Square | Green | Big | Allow |
| Square | Brown | Small | Reject |
| Oval | Green | Big | Reject |
| Oval | Brown | Small | Allow |
| Oval | Green | Small | Reject |

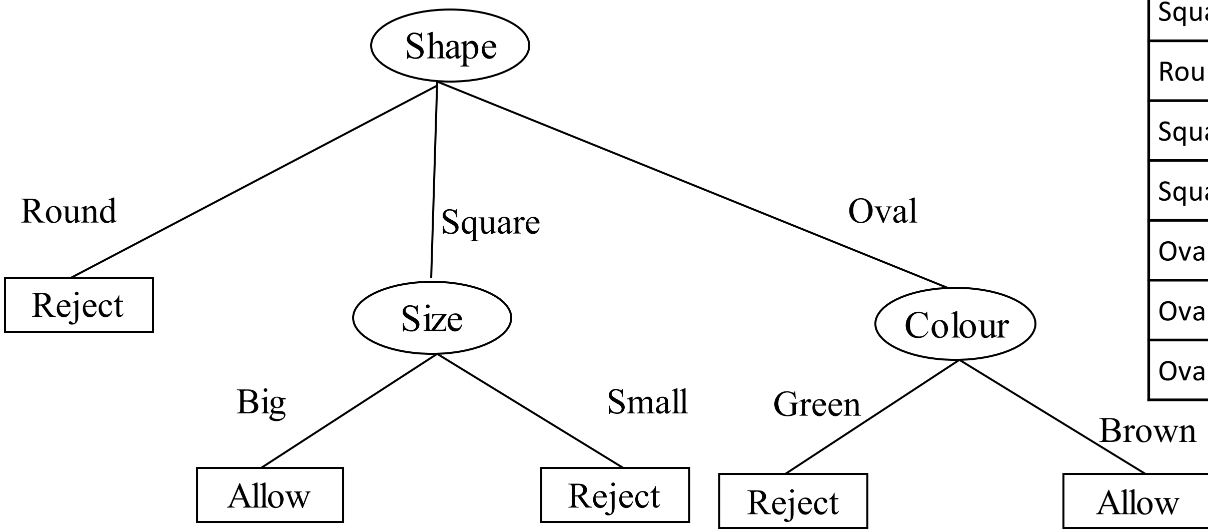
Apply majority voting for converting the pruned subset of data into a class

| Shape | Colour | Size | Action |
|---|---|---|---|
| Oval | Black | Small | Reject |
| Round | Brown | Big | Allow |
| Square | Brown | Big | Allow |
| Oval | Green | Small | Allow |

# Pre-Pruning – Problem Type 8

Assumed model from the output of one intermediate iteration of the decision tree building

| Shape | Colour | Size | Action |
|---|---|---|---|
| Round | Green | Small | Reject |
| Square | Black | Big | Allow |
| Square | Brown | Big | Allow |
| Round | Brown | Small | Reject |
| Square | Green | Big | Allow |
| Square | Brown | Small | Reject |
| Oval | Green | Big | Reject |
| Oval | Brown | Small | Allow |
| Oval | Green | Small | Reject |




Idea :

1. Post construction, scan the tree bottom-up
2. At every decision node
   Retain the attribute node & evaluate it against the prune set (validation set)
   Remove the attribute node & reevaluate it with the same prune set
   If there is a reduction in error, prune the node else retain the node
3. Repeat this in other branches of the tree

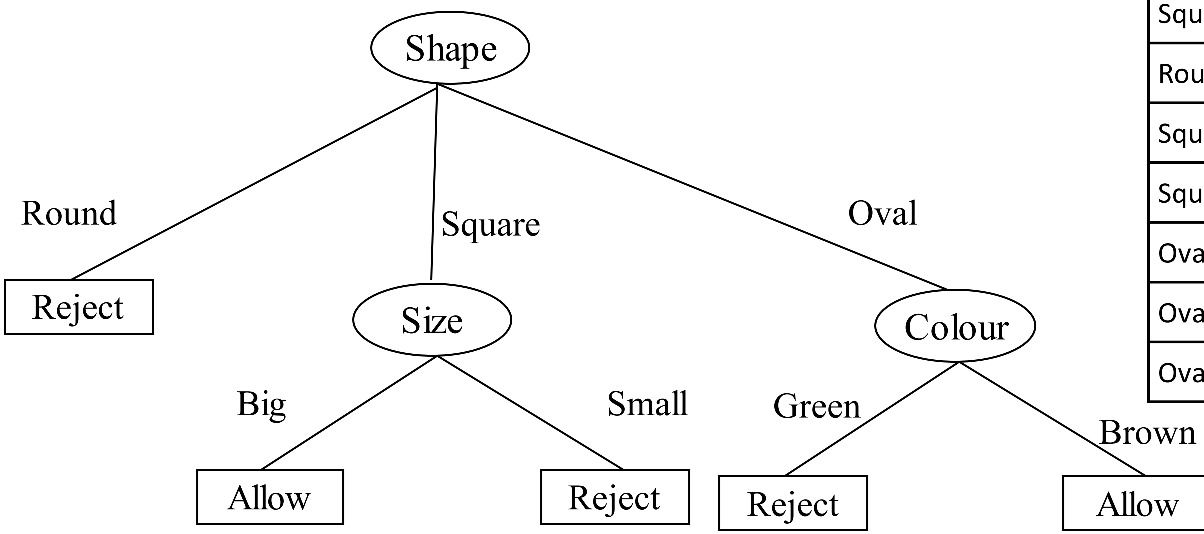
Prune Set

| Shape | Colour | Size | Action |
|---|---|---|---|
| Oval | Black | Small | Reject |
| Round | Brown | Big | Allow |
| Square | Brown | Big | Allow |
| Oval | Green | Small | Allow |

# Pre-Pruning – Problem Type 8

Assumed model from the output of one intermediate iteration of the decision tree building

| Shape | Colour | Size | Action |
|---|---|---|---|
| Round | Green | Small | Reject |
| Square | Black | Big | Allow |
| Square | Brown | Big | Allow |
| Round | Brown | Small | Reject |
| Square | Green | Big | Allow |
| Square | Brown | Small | Reject |
| Oval | Green | Big | Reject |
| Oval | Brown | Small | Allow |
| Oval | Green | Small | Reject |

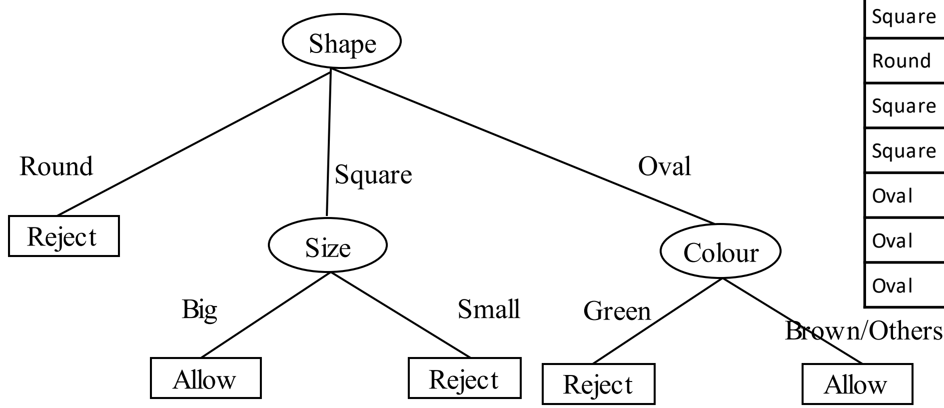



- Error rate is the percentage of tuples misclassified
- Prune set is used to estimate the cost

Prune Set→

| Shape | Colour | Size | Action |
|---|---|---|---|
| Oval | Black | Small | Reject |
| Round | Brown | Big | Allow |
| Square | Brown | Big | Allow |
| Oval | Green | Small | Allow |

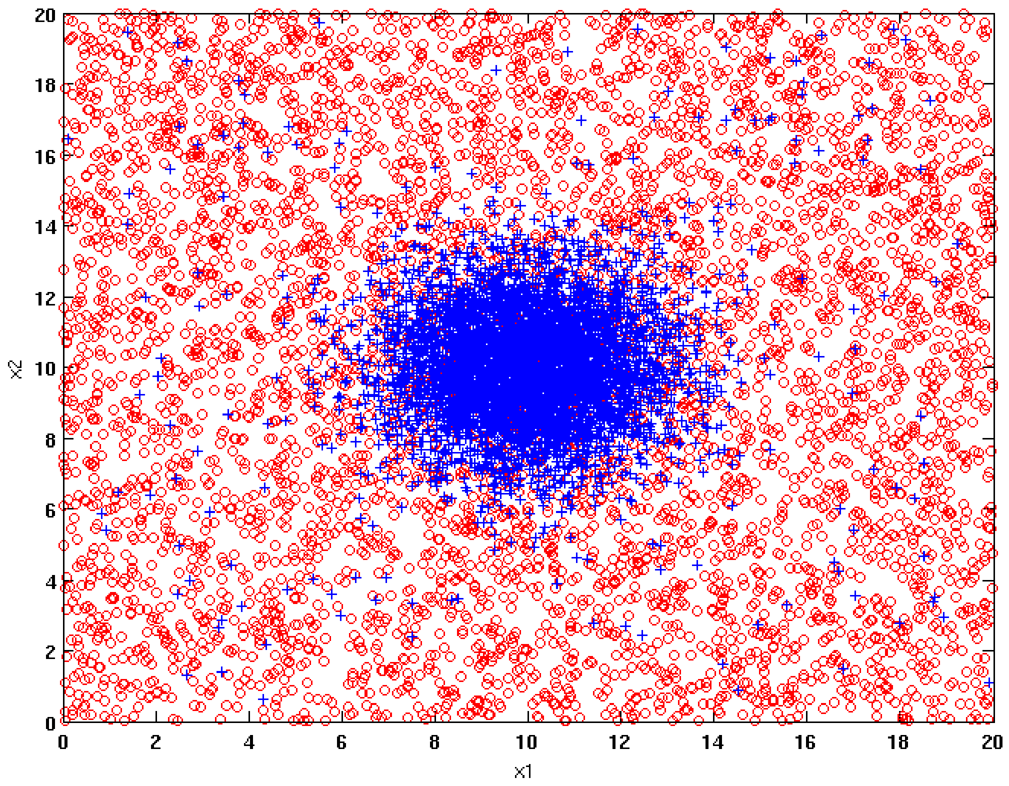
# Pre-Pruning – Problem Type 8




| Shape | Colour | Size | Action |
|---|---|---|---|
| Round | Green | Small | Reject |
| Square | Black | Big | Allow |
| Square | Brown | Big | Allow |
| Round | Brown | Small | Reject |
| Square | Green | Big | Allow |
| Square | Brown | Small | Reject |
| Oval | Green | Big | Reject |
| Oval | Brown | Small | Allow |
| Oval | Green | Small | Reject |

| Prune Size &Predict | Prune Colour & Predict | Above Tree's Prediction | Shape | Colour | Size | Action |
|---|---|---|---|---|---|---|
| Allow | Reject | Allow | Oval | Black | Small | Reject |
| Reject | Reject | Reject | Round | Brown | Big | Allow |
| Allow | Allow | Allow | Square | Brown | Big | Allow |
| Reject | Reject | Reject | Oval | Green | Small | Allow |

# Example Data Set




**Two class problem:**

**+ : 5200 instances**

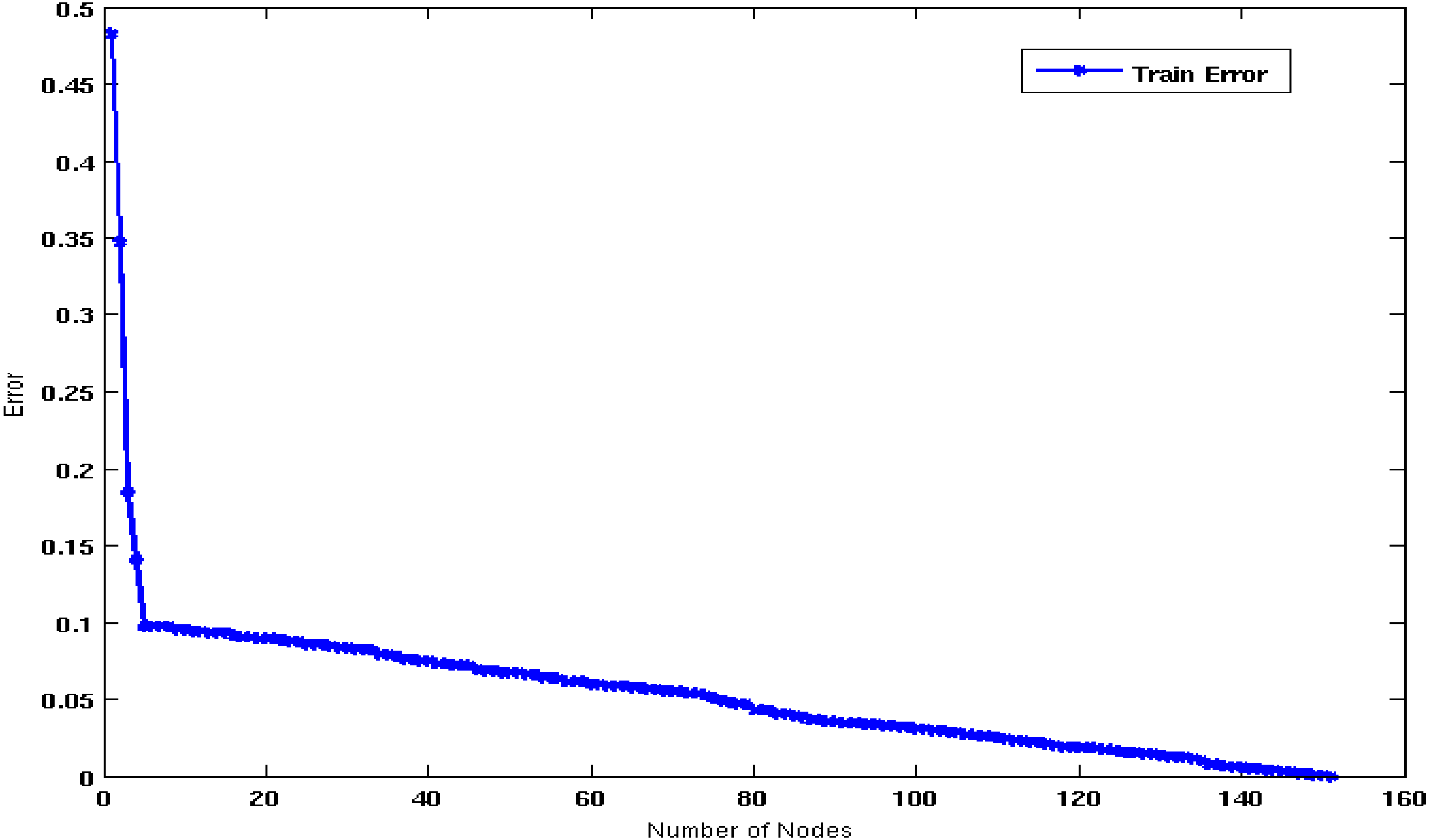
- **5000 instances generated from a Gaussian centered at (10,10)**
- **200 noisy instances added**

**o : 5200 instances**

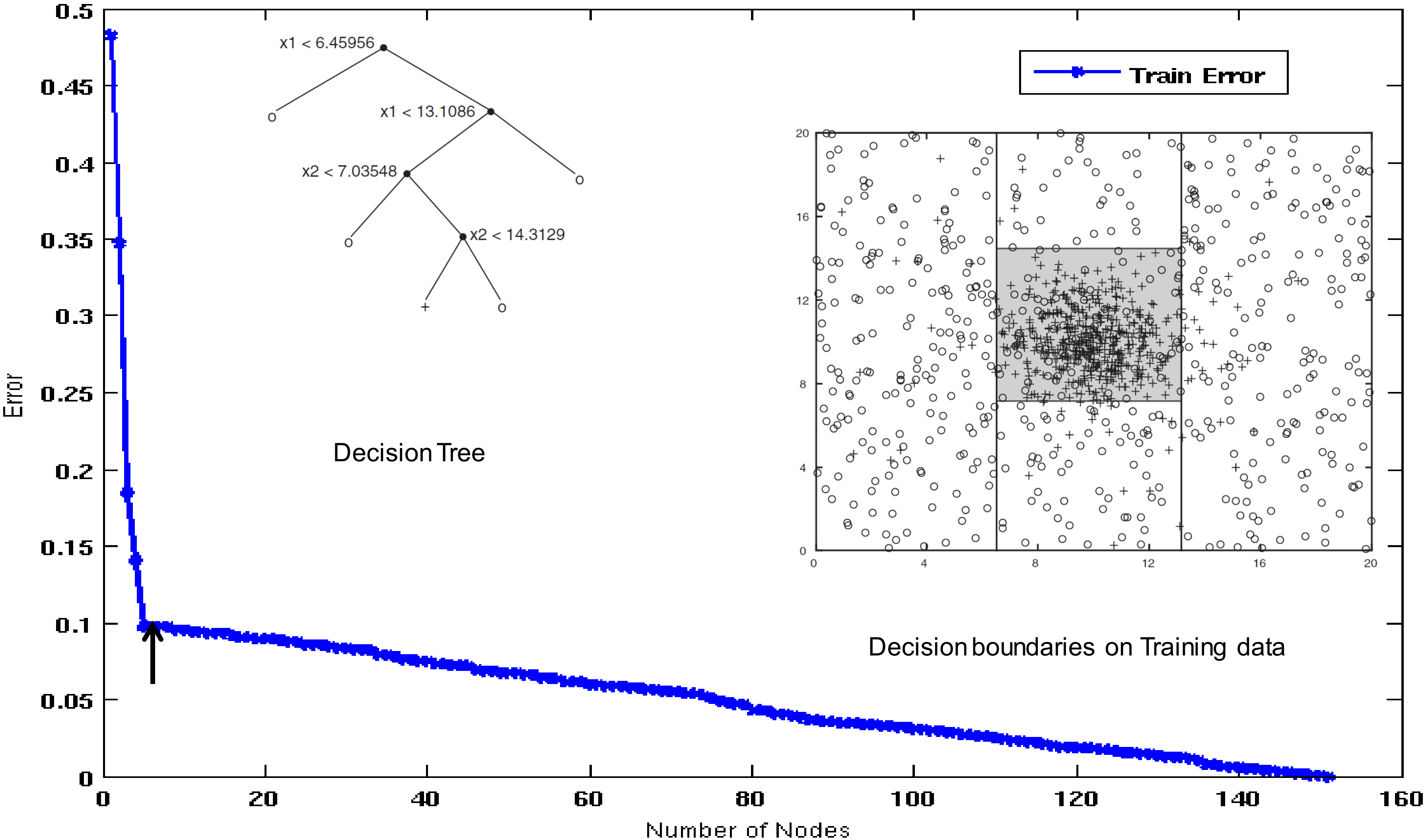
- **Generated from a uniform distribution**

**10 % of the data used for training and 90% of the data used for testing**

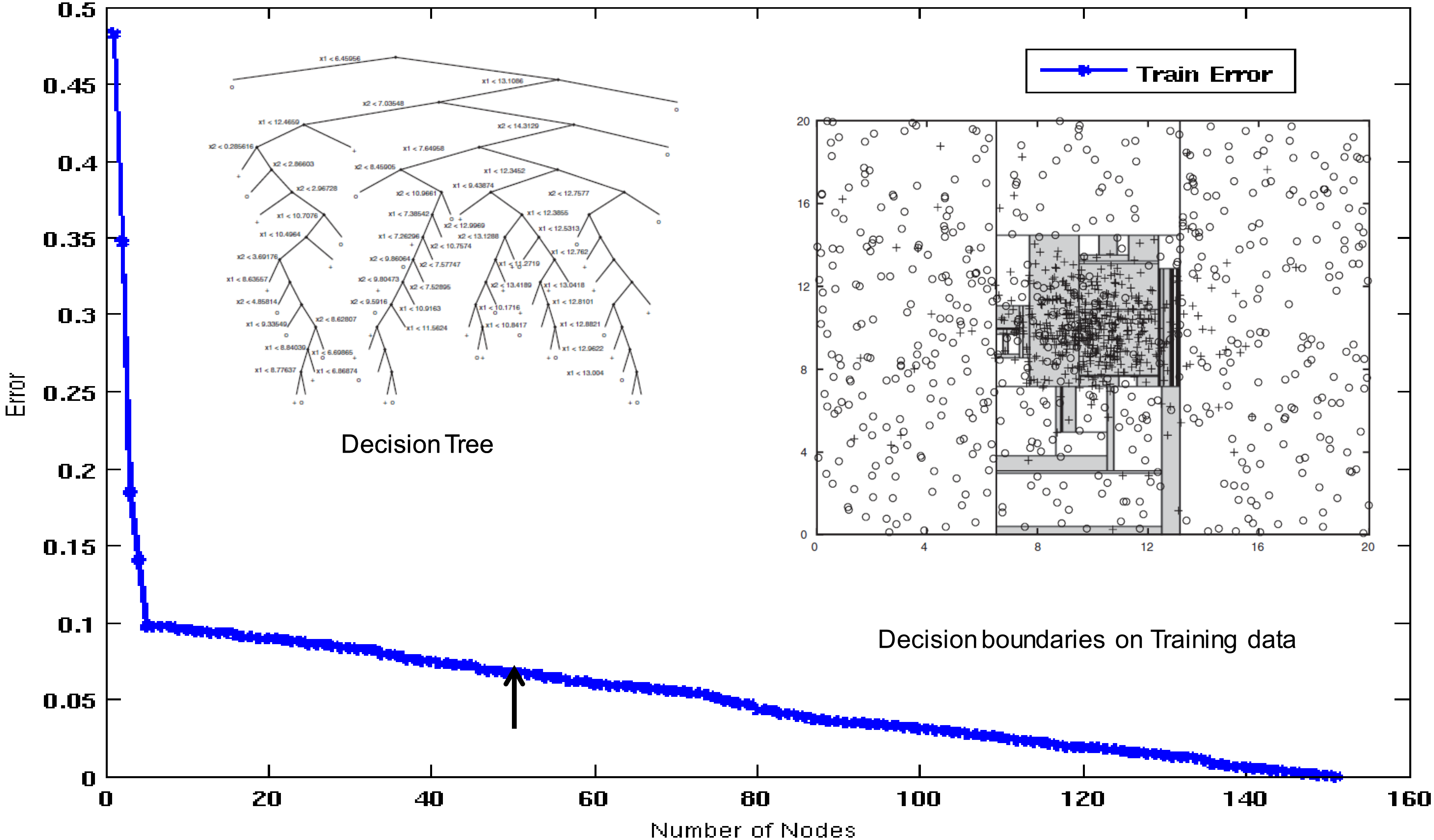
# Increasing number of nodes in Decision Trees



# Decision Tree with 4 nodes

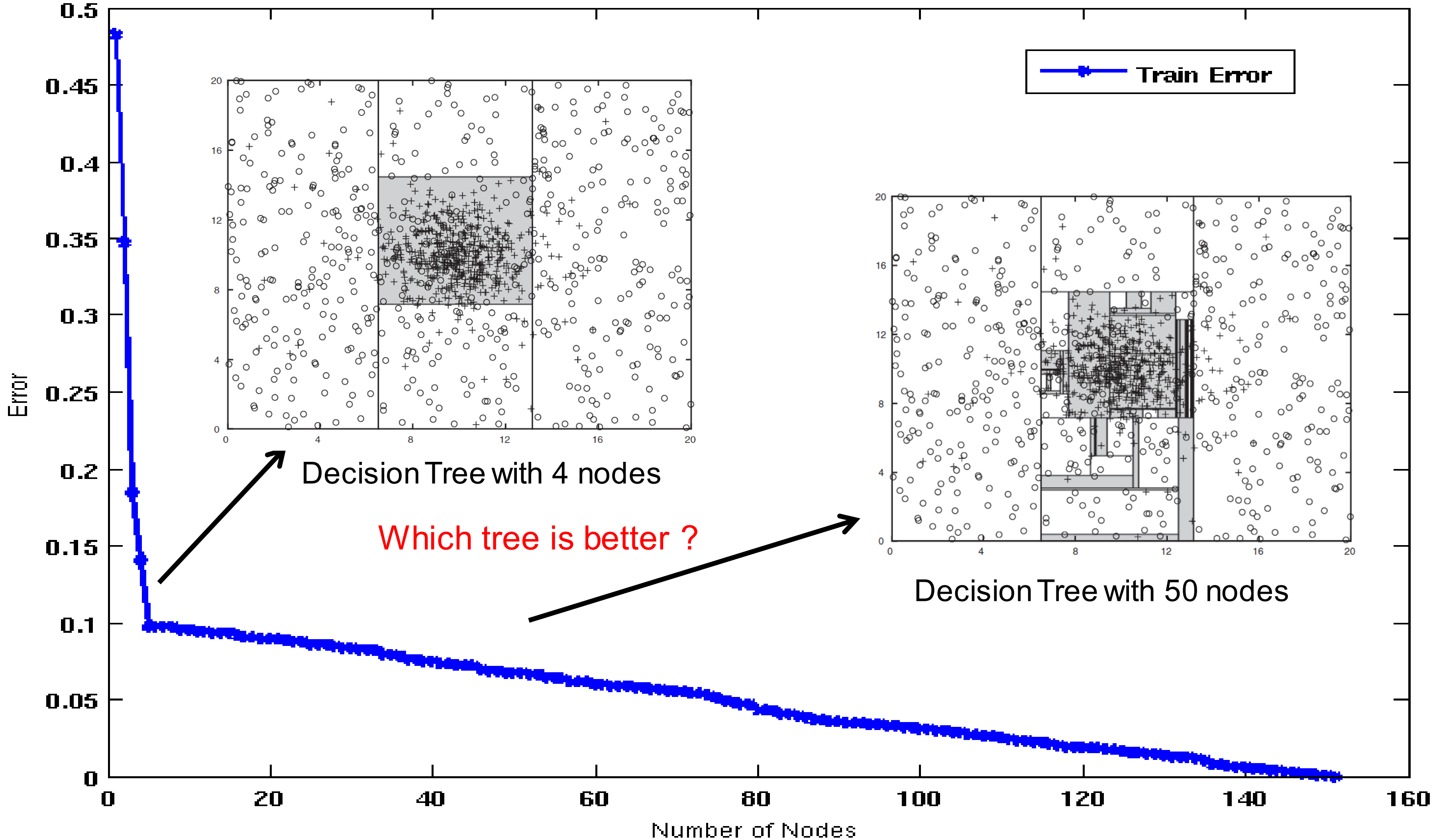


# Decision Tree with 50 nodes

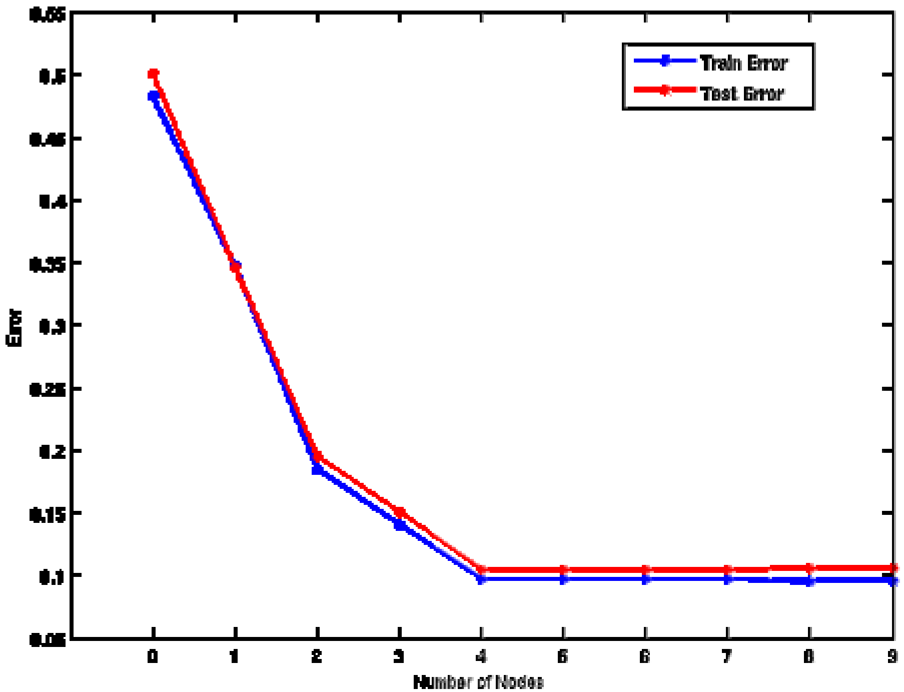


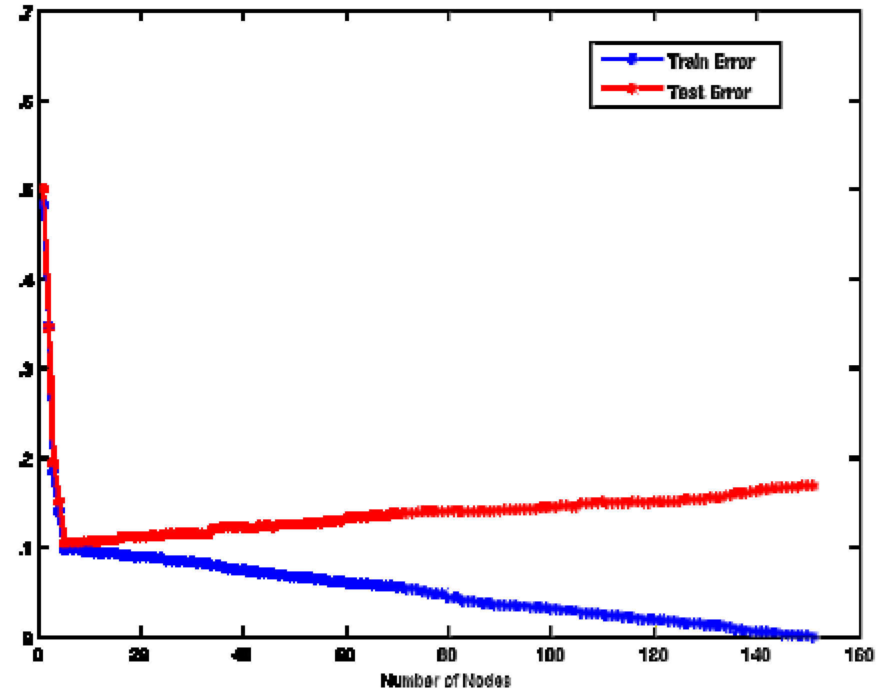
# Which tree is better?

## Problem Type 9 – Model Comparison



# Model Overfitting




Underfitting: when model is too simple, both training and test errors are large

Overfitting: when model is too complex, training error is small but test error is large

# Issues in Decision Tree Learning - Example

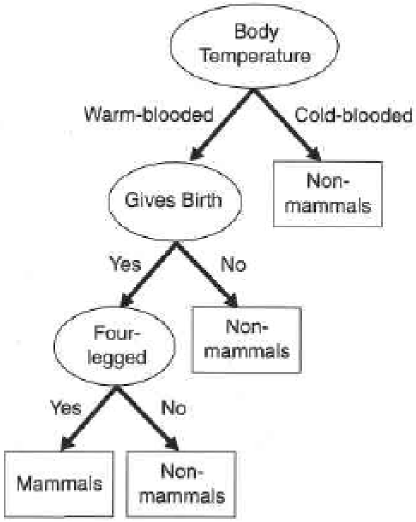
- Overfitting due to the presence of noise

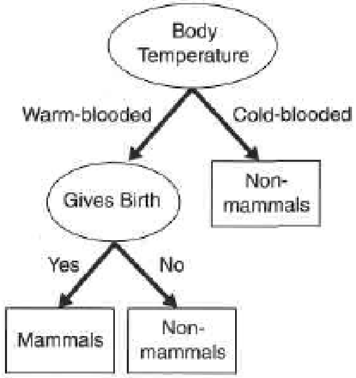
**Training set**

| Name | Body Temperature | Gives Birth | Four-legged | Hibernates | Class Label |
|---|---|---|---|---|---|
| porcupine | warm-blooded | yes | yes | yes | yes |
| cat | warm-blooded | yes | yes | no | yes |
| bat | warm-blooded | yes | no | yes | no* |
| whale | warm-blooded | yes | no | no | no* |
| salamander | cold-blooded | no | yes | yes | no |
| komodo dragon | cold-blooded | no | yes | no | no |
| python | cold-blooded | no | no | yes | no |
| salmon | cold-blooded | no | no | no | no |
| eagle | warm-blooded | no | no | no | no |
| guppy | cold-blooded | yes | no | no | no |

**Test set**

| Name | Body Temperature | Gives Birth | Four-legged | Hibernates | Class Label |
|---|---|---|---|---|---|
| human | warm-blooded | yes | no | no | yes |
| pigeon | warm-blooded | no | no | no | no |
| elephant | warm-blooded | yes | yes | no | yes |
| leopard shark | cold-blooded | yes | no | no | no |
| turtle | cold-blooded | no | yes | no | no |
| penguin | cold-blooded | no | no | no | no |
| eel | cold-blooded | no | no | no | no |
| dolphin | warm-blooded | yes | no | no | yes |
| spiny anteater | warm-blooded | no | yes | yes | yes |
| gila monster | cold-blooded | no | yes | yes | no |



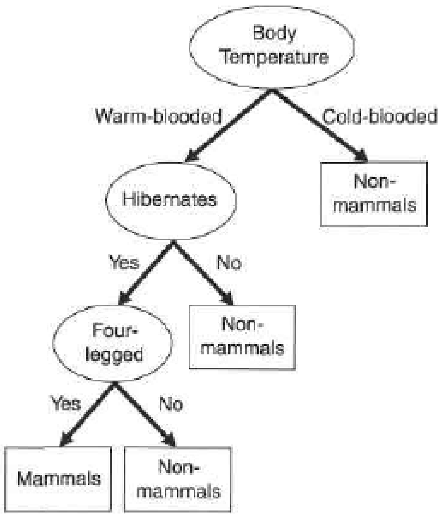

What are the error rates of the decision trees on the test set?

# Issues in Decision Tree Learning - Example

- Overfitting due to the lack of representative samples

**Training set**

| Name | Body Temperature | Gives Birth | Four-legged | Hibernates | Class Label |
|---|---|---|---|---|---|
| salamander | cold-blooded | no | yes | yes | no |
| guppy | cold-blooded | yes | no | no | no |
| eagle | warm-blooded | no | no | no | no |
| poorwill | warm-blooded | no | no | yes | no |
| platypus | warm-blooded | no | yes | yes | yes |



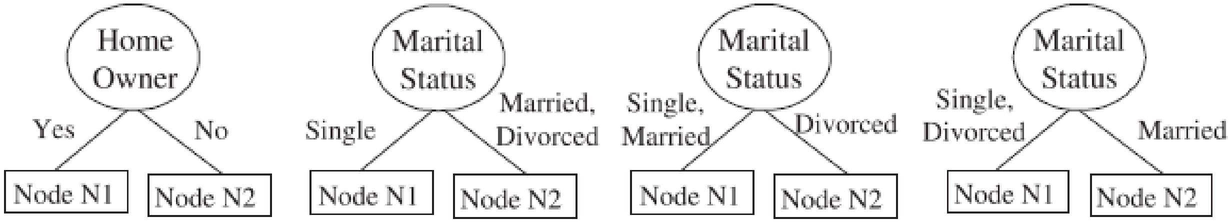

**Test set**

| Name | Body Temperature | Gives Birth | Four-legged | Hibernates | Class Label |
|---|---|---|---|---|---|
| human | warm-blooded | yes | no | no | yes |
| pigeon | warm-blooded | no | no | no | no |
| elephant | warm-blooded | yes | yes | no | yes |
| leopard shark | cold-blooded | yes | no | no | no |
| turtle | cold-blooded | no | yes | no | no |
| penguin | cold-blooded | no | no | no | no |
| eel | cold-blooded | no | no | no | no |
| dolphin | warm-blooded | yes | no | no | yes |
| spiny anteater | warm-blooded | no | yes | yes | yes |
| gila monster | cold-blooded | no | yes | yes | no |

What is the error rate of the decision tree on the test set?

# Alternative to Entropy:
# Gini Index : Binary Splits

| ID | Home Owner | Marital Status | Annual Income | Defaulted? |
|---|---|---|---|---|
| 1 | Yes | Single | 125000 | No |
| 2 | No | Married | 100000 | No |
| 3 | No | Single | 70000 | No |
| 4 | Yes | Married | 120000 | No |
| 5 | No | Divorced | 95000 | Yes |
| 6 | No | Single | 60000 | No |
| 7 | Yes | Divorced | 220000 | No |
| 8 | No | Single | 85000 | Yes |
| 9 | No | Married | 75000 | No |
| 10 | No | Single | 90000 | Yes |



# Alternative to Entropy:
# Gini Index : Binary Splits



**Gini Index:Parent**

$$Gini\ Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

| ID | Home Owner | Marital Status | Annual Income | Defaulted? |
|---|---|---|---|---|
| 1 | Yes | Single | 125000 | No |
| 2 | No | Married | 100000 | No |
| 3 | No | Single | 70000 | No |
| 4 | Yes | Married | 120000 | No |
| 5 | No | Divorced | 95000 | Yes |
| 6 | No | Single | 60000 | No |
| 7 | Yes | Divorced | 220000 | No |
| 8 | No | Single | 85000 | Yes |
| 9 | No | Married | 75000 | No |
| 10 | No | Single | 90000 | Yes |

$$1 - \left(\frac{3}{10}\right)^2 - \left(\frac{7}{10}\right)^2 = 0.420.$$

| | Parent |
|---|---|
| No | 7 |
| Yes | 3 |
| Gini = 0.420 | |

# Alternative to Entropy:
# Gini Index : Binary Splits

## Splitting Attribute: HomeOwner

$$Gini\ Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

| ID | Home Owner | Marital Status | Annual Income | Defaulted? |
|---|---|---|---|---|
| 1 | Yes | Single | 125000 | No |
| 2 | No | Married | 100000 | No |
| 3 | No | Single | 70000 | No |
| 4 | Yes | Married | 120000 | No |
| 5 | No | Divorced | 95000 | Yes |
| 6 | No | Single | 60000 | No |
| 7 | Yes | Divorced | 220000 | No |
| 8 | No | Single | 85000 | Yes |
| 9 | No | Married | 75000 | No |
| 10 | No | Single | 90000 | Yes |

***Gini index for the child nodes***

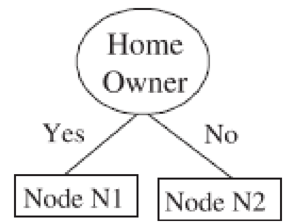
*N1 = 1- $(0/3)^2$ - $(3/3)^2$ = 0*

*N2 =1- $(3/7)^2$ - $(4/7)^2$ = 0.490*

***Weighted Gini Index for children***

$$I(\text{children}) = \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j),$$

$$(3/10) \times 0 + (7/10) \times 0.490 = 0.343,$$



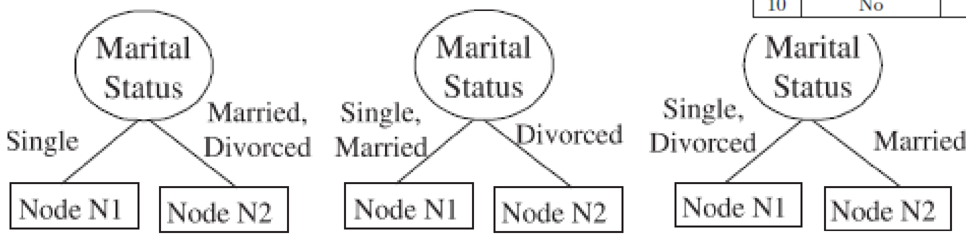

| | N1 | N2 |
|---|---|---|
| No | 3 | 4 |
| Yes | 0 | 3 |
| Gini= 0.343 | | |

**The gain using Home Owner as splitting attribute is 0.420 - 0.343 =0.077**

# Alternative to Entropy:
# Gini Index : Binary Splits

| ID | Home Owner | Marital Status | Annual Income | Defaulted? |
|---|---|---|---|---|
| 1 | Yes | Single | 125000 | No |
| 2 | No | Married | 100000 | No |
| 3 | No | Single | 70000 | No |
| 4 | Yes | Married | 120000 | No |
| 5 | No | Divorced | 95000 | Yes |
| 6 | No | Single | 60000 | No |
| 7 | Yes | Divorced | 220000 | No |
| 8 | No | Single | 85000 | Yes |
| 9 | No | Married | 75000 | No |
| 10 | No | Single | 90000 | Yes |

## Splitting Attribute: MaritalStatus




| | N1 | N2 |
|---|---|---|
| No | 3 | 4 |
| Yes | 2 | 1 |
| Gini= 0.400 | | |

| | N1 | N2 |
|---|---|---|
| No | 6 | 1 |
| Yes | 2 | 1 |
| Gini= 0.400 | | |

| | N1 | N2 |
|---|---|---|
| No | 4 | 3 |
| Yes | 3 | 0 |
| Gini= 0.343 | | |

**Based on these results, Home Owner and the last binary split using Marital Status are clearly the best candidates, since they both produce the lowest weighted average Gini index**

# Problems with information gain

- Natural bias of information gain: it favors attributes with many possible values.
- Consider the attribute *Date* in the *PlayTennis* example.
  - *Date* would have the highest information gain since it perfectly separates the training data.
  - It would be selected at the root resulting in a very broad tree
  - Very good on the training, this tree would perform poorly in predicting unknown instances. Overfitting.
- The problem is that the partition is too specific, too many small classes are generated.
- We need to look at alternative measures.

# An alternative measure: gain ratio

$$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

- $S_i$ are the sets obtained by partitioning on value $i$ of $A$
- *SplitInformation* measures the entropy of $S$ with respect to the values of $A$. The more uniformly dispersed the data the higher it is.

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

- *GainRatio* penalizes attributes that split examples in many small classes such as *Date*. Let $|S|=n$, *Date* splits examples in $n$ classes
  - $SplitInformation(S, Date) = -[(1/n \log_2 1/n) + \ldots + (1/n \log_2 1/n)] = -\log_2 1/n = \log_2 n$
- Compare with $A$, which splits data in two even classes:
  - $SplitInformation(S, A) = -[(1/2 \log_2 1/2) + (1/2 \log_2 1/2)] = -[-1/2 - 1/2] = 1$

# Limitation of Information Gain:
## Ways to tackle

SplitInformation measures the entropy of S with respect to the values of A. The more uniformly dispersed the data the higher it is.

$$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

$SplitInformation(S, customerID) = -[(1/n \, log_2 \, 1/n) + \ldots + (1/n \, log_2 \, 1/n)] = -log_2 1/n = log_2 n$

GainRatio penalizes attributes that split examples in many small classes such as ID by incorporating split information.

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

| Customer ID | Gender | Car Type | Shirt Size | Class |
|---|---|---|---|---|
| 1 | M | Family | Small | C0 |
| 2 | M | Sports | Medium | C0 |
| 3 | M | Sports | Medium | C0 |
| 4 | M | Sports | Large | C0 |
| 5 | M | Sports | Extra Large | C0 |
| 6 | M | Sports | Extra Large | C0 |
| 7 | F | Sports | Small | C0 |
| 8 | F | Sports | Small | C0 |
| 9 | F | Sports | Medium | C0 |
| 10 | F | Luxury | Large | C0 |
| 11 | M | Family | Large | C1 |
| 12 | M | Family | Extra Large | C1 |
| 13 | M | Family | Medium | C1 |
| 14 | M | Luxury | Extra Large | C1 |
| 15 | F | Luxury | Small | C1 |
| 16 | F | Luxury | Small | C1 |
| 17 | F | Luxury | Medium | C1 |
| 18 | F | Luxury | Medium | C1 |
| 19 | F | Luxury | Medium | C1 |
| 20 | F | Luxury | Large | C1 |

Select the best attribute test condition among the following three attributes: **Gender, Car Type, and Customer ID**

# Limitation of Information Gain: Ways to tackle

**Problem Type 10**

$$\text{Entropy(parent)} = -\frac{10}{20}\log_2\frac{10}{20} - \frac{10}{20}\log_2\frac{10}{20} = 1.$$

If `Gender` is used as attribute test condition:

$$\text{Entropy(children)} = \frac{10}{20}\left[-\frac{6}{10}\log_2\frac{6}{10} - \frac{4}{10}\log_2\frac{4}{10}\right] \times 2 = 0.971$$

$$\text{Gain Ratio} = \frac{1 - 0.971}{-\frac{10}{20}\log_2\frac{10}{20} - \frac{10}{20}\log_2\frac{10}{20}} = \frac{0.029}{1} = 0.029$$

If `Car Type` is used as attribute test condition:

$$\text{Entropy(children)} = \frac{4}{20}\left[-\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4}\right] + \frac{8}{20} \times 0 + \frac{8}{20}\left[-\frac{1}{8}\log_2\frac{1}{8} - \frac{7}{8}\log_2\frac{7}{8}\right] = 0.380$$

$$\text{Gain Ratio} = \frac{1 - 0.380}{-\frac{4}{20}\log_2\frac{4}{20} - \frac{8}{20}\log_2\frac{8}{20} - \frac{8}{20}\log_2\frac{8}{20}} = \frac{0.620}{1.52} = 0.41$$

| Customer ID | Gender | Car Type | Shirt Size | Class |
|---|---|---|---|---|
| 1 | M | Family | Small | C0 |
| 2 | M | Sports | Medium | C0 |
| 3 | M | Sports | Medium | C0 |
| 4 | M | Sports | Large | C0 |
| 5 | M | Sports | Extra Large | C0 |
| 6 | M | Sports | Extra Large | C0 |
| 7 | F | Sports | Small | C0 |
| 8 | F | Sports | Small | C0 |
| 9 | F | Sports | Medium | C0 |
| 10 | F | Luxury | Large | C0 |
| 11 | M | Family | Large | C1 |
| 12 | M | Family | Extra Large | C1 |
| 13 | M | Family | Medium | C1 |
| 14 | M | Luxury | Extra Large | C1 |
| 15 | F | Luxury | Small | C1 |
| 16 | F | Luxury | Small | C1 |
| 17 | F | Luxury | Medium | C1 |
| 18 | F | Luxury | Medium | C1 |
| 19 | F | Luxury | Medium | C1 |
| 20 | F | Luxury | Large | C1 |

# Limitation of Information Gain: Ways to tackle

## Problem Type 10

Finally, if `Customer ID` is used as attribute test condition:

$$\text{Entropy(children)} = \frac{1}{20}\left[-\frac{1}{1}\log_2\frac{1}{1} - \frac{0}{1}\log_2\frac{0}{1}\right] \times 20 = 0$$

$$\text{Gain Ratio} = \frac{1-0}{-\frac{1}{20}\log_2\frac{1}{20} \times 20} = \frac{1}{4.32} = 0.23$$

| Customer ID | Gender | Car Type | Shirt Size | Class |
|---|---|---|---|---|
| 1 | M | Family | Small | C0 |
| 2 | M | Sports | Medium | C0 |
| 3 | M | Sports | Medium | C0 |
| 4 | M | Sports | Large | C0 |
| 5 | M | Sports | Extra Large | C0 |
| 6 | M | Sports | Extra Large | C0 |
| 7 | F | Sports | Small | C0 |
| 8 | F | Sports | Small | C0 |
| 9 | F | Sports | Medium | C0 |
| 10 | F | Luxury | Large | C0 |
| 11 | M | Family | Large | C1 |
| 12 | M | Family | Extra Large | C1 |
| 13 | M | Family | Medium | C1 |
| 14 | M | Luxury | Extra Large | C1 |
| 15 | F | Luxury | Small | C1 |
| 16 | F | Luxury | Small | C1 |
| 17 | F | Luxury | Medium | C1 |
| 18 | F | Luxury | Medium | C1 |
| 19 | F | Luxury | Medium | C1 |
| 20 | F | Luxury | Large | C1 |

Inference : Thus, even though Customer ID has the highest information gain, its gain ratio is lower than Car Type since it produces a larger number of splits

# Minimum Description Length Principle

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in H} P(D|h)P(h) \\ &= \arg\max_{h \in H} \log_2 P(D|h) + \log_2 P(h) \\ &= \arg\min_{h \in H} -\log_2 P(D|h) - \log_2 P(h) \quad (1) \end{aligned}$$

Interesting fact from information theory:

The optimal (shortest expected coding length) code for an event with probability $p$ is $-\log_2 p$ bits.

So interpret (1):

- $L_{C1}(h) = length(h) = -\log_2 P(h)$
- $L_{C2}(D|h) = length(misclassifications) = -\log_2 P(D|h)$

$\rightarrow$ prefer the hypothesis that minimizes

$$length(h) + length(misclassifications)$$

# Minimum Description Length Principle

- MDL principle provides a way of trading off hypothesis complexity for the number of errors committed by the hypothesis.
- May select a shorter hypothesis that makes a few errors over a longer hypothesis that perfectly classifies the training data.
- Provides one method for dealing with the issue of overfitting the data.

# Minimum Description Length Principle

**Occam's razor: prefer the shortest hypothesis**

MDL: prefer the hypothesis *h* that minimizes

$$h_{MDL} = \underset{h \in H}{\operatorname{argmin}}\, L_{C_1}(h) + L_{C_2}(D|h)$$

where $L_C(x)$ is the description length of $x$ under encoding $C$

Example: H = decision trees hypothesis, D = training data labels

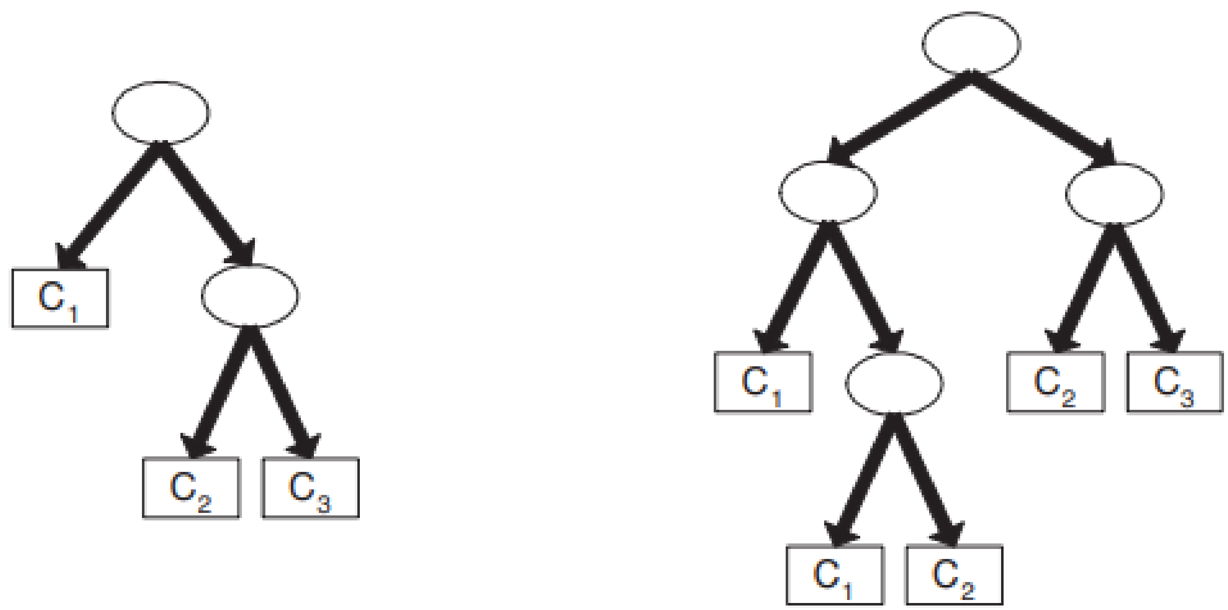
- $L_{C1}(h)$ is # bits to describe tree $h$
- $L_{C2}(D|h)$ is # bits to describe $D$ given $h$
  - Note $L_{C2}(D|h) = 0$ if examples classified perfectly by $h$. Need only describe exceptions
- Hence $h_{MDL}$ trades off tree size for training errors

# Minimum Description Length Principle

**Problem Type 11**

Consider the decision trees shown in Figure . Assume they are generated from a data set that contains 16 binary attributes and 3 classes, $C_1$, $C_2$, and $C_3$. Compute the total description length of each decision tree according to the minimum description length principle.

Which decision tree is better, according to the MDL principle?




(a) Decision tree with 7 errors (b) Decision tree with 4 errors

# Minimum Description Length Principle

## Problem Type 11

**Answer:**

Because there are 16 attributes, the cost for each internal node in the decision tree is:

$$\log_2(m) = \log_2(16) = 4$$

Furthermore, because there are 3 classes, the cost for each leaf node is:

$$\lceil \log_2(k) \rceil = \lceil \log_2(3) \rceil = 2$$

The cost for each misclassification error is $\log_2(n)$.

The overall cost for the decision tree (a) is $2\times4+3\times2+7\times\log_2 n = 14+7\log_2 n$ and the overall cost for the decision tree (b) is $4\times4+5\times2+4\times5 = 26+4\log_2 n$. According to the MDL principle, tree (a) is better than (b) if $n < 16$ and is worse than (b) if $n > 16$.

# Handling missing values training data

- How to cope with the problem that the value of some attribute may be missing?
- The strategy: use other examples to guess attribute
  1. Assign the value that is most common among the training examples at the node
  2. Assign a probability to each value, based on frequencies, and assign values to missing attribute, according to this probability distribution

# Handling missing values training data

**Problem Type 12**

Consider the dataset given below where $A$ and $B$ are attributes which can take the values 0 and 1, and $Y$ is the classification. The values marked "*" represent data values that are corrupted. It is known that during the construction of a decision tree to represent the clean dataset (i.e one without any "*"), the attribute $B$ was chosen at the root instead of attribute $A$ using information gain. Is this information enough to guess the value of the bit that must replace "*"? Give a detailed justification for your answer.

| | Y=Yes | Y=No |
|---|---|---|
| A=0 | 2 | 1 |
| A=1 | 1 | 2 |

| If *=0 | Y=Yes | Y=No |
|---|---|---|
| B=0 | 1 | 1 |
| B=1 | 3 | 1 |

| If *=1 | Y=Yes | Y=No |
|---|---|---|
| B=0 | 0 | 1 |
| B=1 | 3 | 2 |

| A | B | Y |
|---|---|---|
| 1 | 0 | no |
| 1 | 1 | no |
| 0 | * | no |
| 0 | 1 | yes |
| 0 | 1 | yes |
| 1 | 1 | yes |

$$InfGain(A) = Entropy(S) - \frac{|S_{A=0}|}{|S|}Entropy(S_{A=0}) - \frac{|S_{A=1}|}{|S|}Entropy(S_{A=1}) = -\left(\frac{3}{6}\log_2\frac{3}{6} + \frac{3}{6}\log_2\frac{3}{6}\right) - \frac{3}{6}\left(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}\right) - \frac{3}{6}\left(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}\right) = -\left(\frac{3}{6}\log_2\frac{3}{6} + \frac{3}{6}\log_2\frac{3}{6}\right) + \frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3} = 1 - 0$$

# Handling missing values training data



**Problem Type 12**

Consider the dataset given below where $A$ and $B$ are attributes which can take the values 0 and 1, and $Y$ is the classification. The values marked "*" represent data values that are corrupted. It is known that during the construction of a decision tree to represent the clean dataset (i.e one without any "*"), the attribute $B$ was chosen at the root instead of attribute $A$ using information gain. Is this information enough to guess the value of the bit that must replace "*"? Give a detailed justification for your answer.

| | Y=Yes | Y=No |
|---|---|---|
| A=0 | 2 | 1 |
| A=1 | 1 | 2 |

| If *=0 | Y=Yes | Y=No |
|---|---|---|
| B=0 | 1 | 1 |
| B=1 | 3 | 1 |

| If *=1 | Y=Yes | Y=No |
|---|---|---|
| B=0 | 0 | 1 |
| B=1 | 3 | 2 |

| A | B | Y |
|---|---|---|
| 1 | 0 | no |
| 1 | 1 | no |
| 0 | * | no |
| 0 | 1 | yes |
| 0 | 1 | yes |
| 1 | 1 | yes |

If we assume *=**1**, then we $InfGain(B, *=1) = Entropy(S) - \frac{|S_{B=0}|}{|S|}Entropy(S_{B=0}) - \frac{|S_{B=1}|}{|S|}$

$$Entropy(S_{B=1}) = -\left(\tfrac{3}{6}\log_2\tfrac{3}{6} + \tfrac{3}{6}\log_2\tfrac{3}{6}\right) - \tfrac{1}{6}(-1\log_2 1 - 0\log_2 0) - \tfrac{5}{6}\left(-\tfrac{3}{5}\log_2\tfrac{3}{5} - \tfrac{2}{5}\log_2\tfrac{2}{5}\right)$$

$$= 1 - 0.809 = 0.191$$

If we assume *=0, then we have $InfGain(B, *=0) = Entropy(S) - \frac{|S_{B=0}|}{|S|}Entropy(S_{B=0}) - \frac{|S_{B=1}|}{|S|}$

$$Entropy(S_{B=1}) = -\left(\tfrac{3}{6}\log_2\tfrac{3}{6} + \tfrac{3}{6}\log_2\tfrac{3}{6}\right) - \tfrac{2}{6}(-1\log_2 1 - 0\log_2 0) - \tfrac{4}{6}\left(-\tfrac{1}{4}\log_2\tfrac{1}{4} - \tfrac{3}{4}\log_2\tfrac{3}{4}\right) = 1 - 0.54 = 0.46$$

# Practice Exercises (for Students)

1. Which of the below discretization best suits to transform the attribute "Annual Income" for decision tree construction? Use entropy as decision criteria.
   - Annual Income ( <=85k, >85k & <=200k, >200k )
   - Annual Income ( <=90k, >90k )

2. If binary split is recommended for attribute "Marital Status" which of the combination of splits best fits? Justify your comment using entropy.

3. Use the results of part 1) & part 2) and Build a decision tree classifier using ID3 algorithm ie., Information gain and entropy measures. Grow the complete decision tree.
4. Use the test data to post prune the built tree. Try to prune atleast two internal nodes and choose the best of the trials

| | binary | categorical | continuous | class |
|---|---|---|---|---|
| Tid | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
| 1 | Yes | Single | 125K | No |
| 2 | No | Married | 100K | No |
| 3 | No | Single | 70K | No |
| 4 | Yes | Married | 120K | No |
| 5 | No | Divorced | 95K | Yes |
| 6 | No | Married | 60K | No |
| 7 | Yes | Divorced | 220K | No |
| 8 | No | Single | 85K | Yes |
| 9 | No | Married | 75K | No |
| 10 | No | Single | 90K | Yes |

| Test Data / Prune Set/ Validation Set | | | |
|---|---|---|---|
| Owner | Marital | Income | Default |
| No | Single | 85k | No |
| Yes | Single | 100k | Yes |
| Yes | Divorced | 70k | No |
| No | Married | 90k | Yes |

# Practice Exercises (for Students)



- Compute the information gain for every possible split, for the given continuous valued attribute?
- Sort the values and find the the midpoint between each

  pair of adjacent values (split_point).

| Class | + | - | + | - | + | - | - |
|---|---|---|---|---|---|---|---|
| Sorted A2 | 1 | 3 | 4 | 5 | 6 | 7 | 8 |
| Split_Point | | 2 | 3.5 | 4.5 | 5.5 | 6.5 | 7.5 |

- Calculate Info(D)
- Calculate the entropy for each split_point for "<=" and ">"
- Find Gain for each split_point

| A1 | A2 | Class |
|---|---|---|
| T | 1 | + |
| T | 6 | + |
| T | 5 | - |
| F | 4 | + |
| F | 7 | - |
| F | 3 | - |
| F | 8 | - |
| T | 7 | + |
| F | 5 | - |

# Additional References

Decision Tree

- https://www.youtube.com/watch?v=eKD5gxPPeY0&list=PLBv09BD7ez_4temBw7vLA19p3tdQH6FYO&index=1

Overfitting

- https://www.youtube.com/watch?time_continue=1&v=t56Nid85Thg
- https://www.youtube.com/watch?v=y6SpA2Wuyt8

Random Forest

- https://www.stat.berkeley.edu/~breiman/RandomForests/

# Thank you !

**Required Reading for completed session :**

T1 - Chapter # 6 (Tom M. Mitchell, Machine Learning)

R1 – Chapter # 3,#4 (Christopher M. Bhisop, Pattern Recognition & Machine Learning)

**Prerequisite for next module:**

Refresh on the distance measure (L1 norm, L2 norm etc., ) from the Math course