Decision Tree Model

Decision trees are especially attractive for a data mining environment for three reasons.

- Due to their intuitive representation, they are easy to assimilate by humans.
- They can be constructed relatively fast compared to other methods.
- The accuracy of decision tree classifiers is comparable or superior to other models.

Decision Tree Learning

Building Decision Trees

Two steps for the decision tree learning

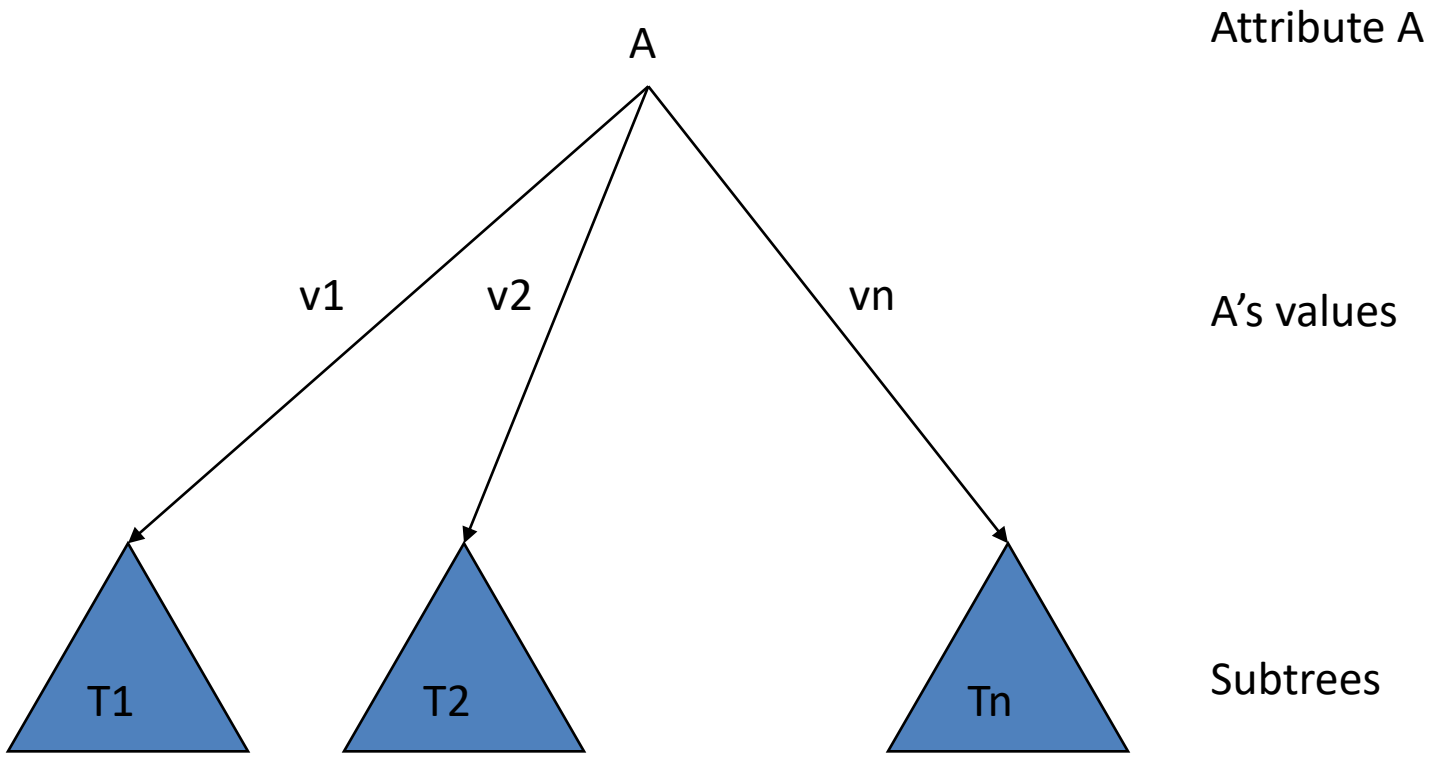
(1) Growing phase → maximizing the purity of the leaves

(2) (Post) pruning phase → minimizing the "true" error rate

* Pruning is an additional step to avoid the over-dependence to the growing sample(i.e.,overfitting)

Recursive partitioning

Resulting tree T is:



Issues in Decision Tree Learning

Choosing the splitting criterion

- Impurity based criteria
- Information gain
- Statistical measures of association…

Binary or multiway splits

- Multiway split: 1 value of the splitting attribute = 1 leaf
- Binary split: finding the best binary grouping
- Grouping only the leaves which are similar regarding the classes distribution

Finding the right sized tree

- Pre-pruning
- Post-pruning

Splitting Criteria

I. Splitting Criterion

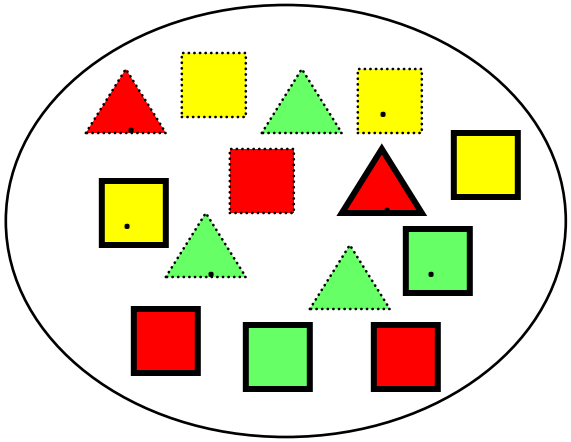
- Central Idea : Select attribute which partitions the learning set into subsets as “pure” as possible

- A partition is PURE if all of the observations in it belong to the same class.

Example: Triangles and Squares

#	Attribute			Shape
	Color	Outline	Dot	
1	green	dashed	no	triange
2	green	dashed	yes	triange
3	yellow	dashed	no	square
4	red	dashed	no	square
5	red	solid	no	square
6	red	solid	yes	triange
7	green	solid	no	square
8	green	dashed	no	triange
9	yellow	solid	yes	square
10	red	solid	no	square
11	green	solid	yes	square
12	yellow	dashed	yes	square
13	yellow	solid	no	square
14	red	dashed	yes	triange

Data Set:
A set of classified objects



I. Entropy & Information Gain – C4.5

Shannon entropy

Measure of uncertainty

$$E(Y) = -\sum_{k=1}^{K} \frac{n_{k.}}{n} \times \log_2\left(\frac{n_{k.}}{n}\right)$$

Condition entropy

Expected entropy of Y knowing the values of X

$$E(Y/X) = -\sum_{l=1}^{L} \frac{n_{.l}}{n} \sum_{k=1}^{K} \frac{n_{kl}}{n_{.l}} \times \log_2\left(\frac{n_{kl}}{n_{.l}}\right)$$

Information gain

Reduction of uncertainty

$$G(Y/X) = E(Y) - E(Y/X)$$

(Information) Gain ratio

Favors the splits with low number of leaves

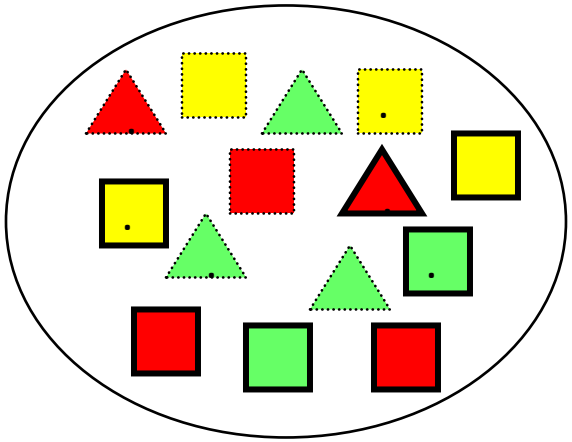
$$GR(Y/X) = \frac{E(Y) - E(Y/X)}{E(X)}$$

Example: Entropy of given dataset

- 5 triangles
- 9 squares
- class probabilities

$$p(\square) = \frac{9}{14}$$

$$p(\triangle) = \frac{5}{14}$$

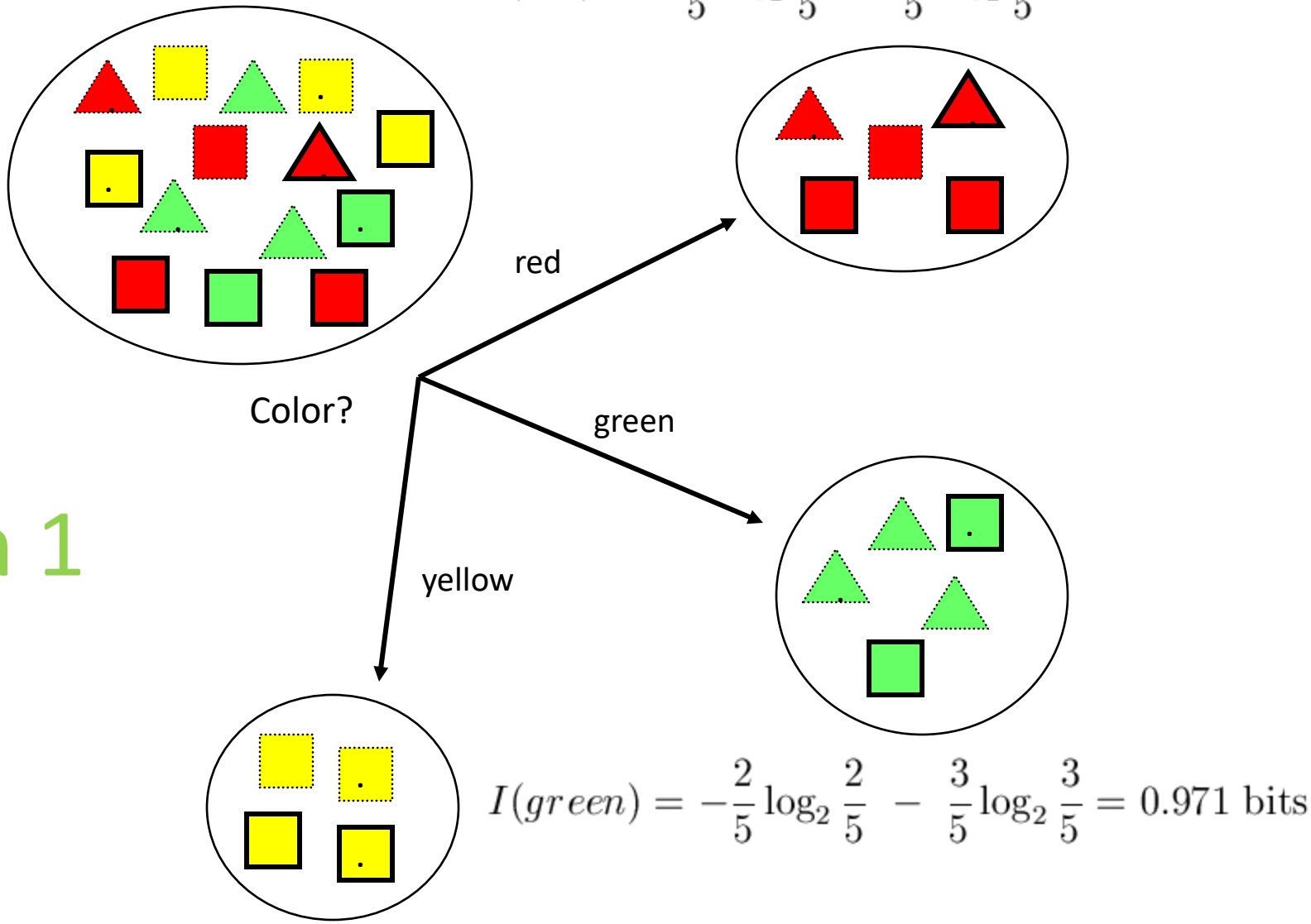


- entropy

$$I = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940 \text{ bits}$$

Depth 1

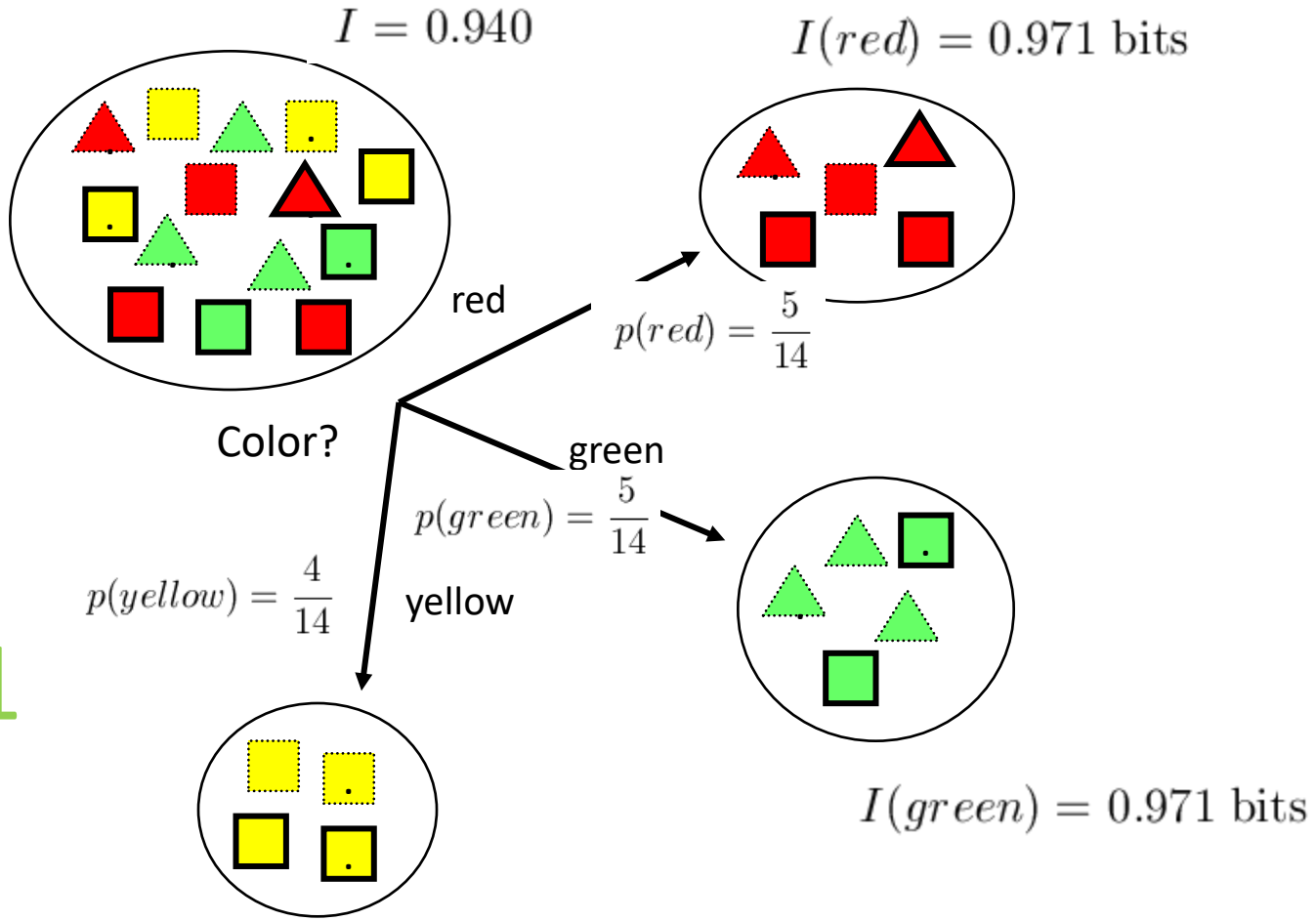
$$I(red) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.971 \text{ bits}$$



$$I(green) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971 \text{ bits}$$

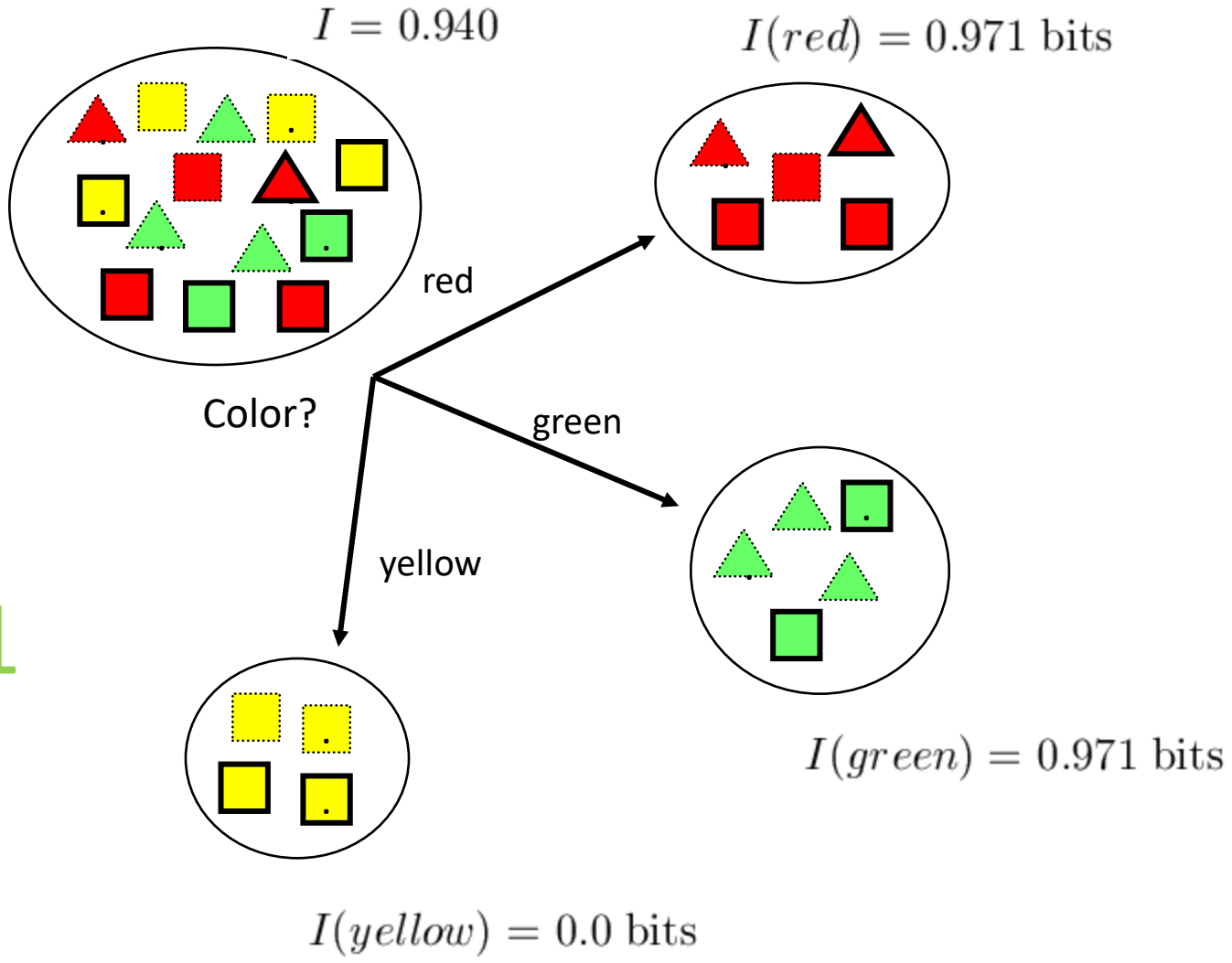
$$I(yellow) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0.0 \text{ bits}$$

Depth 1



$$I_{res}(\text{Color}) = \sum p(v)I(v) = \frac{5}{14}0.971 + \frac{5}{14}0.971 + \frac{4}{14}0.0 = 0.694 \; bits$$

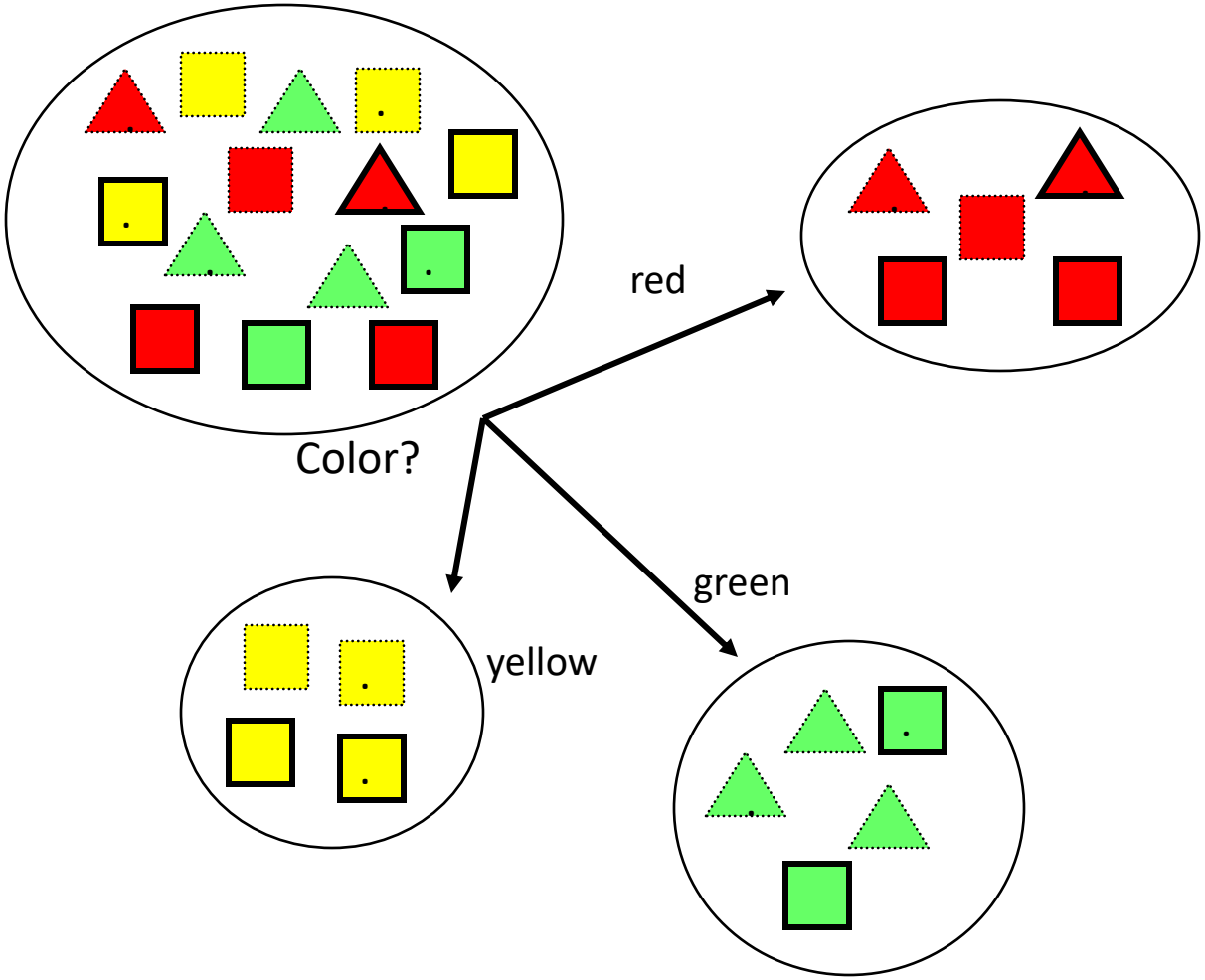
Depth 1



$$Gain(\text{Color}) = I - I_{res}(\text{Color}) = 0.940 - 0.694 = 0.246\ bits$$

Depth1 :Information Gains

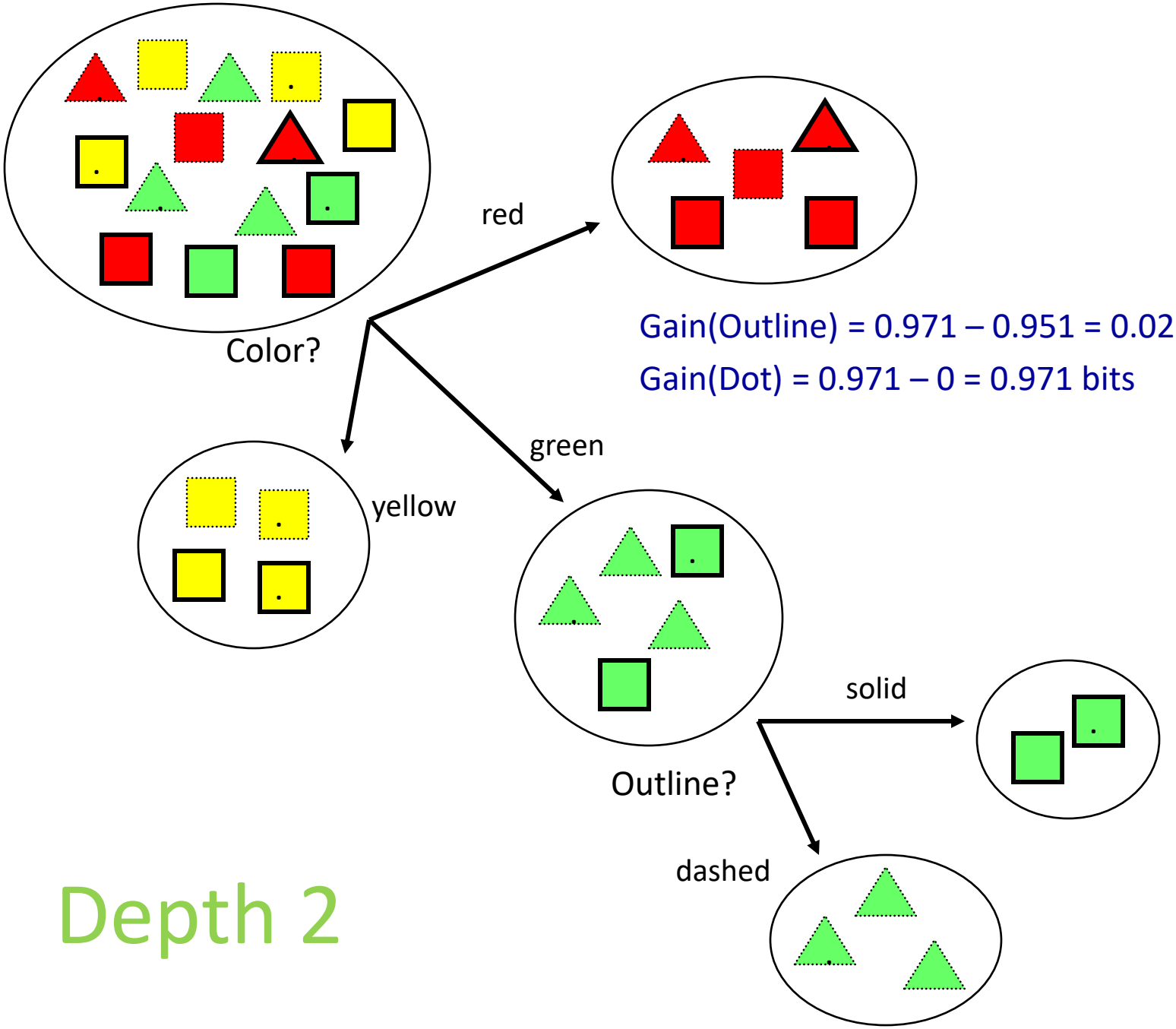
- Attributes
  - Gain(Color) = 0.246
  - Gain(Outline) = 0.151
  - Gain(Dot) = 0.048
- The attribute with the highest gain is chosen
- This heuristics is local (local minimization of impurity)



Gain(Outline) = 0.971 – 0 = 0.971 bits

Gain(Dot) = 0.971 – 0.951 = 0.020 bits

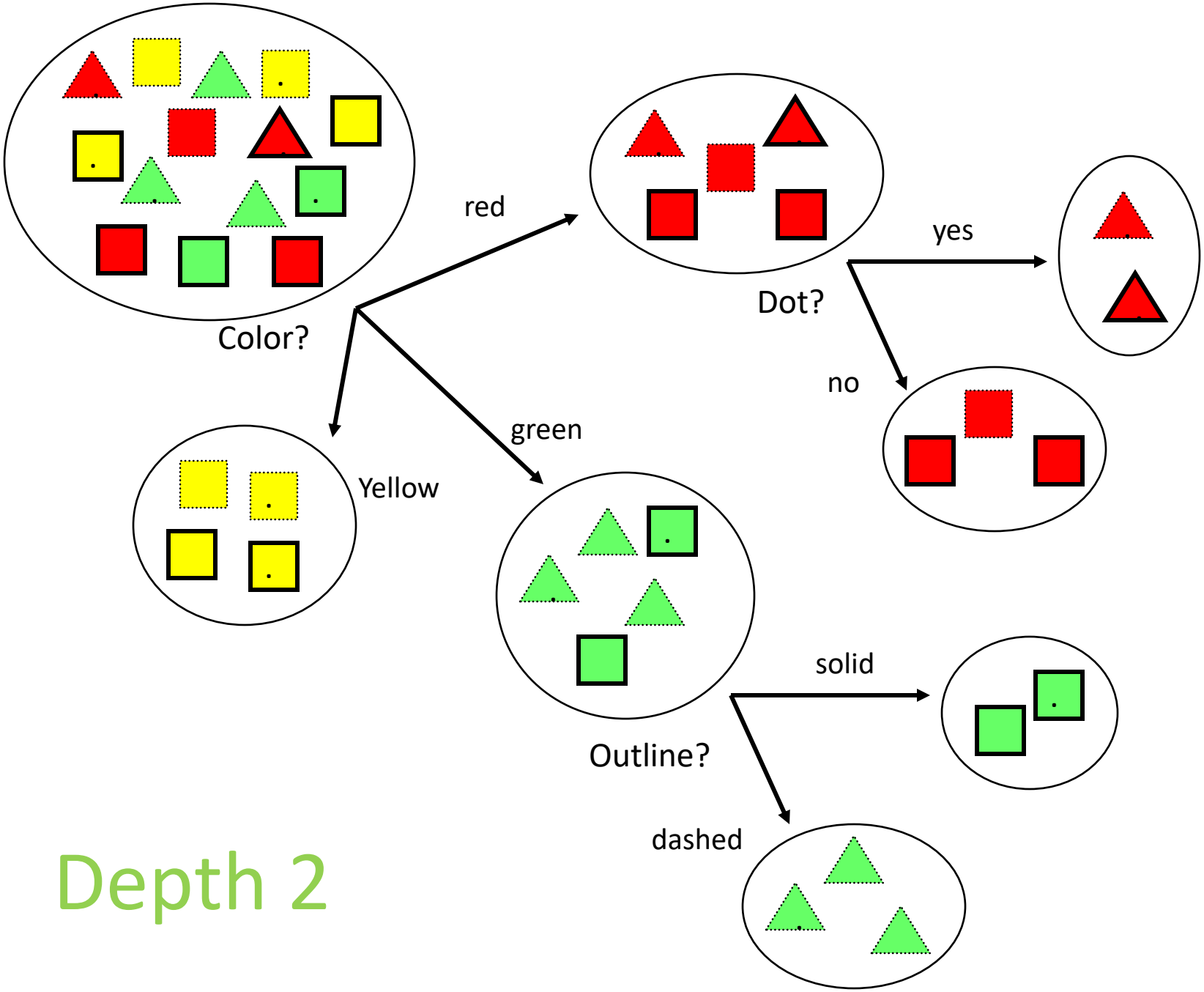
Depth 2



Gain(Outline) = 0.971 – 0.951 = 0.020 bits

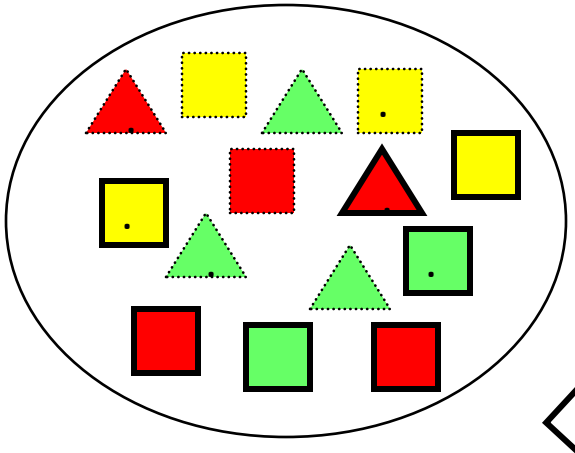
Gain(Dot) = 0.971 – 0 = 0.971 bits

Depth 2

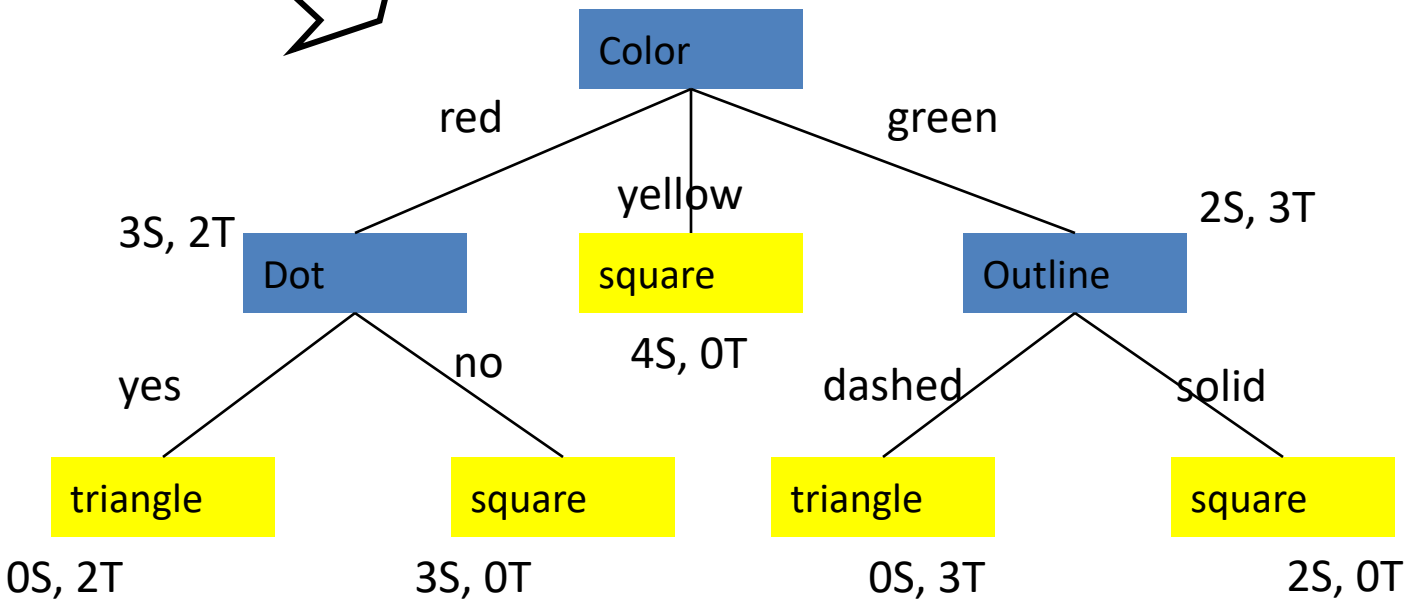


Depth 2

Final Decision Tree



9S, 5T



I. Gini Gain – CART

Gini index

Measure of impurity

$$I(Y) = -\sum_{k=1}^{K} \frac{n_{k.}}{n} \times \left(1 - \frac{n_{k.}}{n}\right)$$

Conditional impurity

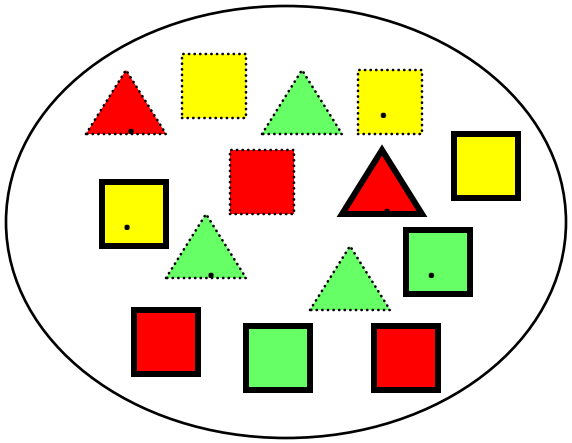
Average impurity of Y conditionally to X

$$I(Y/X) = -\sum_{l=1}^{L} \frac{n_{.l}}{n} \sum_{k=1}^{K} \frac{n_{kl}}{n_{.l}} \times \left(1 - \frac{n_{kl}}{n_{.l}}\right)$$

Gain

$$D(Y/X) = I(Y) - I(Y/X)$$

Gini Index



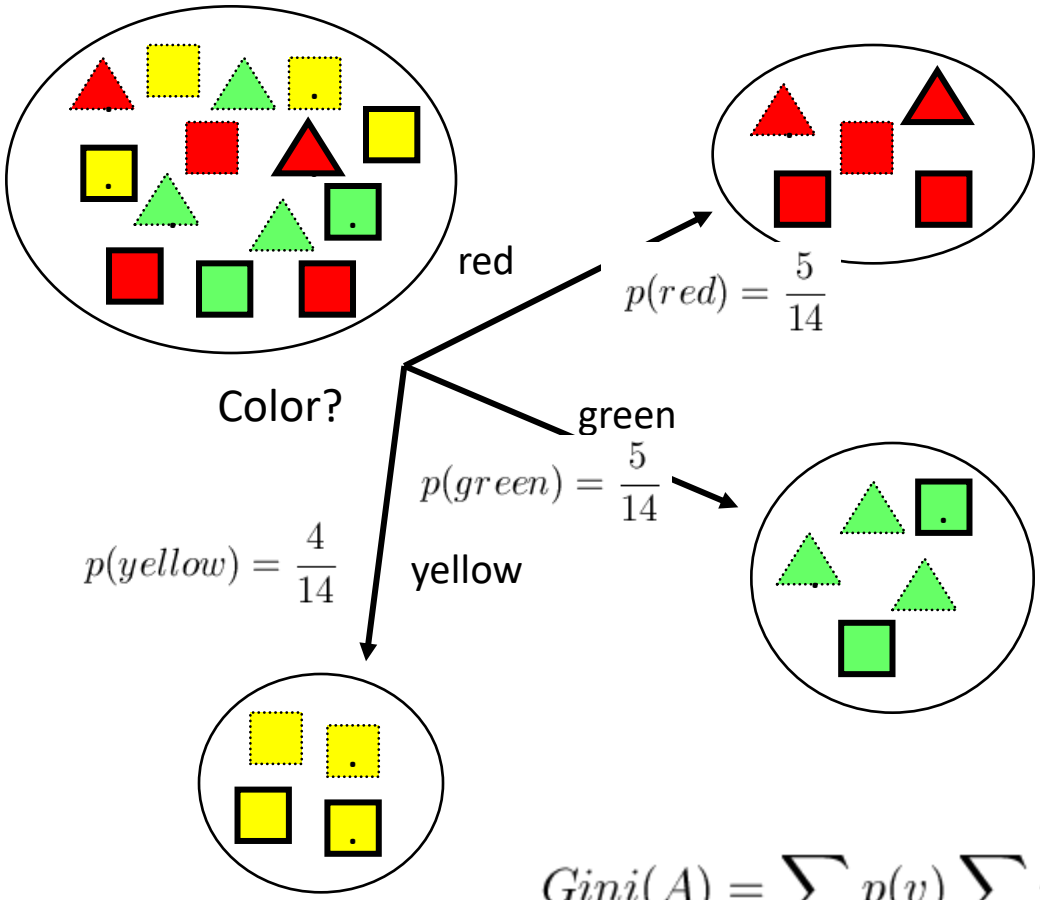
$$p(\square) = \frac{9}{14}$$

$$p(\triangle) = \frac{5}{14}$$

$$Gini = \sum_{i \neq j} p(i)p(j)$$

$$Gini = \frac{9}{14} \times \frac{5}{14} = 0.230$$

Gini Index for Color



$$Gini(A) = \sum_{v} p(v) \sum_{i \neq j} p(i|v)p(j|v)$$

$$Gini(\text{Color}) = \frac{5}{14} \times (\frac{3}{5} \times \frac{2}{5}) + \frac{5}{14} \times (\frac{2}{5} \times \frac{3}{5}) + \frac{4}{14} \times (\frac{4}{4} \times \frac{0}{4}) = 0.171$$

Gain of Gini Index

$$Gini = \frac{9}{14} \times \frac{5}{14} = 0.230$$

$$Gini(\text{Color}) = \frac{5}{14} \times (\frac{3}{5} \times \frac{2}{5}) + \frac{5}{14} \times (\frac{2}{5} \times \frac{3}{5}) + \frac{4}{14} \times (\frac{4}{4} \times \frac{0}{4}) = 0.171$$

$$\boxed{GiniGain(\text{Color}) = 0.230 - 0.171 = 0.058}$$

I. Un-biased measures

- Allows to alleviate the data fragmentation problem

- Gain Ratio corrects the bias of the information gain
- The Gini reduction in impurity is biased in favor of variables with more levels

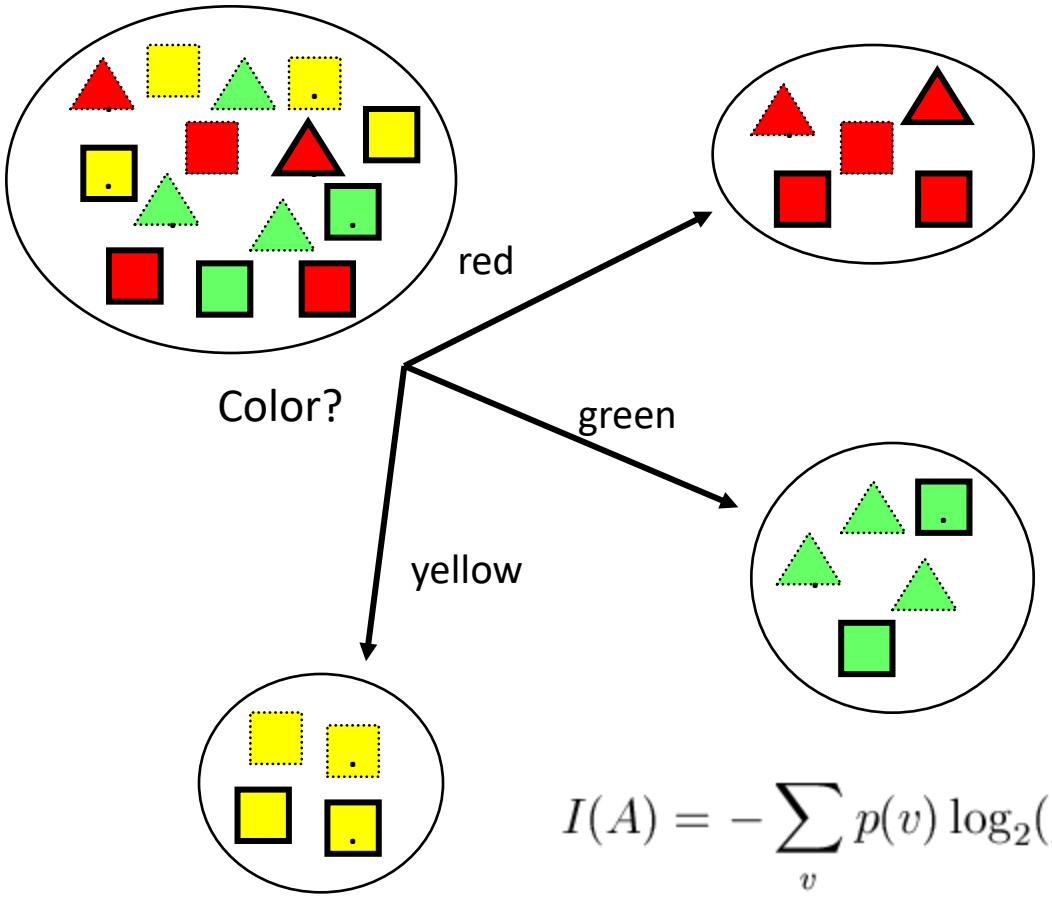
(but the CART algorithm constructs necessarily a binary decision tree)

Problems with Information Gain

Attributes which have a large number of possible values -> leads to many child nodes.

- Information gain is biased towards choosing attributes with a large number of values
- This may result in *overfitting* (selection of an attribute that is non-optimal for prediction)

Information Gain Ratio



$$I(A) = -\sum_{v} p(v)\log_2(p(v))$$

$$I(\text{Color}) = -\frac{5}{14}\log_2\frac{5}{14} - \frac{5}{14}\log_2\frac{5}{14} - \frac{4}{14}\log_2\frac{4}{14} = 1.58 \text{ bits}$$

$$GainRatio(\text{Color}) = \frac{Gain(\text{Color})}{I(\text{Color})} = \frac{0.940 - 0.694}{1.58} = 0.156$$

Information Gain and Information Gain Ratio

A	|v(A)|	Gain(A)	GainRatio(A)
Color	3	0.247	0.156
Outline	2	0.152	0.152
Dot	2	0.048	0.049

Three Impurity Measures

A	Gain(A)	GainRatio(A)	GiniGain(A)
Color	0.247	0.156	0.058
Outline	0.152	0.152	0.046
Dot	0.048	0.049	0.015

Merging/Splitting Process

II. Multi-way splitting in C4.5

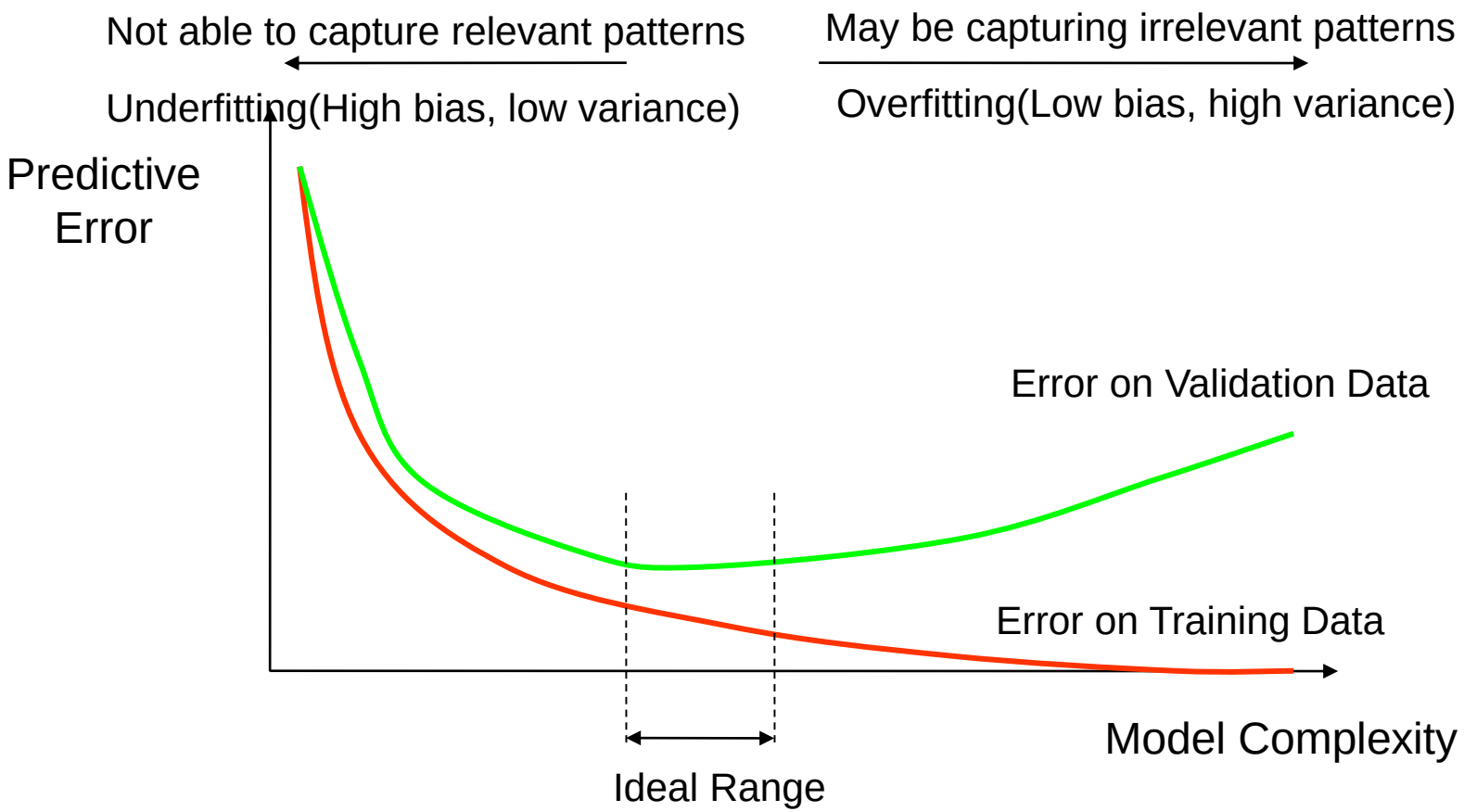
- The prediction rules are easy to read
- Data fragmentation problem, especially for small dataset
- "Large" decision tree with a high number of leaves
- "Low depth" decision tree

II. Binary splitting in CART

- This grouping allows to overcome the bias of the splitting measure used
- The data fragmentation problem is alleviated
- "High depth" decision tree (CART uses a post pruning process for remedy to this)
- Merging into two groups is not always relevant !

Finding the Right sized tree

III. Finding the right sized tree



Bias ~ how powerful is the model

Variance ~ how sensitive the model is to the training set

III. Avoid overfitting

Pruning is an additional step in order to avoid the over-dependence to the growing sample (i.e.,overfitting)

III. Pre-pruning: Stopping the growing process

Confidence and support criteria

- Group purity: confidence threshold
- Support criterion: min. size node to split, min. instances in leaves

III. Postpruning in CART

- The error cost of any subtree is defined as follows:

$$\text{cost}(\text{subtree}) = \sum_{\text{leaves } j} \text{misclassification error at node j} * \text{proportion of data instances reached to node j}$$

- The above measure alone is not sufficient for determining which subtree is better, because larger trees typically have smaller values of misclassification cost and a large number of leaves.

III. Postpruning in CART

- We would like to find another tree with a smaller number of leaves at the expense of allowing cost to rise somewhat. That is, we seek a balance between the number of leaves and the misclassification cost.

- The complexity of a tree is the number of its leaves and the cost-complexity measure is:

$$\text{cost}(\text{subtree}) = \sum_{\text{leaves } j} \underbrace{\text{misclassification error at node j}}_{} * \underbrace{\text{proportion of data instances reached to node j}}_{} + \alpha\,[\#\text{leaves in subtree}]$$

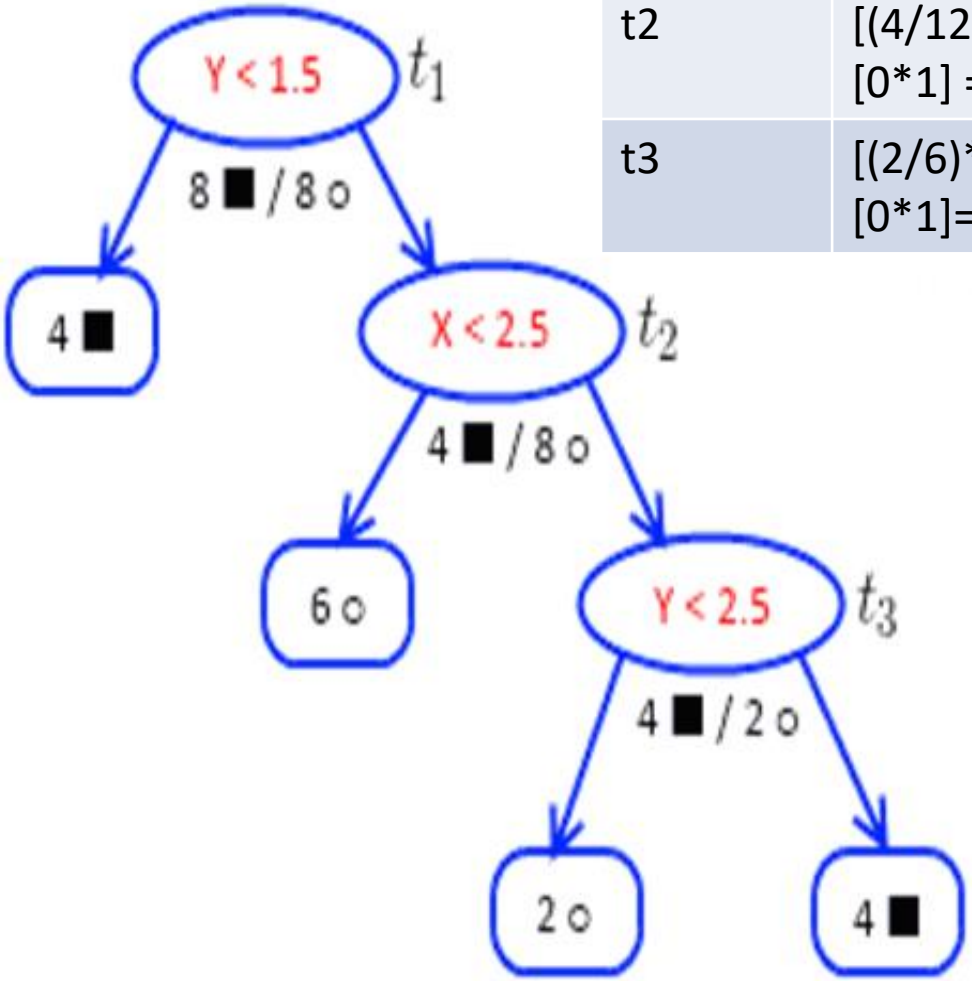
III. Postpruning in CART

- If the complexity parameter α is 0, then the initial tree is the best, i.e., having the smallest cost-complexity measure; if the complexity parameter goes to infinity, then the tree containing the root node only is the best.

- As the complexity parameter increases from 0, there will be some internal node that becomes ineffective and can be pruned.

Example

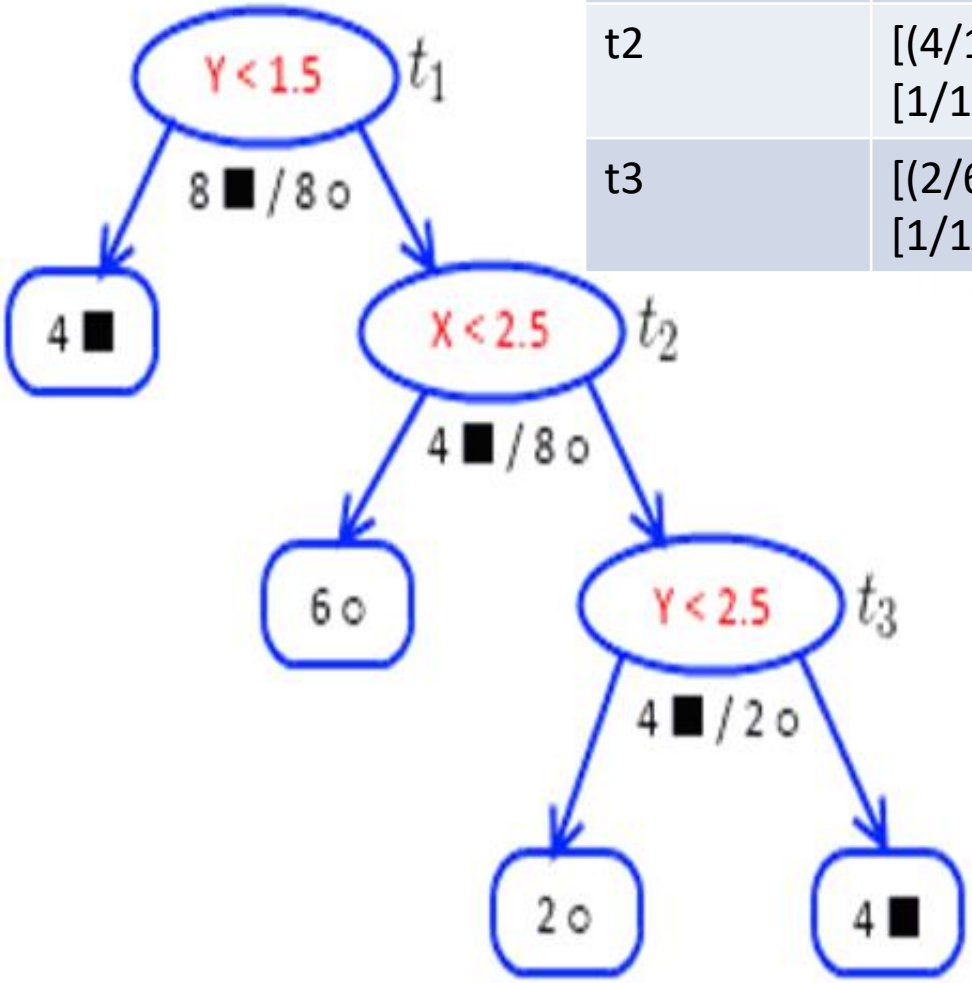
alpha=0	Prune error	Unpruned error
t1	[(8/16) * (16/16)] + [0*1] = 0.5	[(0 * 4/16) + (0* 6/16) + (0* 2/16)+ (0* 4/16)] +[0*4] = 0
t2	[(4/12)*(12/16)]+ [0*1] = 0.25	[0] + [0*3] = 0
t3	[(2/6)*(6/16)] + [0*1]=0.125	[0]+[0*2] = 0



** The unpruned tree cc error is less compared to pruned tree cc error at each node. So, no pruning required for alpha=0.*

Example

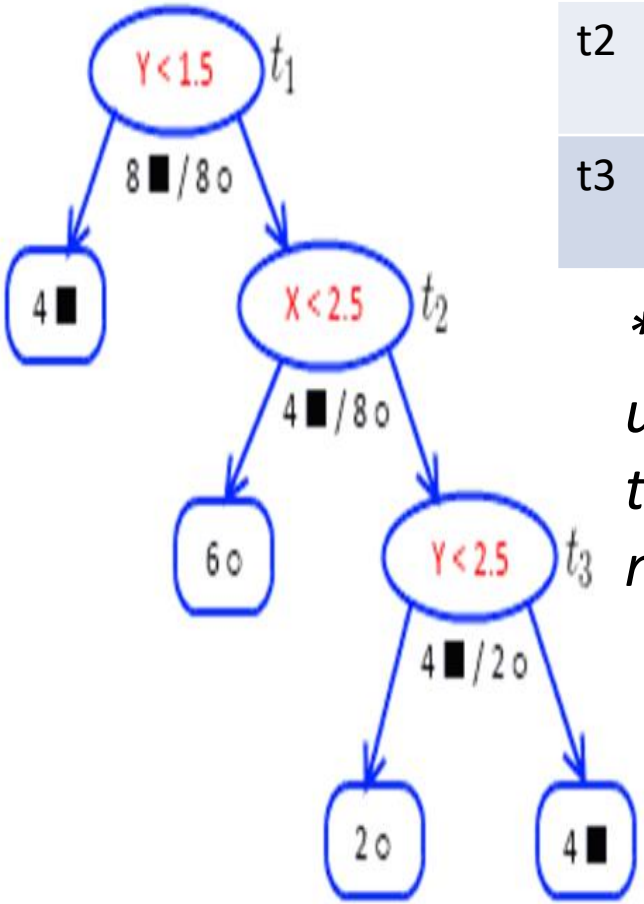
alpha=1/10	Prune error	Unpruned error
t1	[(8/16) * (16/16)]+ [1/10 *1] = 0.6	[(0 * 4/16) + (0* 6/16) + (0* 2/16)+ (0* 4/16)] +[1/10 *4] = 0.4
t2	[(4/12)*(12/16)]+ [1/10 *1] = 0.35	[0] + [1/10 *3] = 0.3
t3	[(2/6)*(6/16)] + [1/10 *1]=0.225	[0]+[1/10 *2] = 0.2



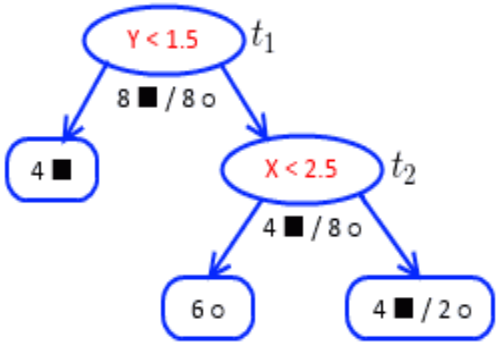
** The unpruned tree cc error is less compared to pruned tree cc error at each node. So, no pruning required for alpha=1/10.*

Example

alpha=1/8	Prune error	Unpruned error
t1	[(8/16) * (16/16)]+ [1/8*1] = 0.625	[(0 * 4/16) + (0* 6/16) + (0* 2/16)+ (0* 4/16)] +[1/8 *4] = 0.5
t2	[(4/12)*(12/16)]+ [1/8*1] = 0.375	[0] + [1/8 *3] = 0.375
t3	[(2/6)*(6/16)] + [1/8*1]=0.25	[0]+[1/8 *2] = 0.25

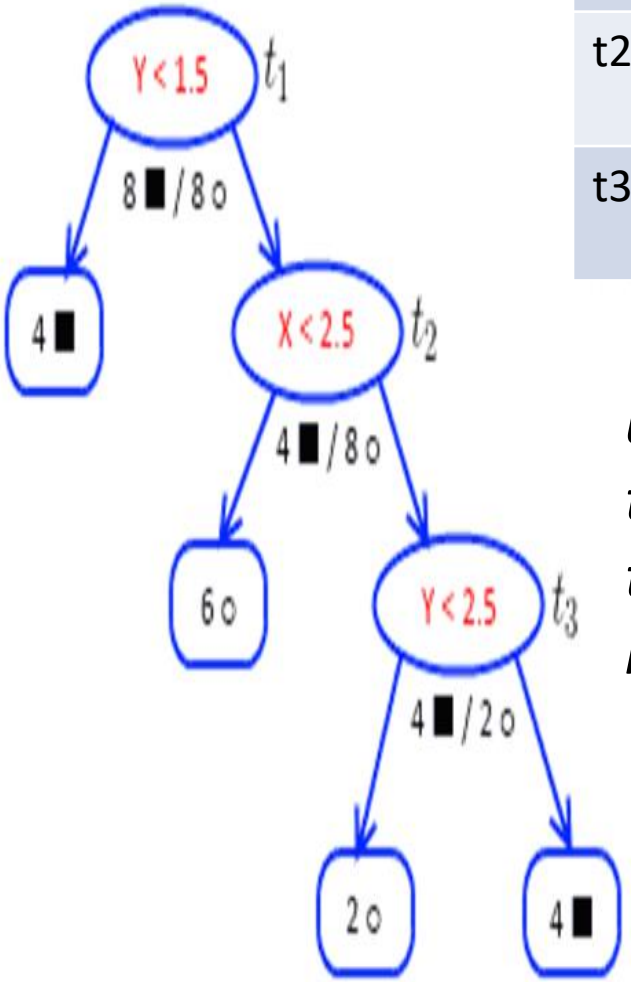


** The pruned tree cc error is same as that of unpruned tree cc error at t2 and t3. To resolve tie, we select the node that eliminates less number of leaf nodes(i.e.,t3).*

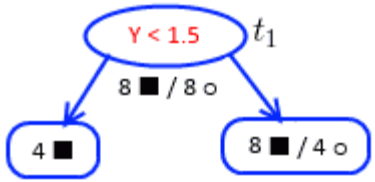


Example

alpha=1/4	Prune error	Unpruned error
t1	[(8/16) * (16/16)]+ [1/4 *1] = 0.75	[(0 *4/16) + (0*6/16) + (0*2/16)+ (0*4/16)] +[1/4 *4] = 1
t2	[(4/12)*(12/16)]+ [1/4 *1] = 0.5	[0] + [1/4 *3] = 0.75
t3	[(2/6)*(6/16)] + [1/4 *1]=0.375	[0]+[1/4 *2] = 0.5

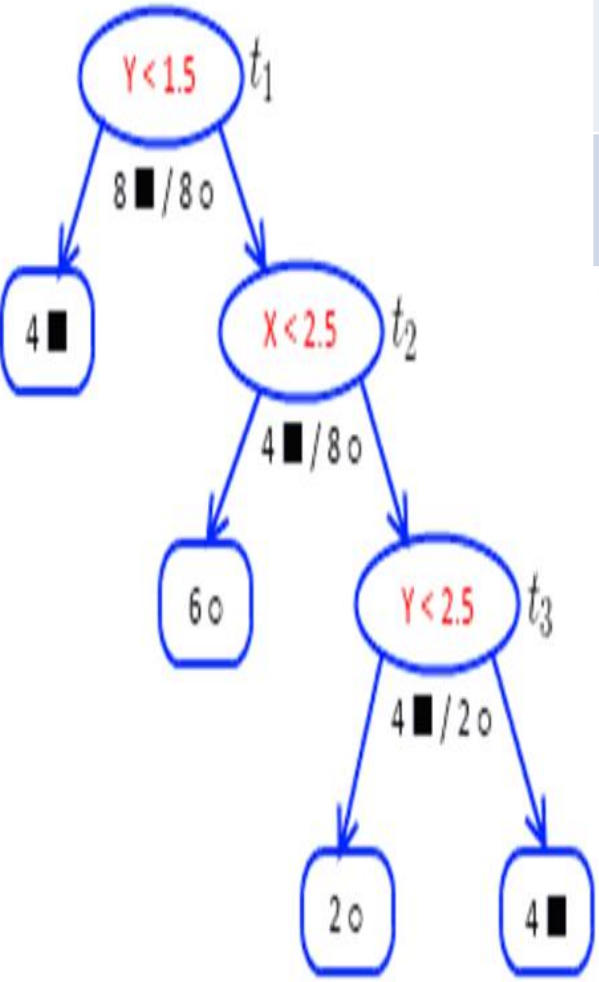


** The pruned tree cc error is less as that of unpruned tree cc error at t1, t2 and t3. t1 and t2 are reducing error more than t3. To resolve tie, we select the node that eliminates less number of leaf nodes(i.e.,t2).*

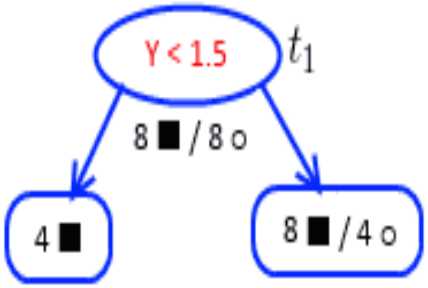


Example

alpha=1/2	Prune error	Unpruned error
t1	[(8/16) * (16/16)]+ [1/2 *1] = 1	[(0 *4/16) + (0*6/16) + (0*2/16)+ (0*4/16)] +[1/2 *4] = 0.125
t2	[(4/12)*(12/16)]+ [1/2 *1] = 0.75	[0] + [1/2 *3] = 1.5
t3	[(2/6)*(6/16)] + [1/2 *1]=0.625	[0]+[1/2 *2] = 1

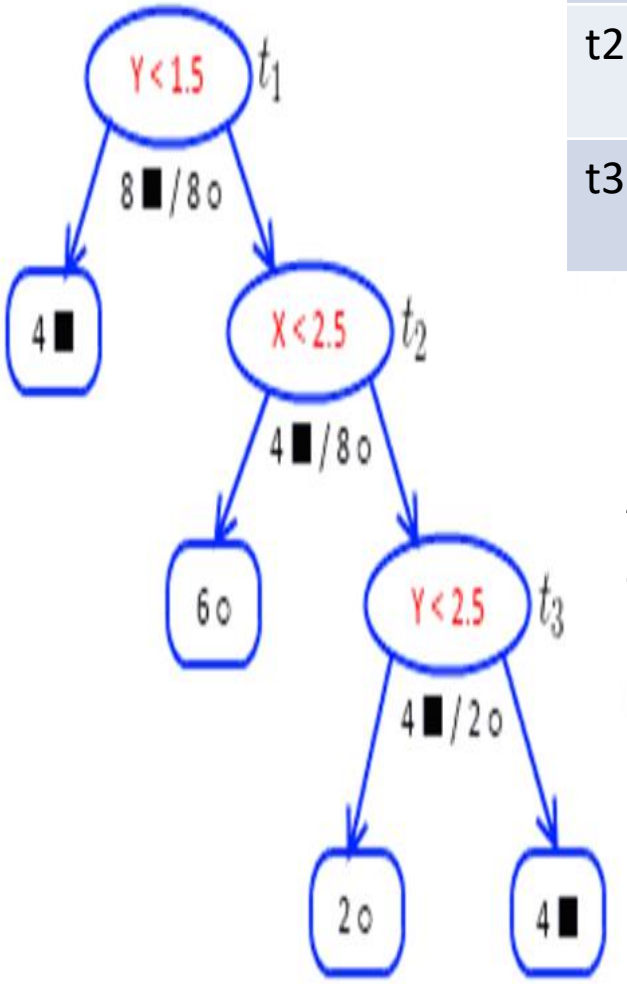


** The pruned tree cc error is less as that of unpruned tree cc error at t2 and t3. t2 is reducing error more than t3. So, we prune at t2.*

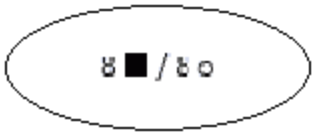


Example

alpha=1	Prune error	Unpruned error
t1	[(8/16) * (16/16)]+ [1 *1] = 0.5	[(0 *4/16) + (0*6/16) + (0*2/16)+ (0*4/16)] + [1 *4] = 4
t2	[(4/12)*(12/16)]+ [1 *1] = 0.25	[0] + [1 *3] = 3
t3	[(2/6)*(6/16)] + [1 *1]=0.125	[0]+ [1 *2] = 2



** The pruned tree cc error is less as that of unpruned tree cc error at t1, t2 and t3. Since t1 is reducing error more than t2 and t3, we prune at t1.*



III.Postpruning in C4.5

- No Validation dataset for pruning
- Predicted (pessimistic) error rate = upper bound of the confidence limit of the resubstitution error rate

III. Postpruning in C4.5

- Make estimates of the error based on the whole training data.
- Build full decision tree from training set
- For every non-leaf node N
  - The majority class is chosen to represent the node
  - Count the number of errors, #e/N = error rate
  - Establish a confidence interval and use the upper limit, pessimistic estimate of the error rate
  - Compare such estimate with the combined estimate of the error estimates for the leaves
  - If the first is smaller replace the node by a leaf

CART vs C4.5

Method	CART	C4.5
Splitting criterion	Gini index	Information Gain (Gain Ratio)
Merging process	Binary grouping	No merging 1 value = 1 leaf
Determining the right sized tree (overall)	Min. size to node split Min. instances in leaves Confidence threshold Tree depth	
Determining the right sized tree (specific)	Post-pruning Cost complexity pruning	Post-pruning Error based pruning
Recommended when...	Classification performance - Reliability No complicated settings	Small dataset Not sensitive to the settings

Pros & Cons

Advantages over other approaches

- No formal distributional assumptions
- Can automatically fit highly non-linear interactions
- Automatic variable selection
- Handle missing values
- Very easy to interpret if the tree is small
- The terminal nodes suggest a natural clustering
- Same tool for regression and classification
- Handle categorical predictors naturally
- Quick to fit, even for large problems

Disadvantages

- *Accuracy* – newer methods can have lower error rates than CART & C4.5

- *Instability* – if we change the data a little, the tree picture can change a lot