Decision Trees

Dr. G. Bharadwaja Kumar
VIT Chennai

Decision Tree

- Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression.

- The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

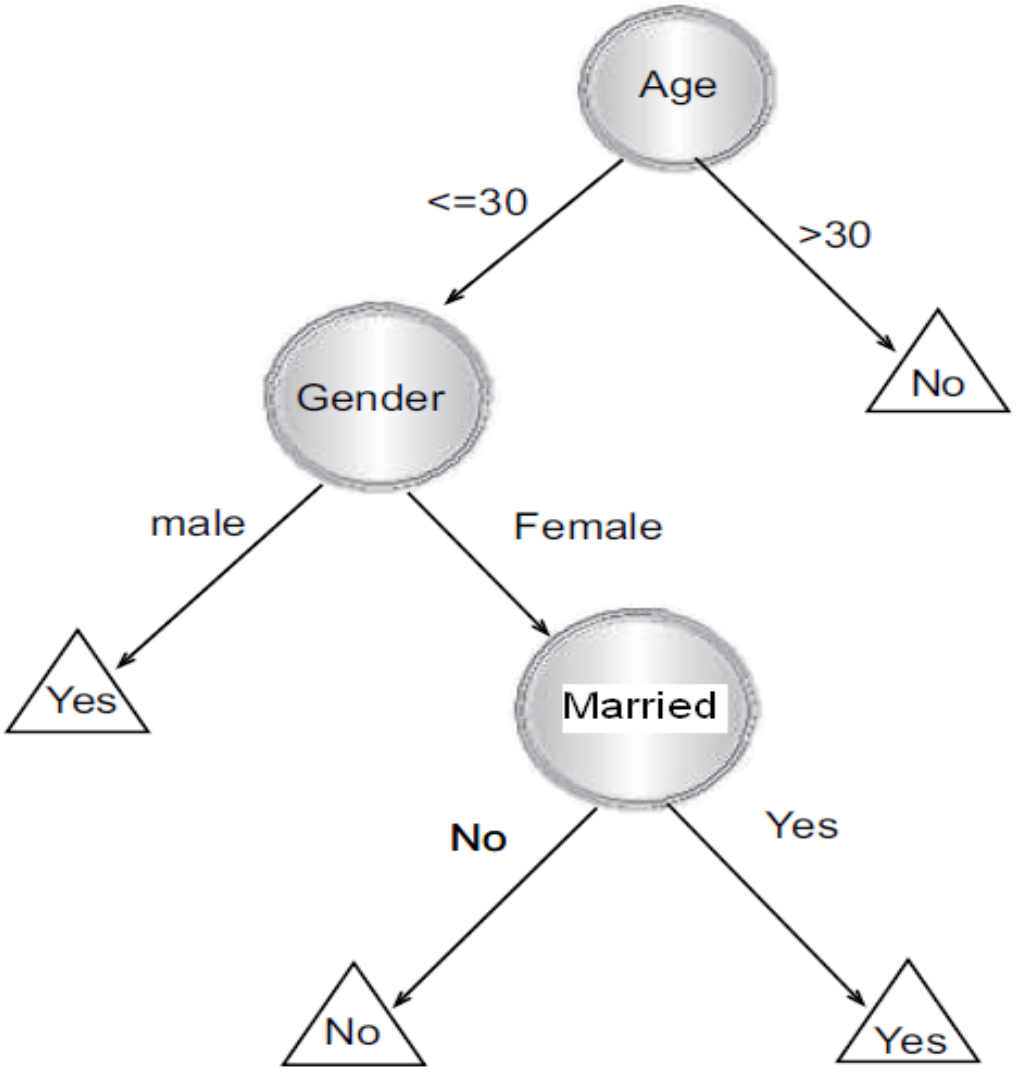
Decision Tree

- A decision tree is a classifier expressed as a recursive partition of the instance space.
- The decision tree consists of nodes that form a *rooted tree*, meaning it is a *directed tree* with a node called “root” that has no incoming edges.
- All other nodes have exactly one incoming edge.
- A node with outgoing edges is called an *internal* or test node.
- All other nodes are called leaves (also known as terminal or decision nodes).

Decision Tree Representation

- Each internal node tests an attribute
- Each branch corresponds to attribute value
- Each leaf node assigns a classification label
- Path: a disjunction of test to make the final decision

Decision Tree Example

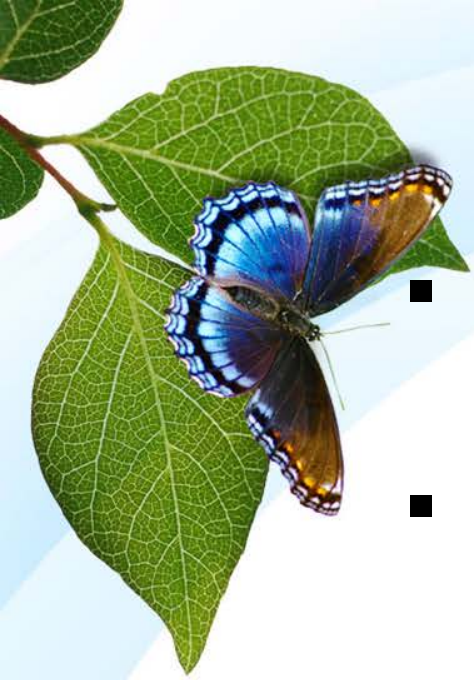


When to consider Decision Trees

- Instances describable by attribute-value pairs
- Attributes with both numerical & categorical values
- Target function is discrete valued
- When disjunctive hypothesis are required
- Possibly noisy training data

Noise in the data

- There are many kinds of "noise" that could occur in the examples:
  - Two examples have the same attribute, value pairs, but different classifications
  - Some values of attributes are incorrect because of errors in the data acquisition process or the preprocessing phase
  - The classification is wrong because of some error
  - Some attributes are irrelevant to the decision-making process.



Advantages

- Are simple to understand and interpret.
- Decision tree displays all the alternatives and their outcomes.
- Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by boolean logic.
- The systematic display of the decision process guides to approach decision making in orderly manner.

Advantages

- Able to handle both numerical and categorical data.
- Decision tree device is especially useful in cases where the decisions in the initial stages affect the subsequent decision process.
- Decision trees are capable of handling datasets that may have errors.
- Decision trees are capable of handling datasets that may have missing values.

Disadvantages

- Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

- Decision-tree learners can create over-complex trees that do not generalise the data well. This is called overfitting.

- Calculations can get very complex particularly if many values are uncertain and/or if many outcomes are linked.

Disadvantages

- There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.

Ways to Overcome some of the Disadvantages

- Mechanisms such as pruning (not currently supported), setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid overfitting problem.

- Instability problem is mitigated by using decision trees within an ensemble.

Decision Tree Algorithms

- The basic idea behind any decision tree algorithm is as follows:
  - Choose the *best* attribute(s) to split the remaining instances and make that attribute as root node
  - Repeat this process for recursively for each child

Decision Tree Algorithms

- Stopping Criteria:
  - All the instances have the same target attribute value
  - There are no more attributes
  - There are no more instances
  - The best splitting criteria is not greater than a certain threshold

Principle Criterion

- Selection of an attribute to test at each node - choosing the most useful attribute for classifying examples.

- In most of the cases, the discrete splitting functions are univariate i.e. an internal node is split according to the value of a single attribute.

Principle Criterion

- Consequently, the inducer searches for the best attribute upon which to split. There are various univariate criteria. These criteria can be characterized in different ways, such as:
- According to the origin of the measure: information theory, dependence, and distance.
- According to the measure structure: impurity based criteria, normalized impurity based criteria and Binary criteria.

Principle Criterion

- Purity (Diversity) Measures:
  - Gini (population diversity)
  - Entropy (information gain)
  - Information Gain Ratio
  - Chi-square Test

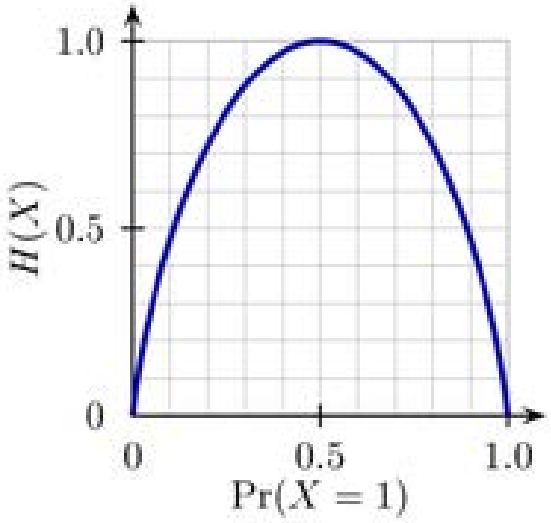
Entropy

- A measure of homogeneity of the set of examples.
- Given a set S of positive and negative examples of some target concept (a 2-class problem), the entropy of set S relative to this binary classification is

$$E(S) = - p(P)\log_2 p(P) - p(N)\log_2 p(N)$$

Entropy

- Entropy is minimized when all values of the target attribute are the same.
- Entropy is maximized when there is an equal chance of all values for the target attribute (i.e. the result is random)
- For a two-class problem:



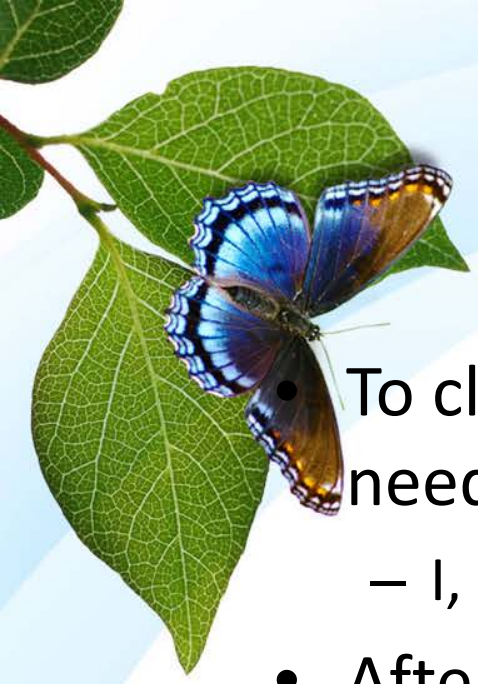
- “High Entropy”
  - X is from a uniform like distribution
  - Flat histogram
  - Values sampled from it are less predictable
- “Low Entropy”
  - X is from a varied (peaks and valleys) distribution
  - Histogram has many lows and highs
  - Values sampled from it are more predictable

Information Gain

- information gain
  - measures how well a given attribute separates the training examples according to their target classification
  - This measure is used to select among the candidate attributes at each step while growing the tree

Information Gain

- We want to determine which attribute in a given set of training feature vectors is most useful for discriminating between the classes to be learned.
- Information gain tells us how important a given attribute of the feature vectors is.
- We will use it to decide the ordering of attributes in the nodes of a decision tree.



Information-Theoretic Approach

- To classify an object, a certain information is needed
  - I, information
- After we have learned the value of attribute A, we only need some remaining amount of information to classify the object
  - Ires, residual information
- Gain
  - Gain(A) = I – Ires(A)
- The most ‘informative’ attribute is the one that minimizes Ires, *i.e.*, maximizes Gain

Entropy

- The average amount of information *I* needed to classify an object is given by the entropy measure

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

Residual Information

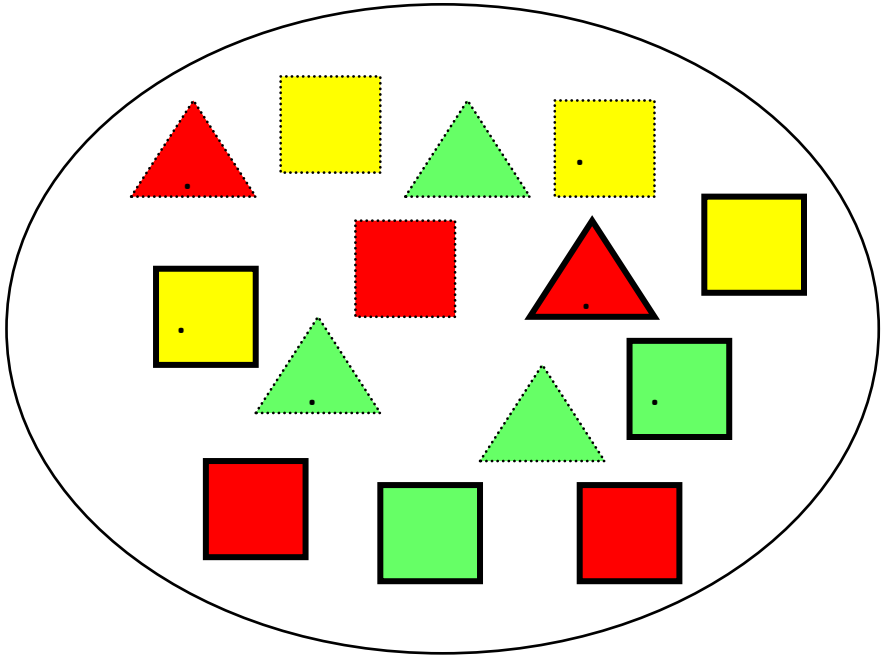
- After applying attribute *A*, *S* is partitioned into subsets according to values *v* of *A*
- *Ires* is equal to weighted sum of the amounts of information for the subsets

$$I_{res} = -\sum_{v} p(v) \sum_{c} p(c|v) \log_2 p(c|v)$$

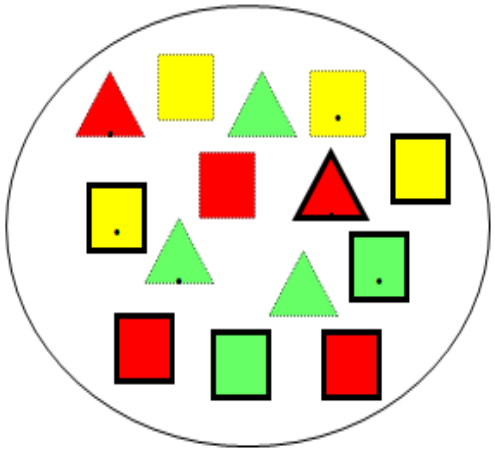
Example Data set

#	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



Entropy



- 5 triangles
- 9 squares
- class probabilities

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

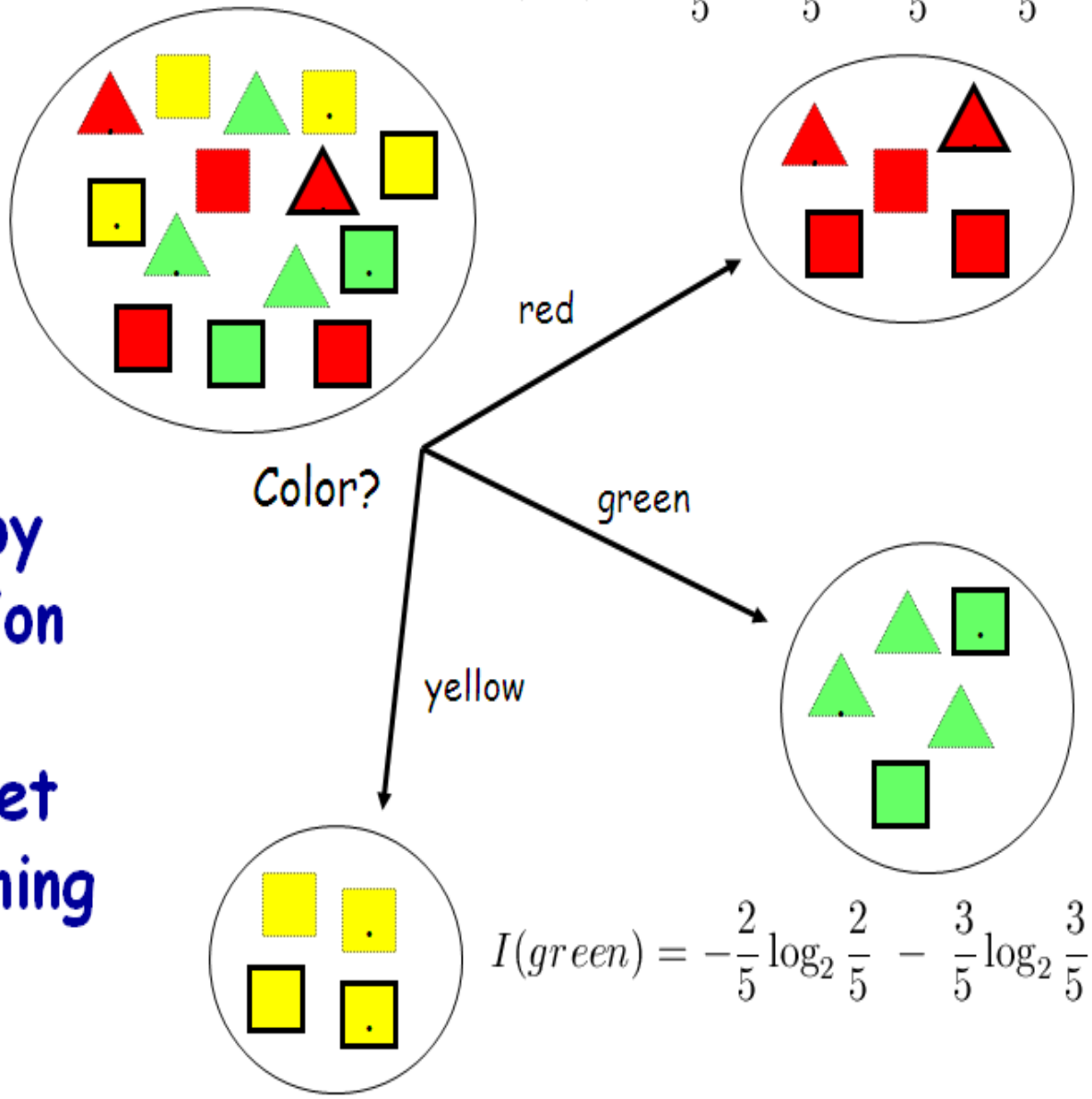
$$p(\triangle) = \frac{5}{14}$$

- entropy

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

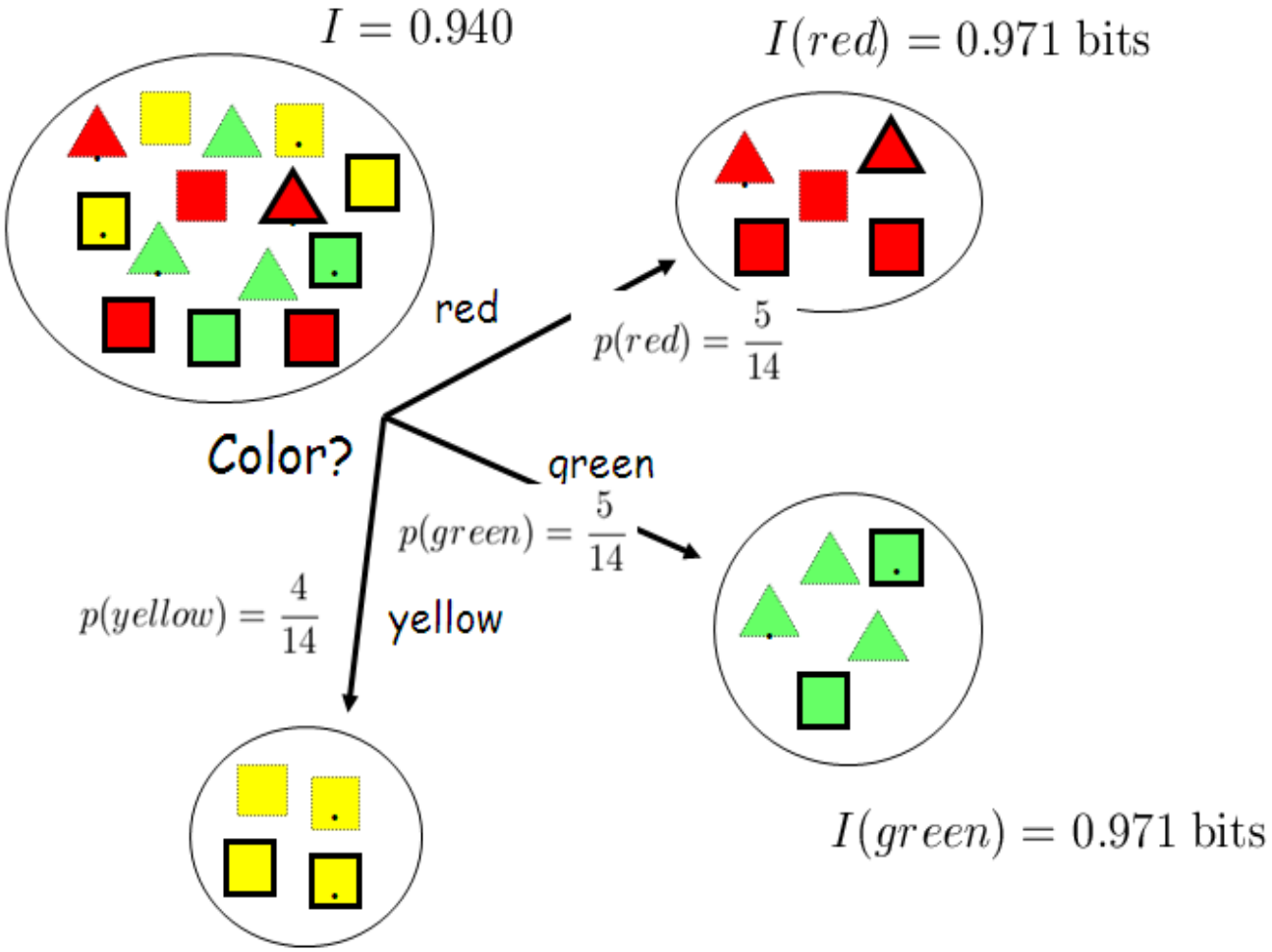
Entropy reduction by data set partitioning

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



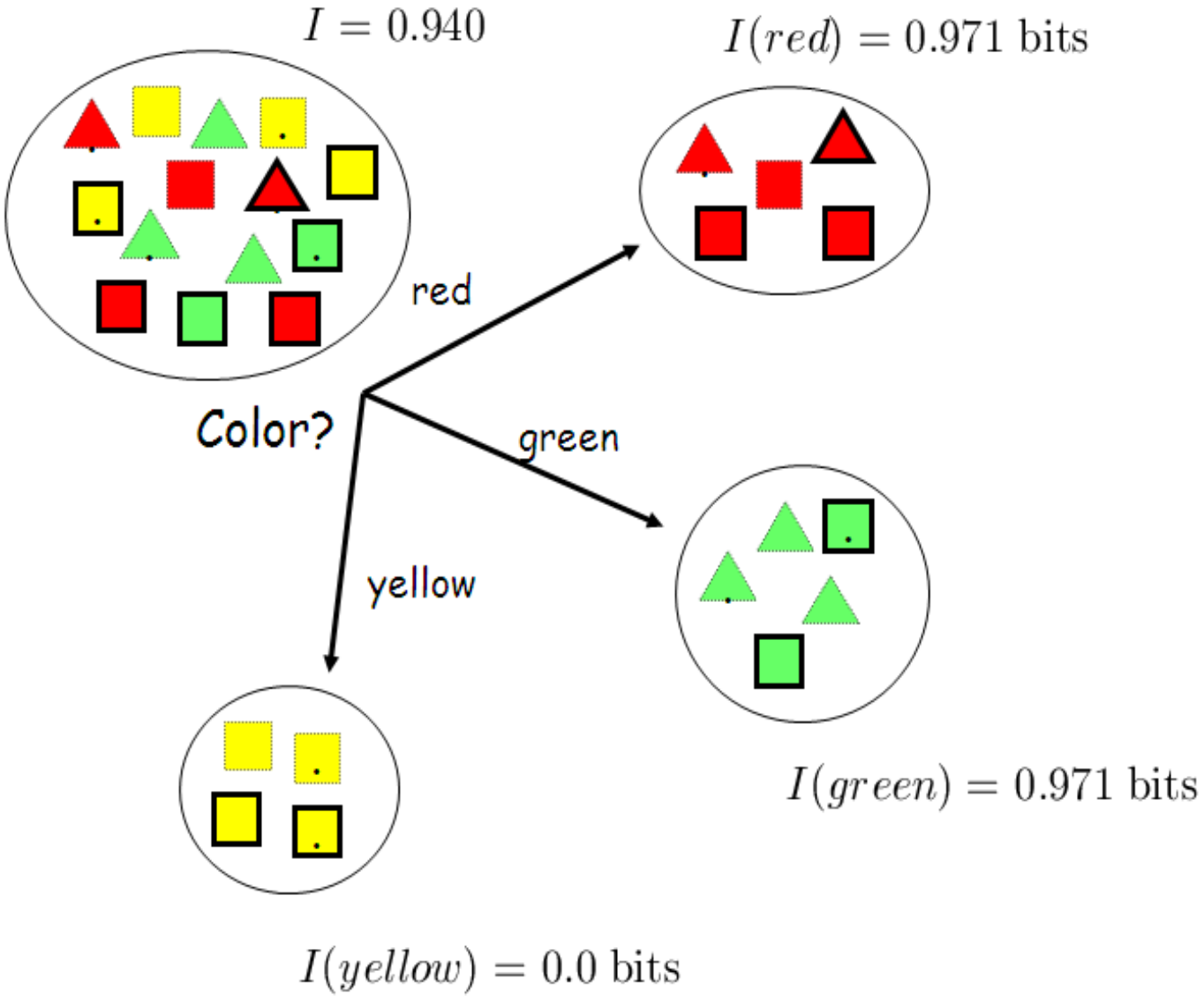
$$I(green) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971 \text{ bits}$$

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



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

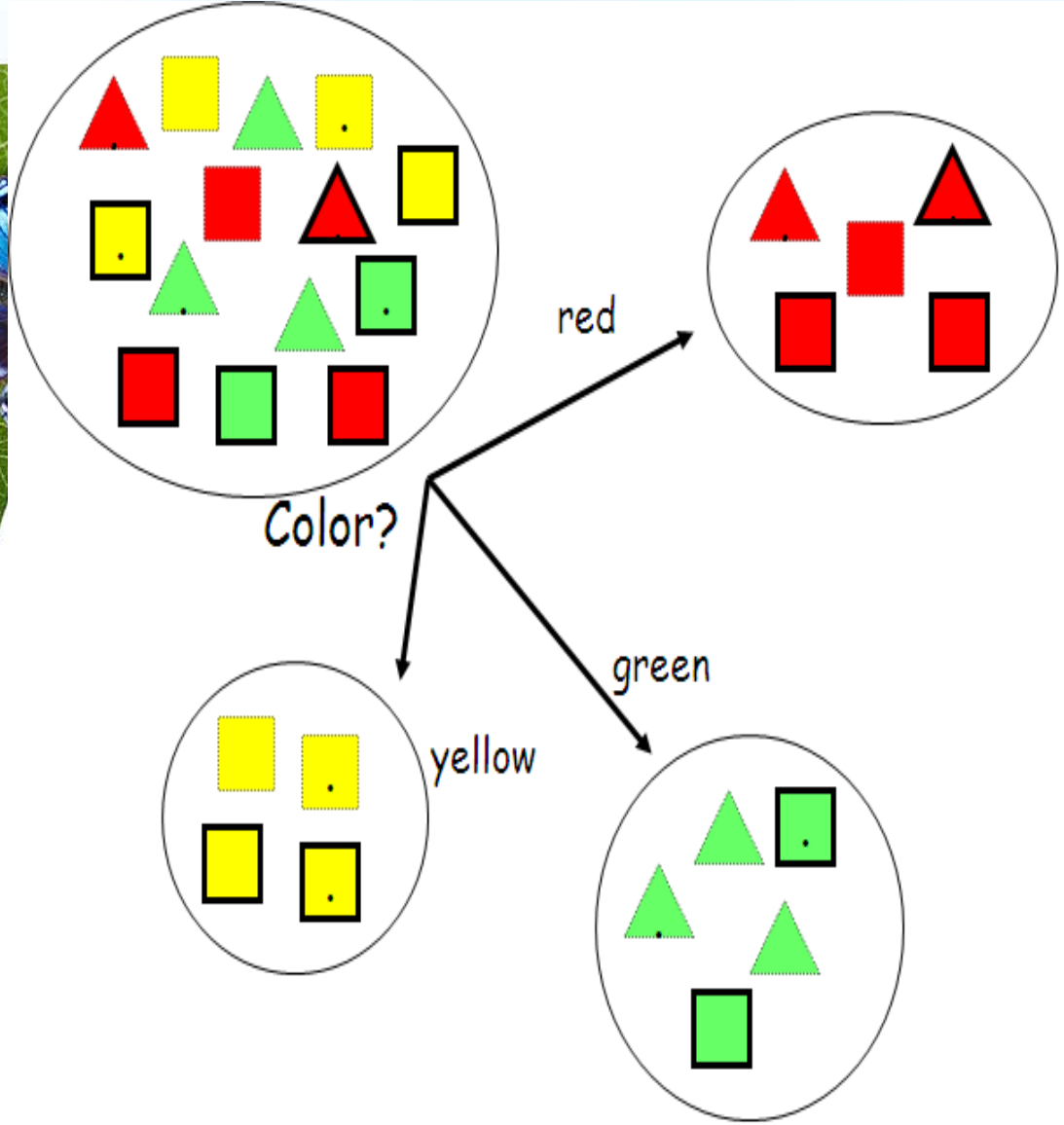
Information Gain



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

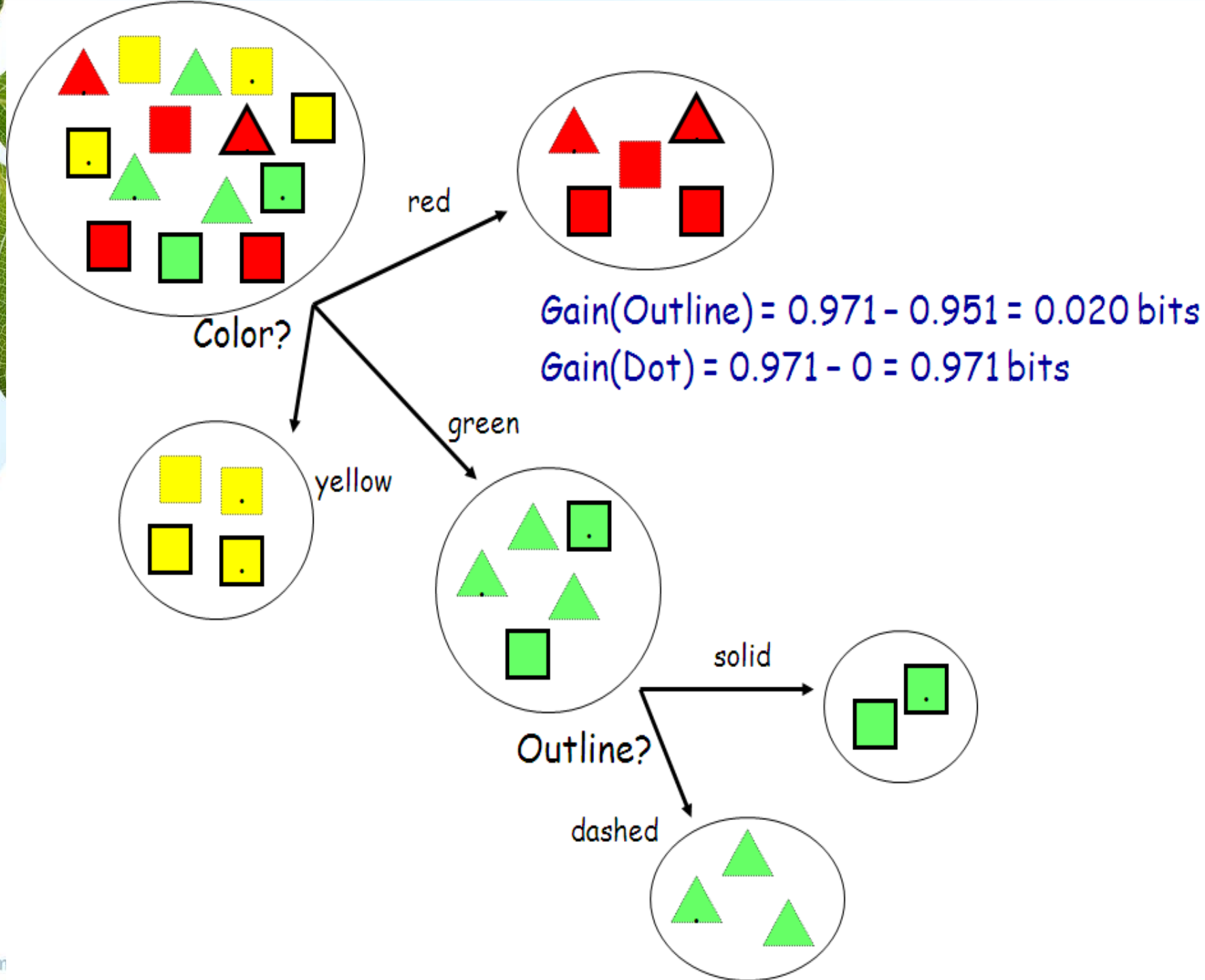
Information Gain of The Attribute

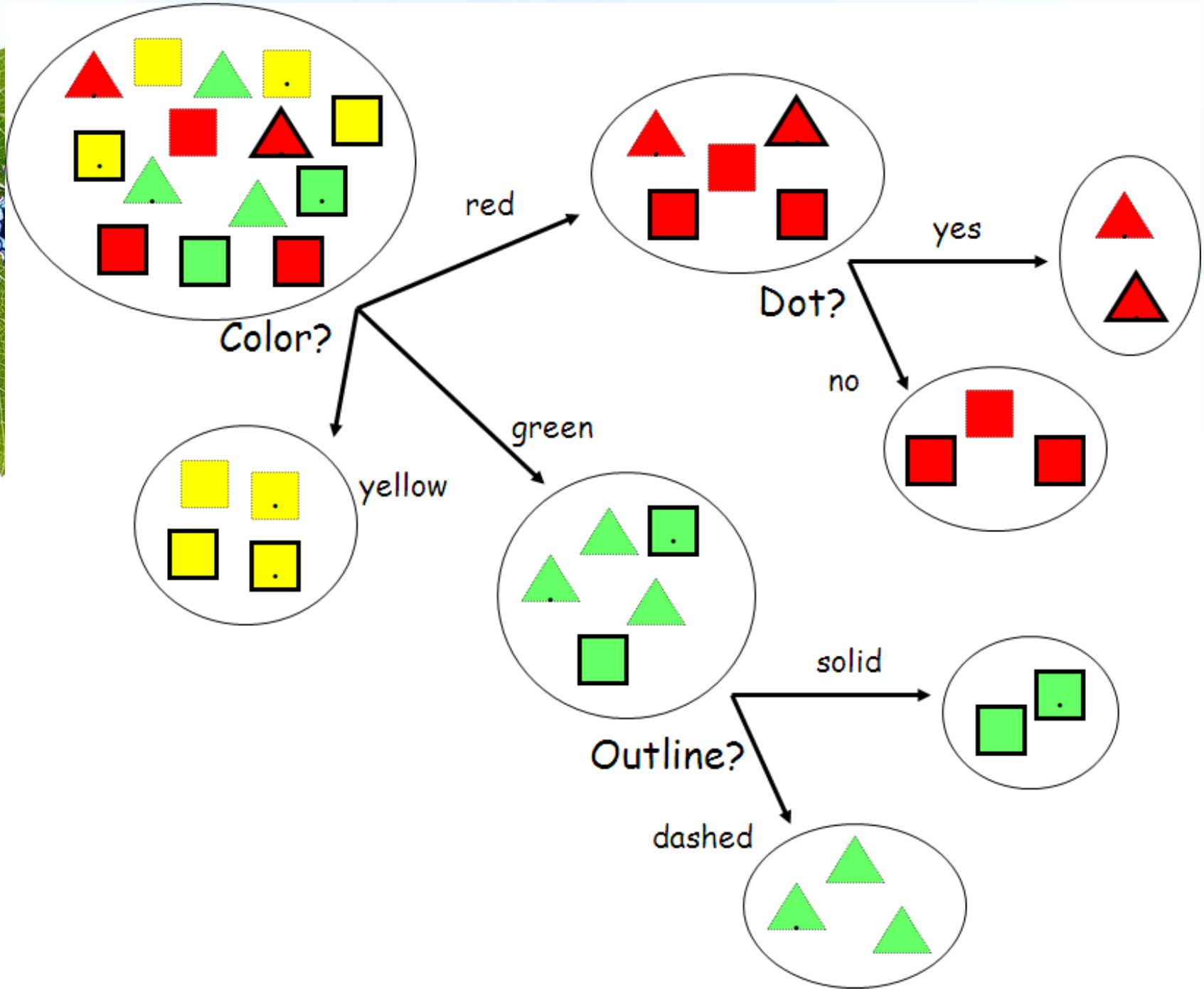
- Attributes
  - Gain(Color) = 0.246
  - Gain(Outline) = 0.151
  - Gain(Dot) = 0.048
- Heuristics: attribute with the highest gain is chosen
- This heuristics is local (local minimization of impurity)



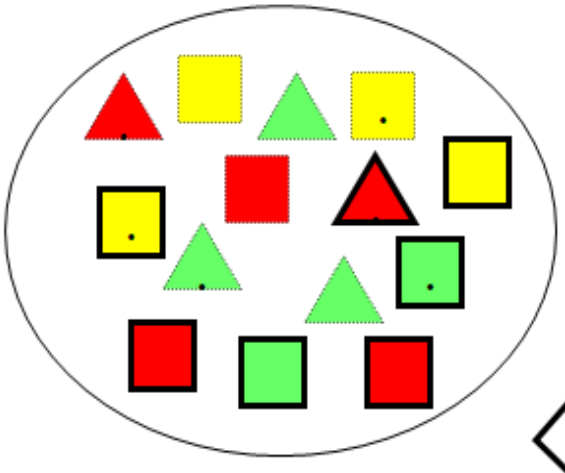
Gain(Outline) = 0.971 - 0 = 0.971 bits

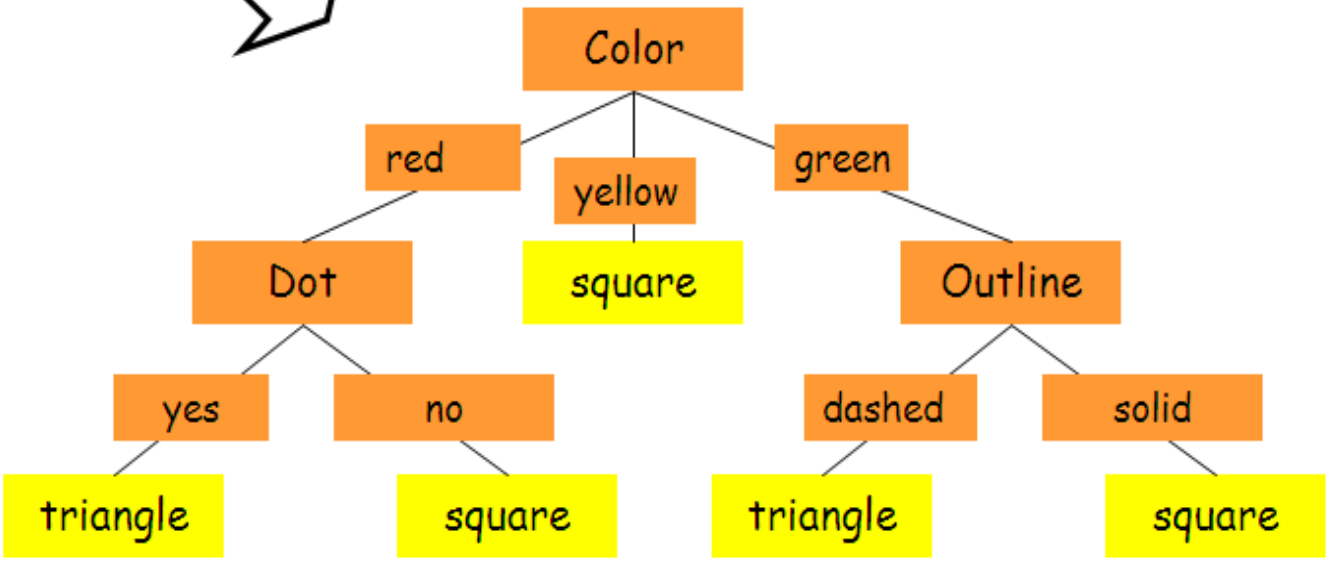
Gain(Dot) = 0.971 - 0.951 = 0.020 bits


red
Color?
green
yellow
Gain(Outline) = 0.971 - 0.951 = 0.020 bits
Gain(Dot) = 0.971 - 0 = 0.971 bits
solid
Outline?
dashed



Decision Tree





A Defect of *Ires*

- *Ires* favors attributes with many values
- Such attribute splits S to many subsets, and if these are small, they will tend to be pure anyway
- One way to rectify this is through a corrected measure of **information gain ratio**.

Information Gain Ratio

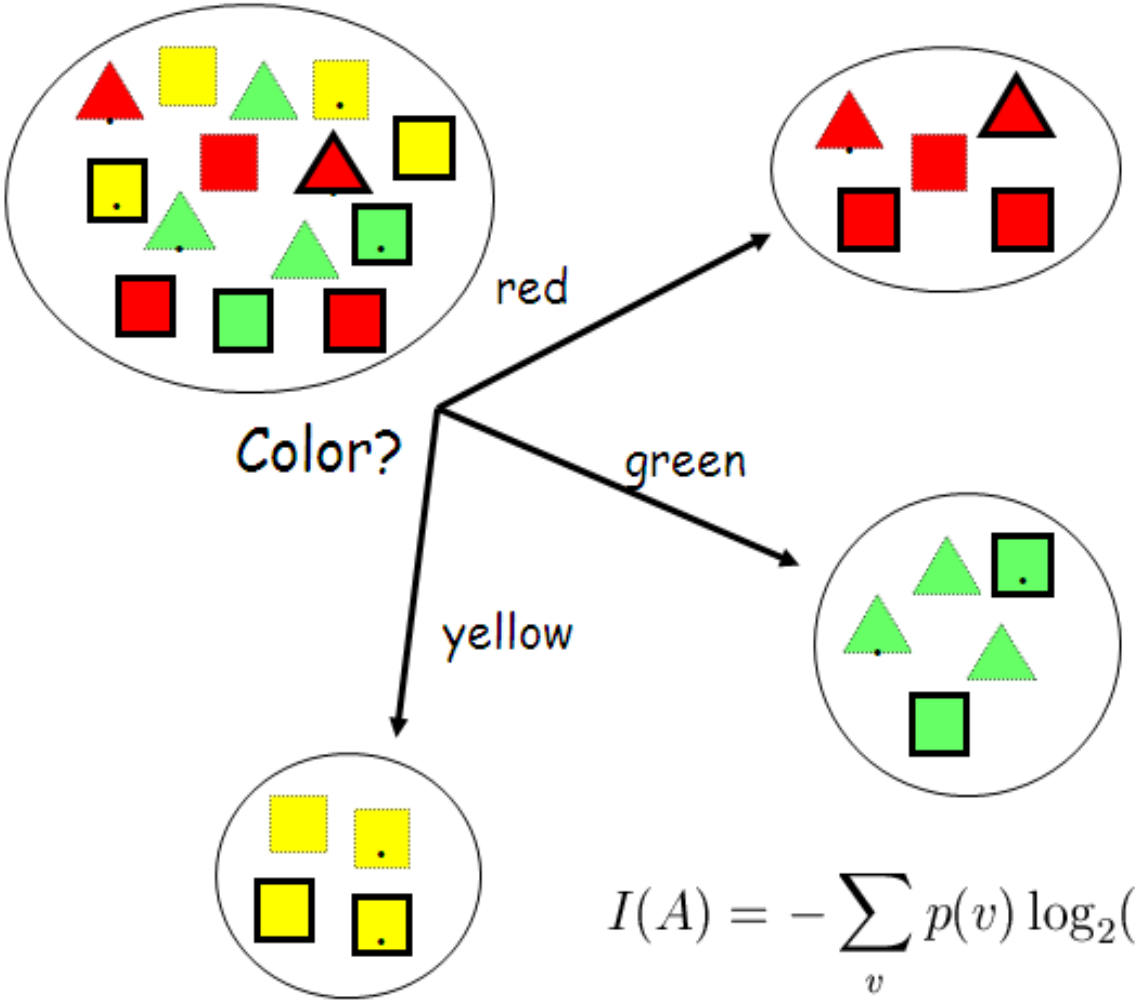
- I(A) is amount of information needed to determine the value of an attribute A

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

- Information gain ratio

$$GainRatio(A) = \frac{Gain(A)}{I(A)} = \frac{I - I_{res}(A)}{I(A)}$$

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

ID3

- The ID3 algorithm is considered as a very simple decision tree algorithm (Quinlan, 1986).
- ID3 uses information gain as splitting criteria. The growing stops when all instances belong to a single value of target feature or when best information gain is not greater than zero.
- ID3 does not apply any pruning procedures nor does it handle numeric attributes or missing values.

C4.5

- C4.5 is an evolution of ID3, presented by the same author (Quinlan, 1993).
- It uses gain ratio as splitting criteria. The splitting ceases when the number of instances to be split is below a certain threshold.
- Error–based pruning is performed after the growing phase.
- C4.5 can handle numeric attributes. It can induce from a training set that incorporates missing values by using corrected gain ratio criteria.

CART (Classification and Regression Trees)

- It is characterized by the fact that it constructs binary trees, namely each internal node has exactly two outgoing edges.
- The splits are selected using the twoing criteria and the obtained tree is pruned by cost–complexity Pruning.
- When provided, CART can consider misclassification costs in the tree induction.
- It also enables users to provide prior probability distribution.

- An important feature of CART is its ability to generate regression trees.
- Regression trees are trees where their leaves predict a real number and not a class.
- Regression trees are needed when the response variable is numeric or continuous.
- In case of regression, CART looks for splits that minimize the prediction squared error (the least–squared deviation). The prediction in each leaf is based on the weighted mean for node.

Advantages of CART

- Can cope with any data structure or type
- Classification has a simple form
- Uses conditional information effectively
- Invariant under transformations of the variables
- Is robust with respect to outliers
- Gives an estimate of the misclassification rate

Disadvantages of CART

- CART does not use combinations of variables
- Tree can be deceptive – if variable not included it could be as it was “masked” by another
- Tree structures may be unstable – a change in the sample may give different trees
- Tree is optimal at each split – it may not be globally optimal.

- Assuming that the P_L, P_R probabilities of right and left nodes, t_L, t_R right and left nodes,

- The *twoing rule*: At a node t, choose the split s that maximizes

$$\frac{p_L p_R}{4}\left[\sum_j |p(j \mid t_L) - p(j \mid t_R)|\right]^2 .$$

- Although Twoing splitting rule allows us to build more balanced trees, this algorithm works slower than Gini rule. For example, if the total number of classes is equal to K, than we will have 2^{K-1} possible splits.

- Regression trees do not have classes. Instead there is response vector Y which represents the response values for each observation in variable matrix X.

- Since regression trees do not have pre-assigned classes, classification splitting rules like Gini or Twoing can not be applied.

- Splitting in regression trees is made in accordance with squared residuals minimization algorithm which implies that expected sum variances for two resulting nodes should be minimized.

$$\underset{x_j \leq x_j^R,\ j=1,\ldots,M}{\arg\min} \left[P_l \mathrm{Var}(Y_l) + P_r \mathrm{Var}(Y_r) \right]$$

where $\mathrm{Var}(Y_l)$,$\mathrm{Var}(Y_r)$ - reponse vectors for corresponding left and right child nodes;

Variables

- A categorical variable (sometimes called a nominal variable) is one that has two or more categories, but there is no intrinsic ordering to the categories
  - Hair color (blonde, brown, brunette, red, etc.)

- An ordinal variable is similar to a categorical variable. The difference between the two is that there is a clear ordering of the variables.
  - economic status (low, medium and high).

- An interval variable is similar to an ordinal variable, except that the intervals between the values of the interval variable are equally spaced.
  - annual income ($10,000, $15,000 and $20,000) where the size of that interval between two people is same ($5,000)