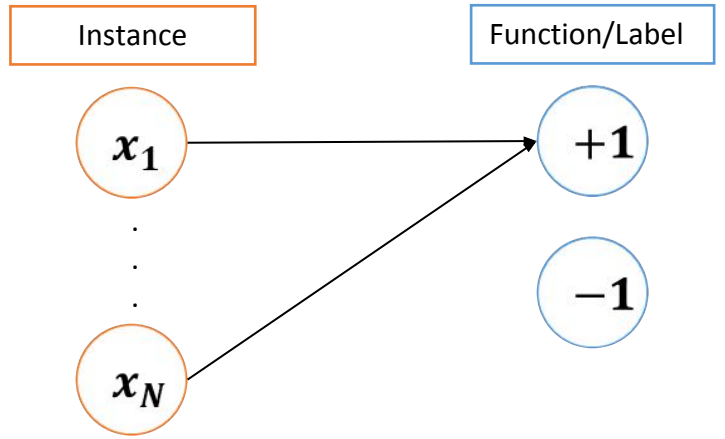
Introduction to Classification

Classification Problem

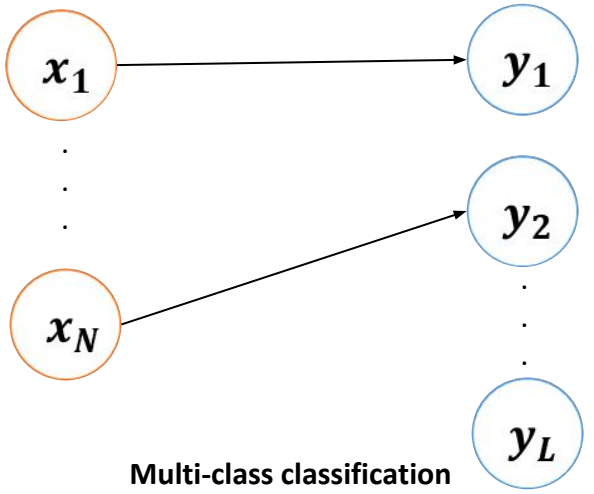
- A machine learning task that deals with identifying the class to which an instance belongs
- A classifier performs classification

Classification Problem

- **Classical Classification**
  - The classes are exclusive, if an example belongs to one class, it can't be belonging to others
  - We assign one class label l_i from a set $L = \{l_1, \ldots, l_k\}$ to each example
- **Binary Classification**
  - |L| = 2
  - Classification of email into spam and non-spam
- **Multi-class Classification**
  - |L| > 2
  - Classification of email into one of the predefined classes such as *primary, social, promotions, updates, forum, etc.*



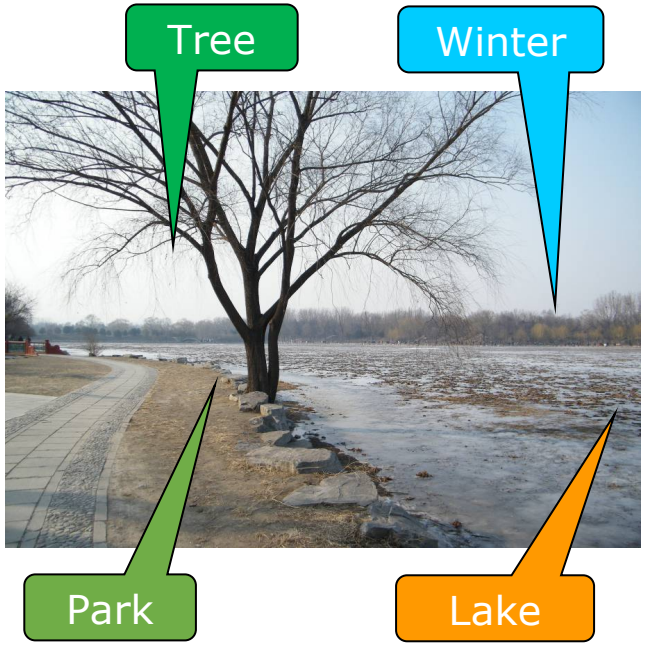
Binary classification



Multi-class classification

Multi-Label Classification

- However, in many real-world classification tasks, the data object can simultaneously belong to one or more classes in L
- Example
  - In protein function prediction, a protein can be associated with a set of functional role such as metabolism, energy, cell fate, storage protein, localization
  - In web mining, a web page can be classified as news, academic, e-commerce, blog, forum etc.
  - Similarly, in image classification, an image can be annotated with several classes such as sea, sky, tree, mountain, valley, and so on



Single-label classification: Is this a picture of a tree?

$L \in \{yes, no\}$

Multi-label classification: Which labels are relevant to this picture?

$L \subseteq \{tree, lake, sea, boat, park, winter\}$

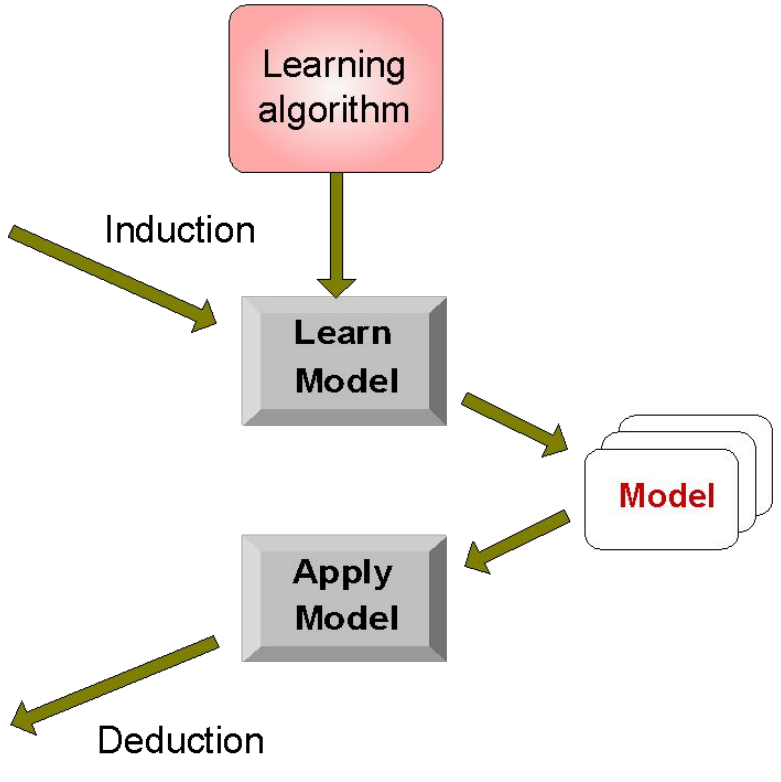
Illustrating Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

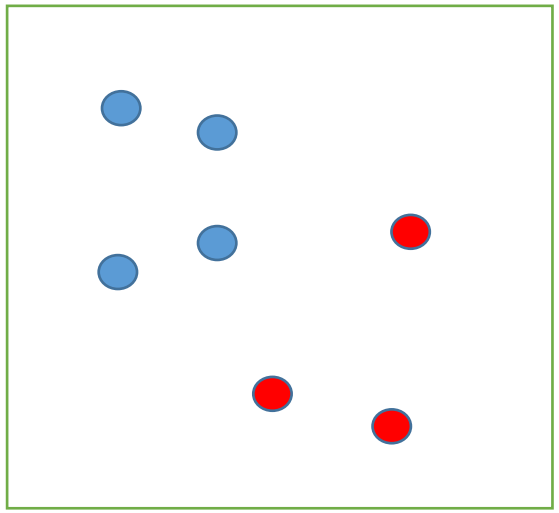
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



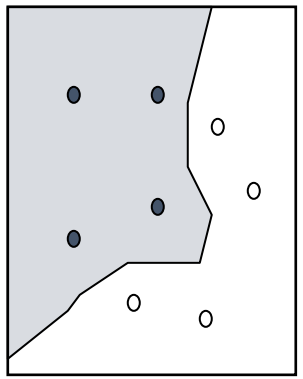
Discriminant Function

- One way to represent a classifier is by using
  - Discriminant functions which defines the decision boundary
- Idea
  - For every class $i = 0,1, \ldots k$ define a function $f_i(x)$ mapping $X \rightarrow \mathcal{R}$
  - When the decision on input x should be made choose the class with the highest value of

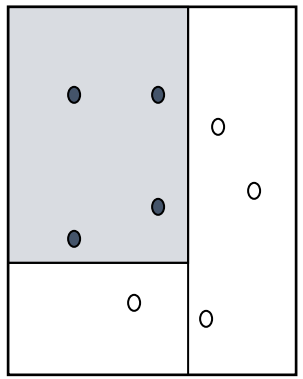


Discriminant Function

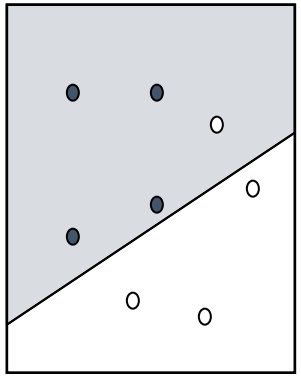
- **It can be arbitrary functions of *x*, such as:**



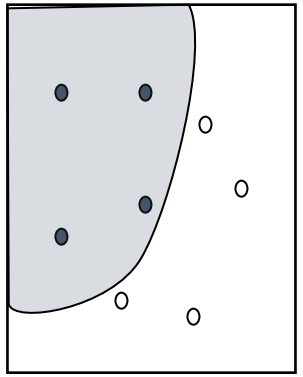
Nearest Neighbor



Decision Tree



Linear Functions



Nonlinear Functions

Nearest-Neighbor Classifier

k-NN Approach

- The simplest, most used instance-based learning algorithm is the *k*-NN algorithm
- *k*-NN assumes that all instances are points in some *d*-dimensional space and defines neighbors in terms of distance
  - Where *k* is the number of neighbors considered

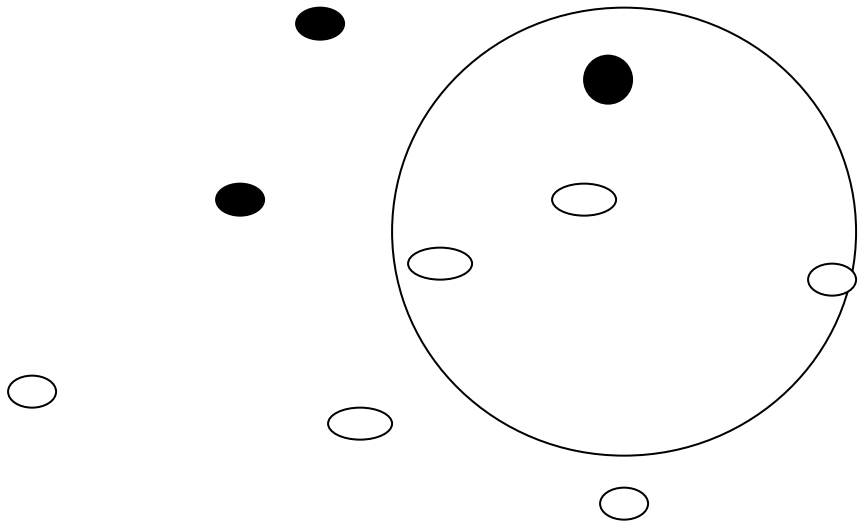
k-NN Approach

- Requires three things:
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of k, the number of nearest neighbors to retrieve
- To classify an unknown record:
  - Compute distance to other training records
  - Identify k nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

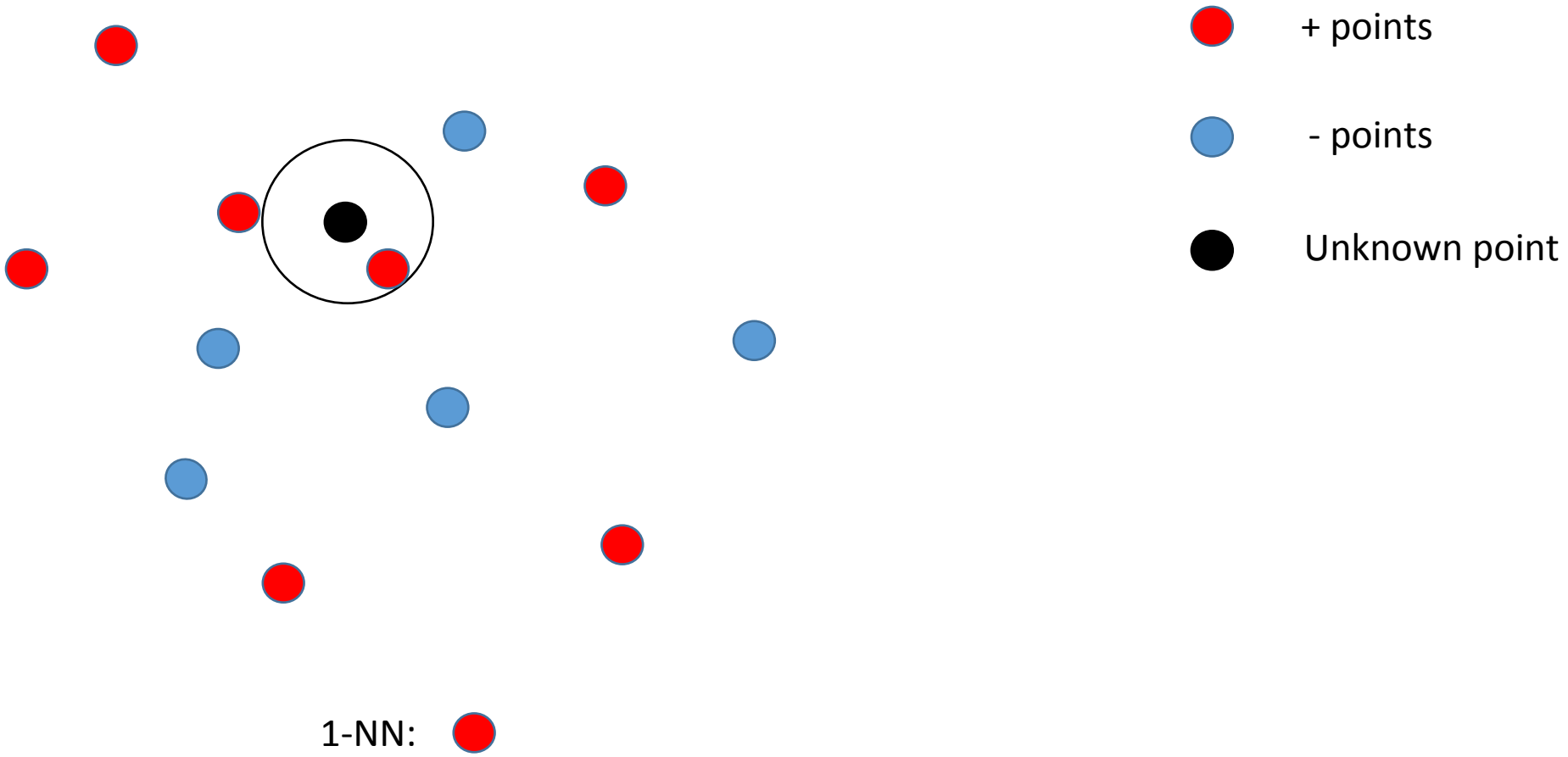
Source: Nearest-Neighbor Classifier MTL 782 IIT DELHI

Definition of Nearest Neighbor

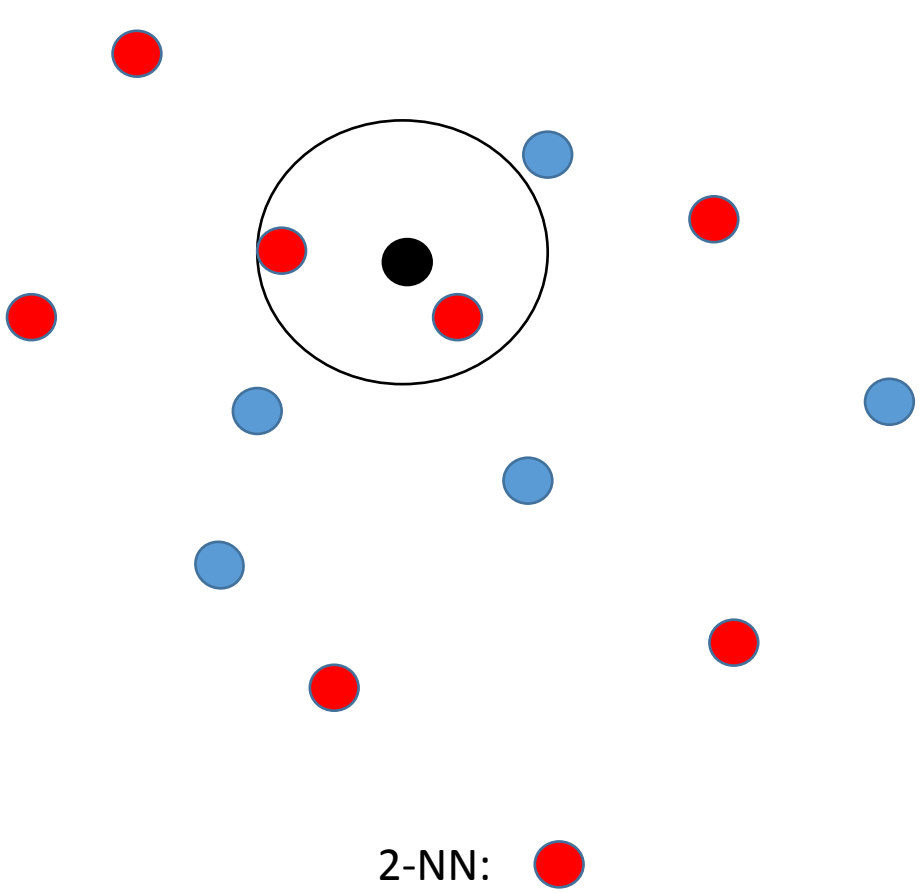
- K-nearest neighbors of a record x are data points that have the k smallest distance to x



k-NN Visualization



k-NN Visualization

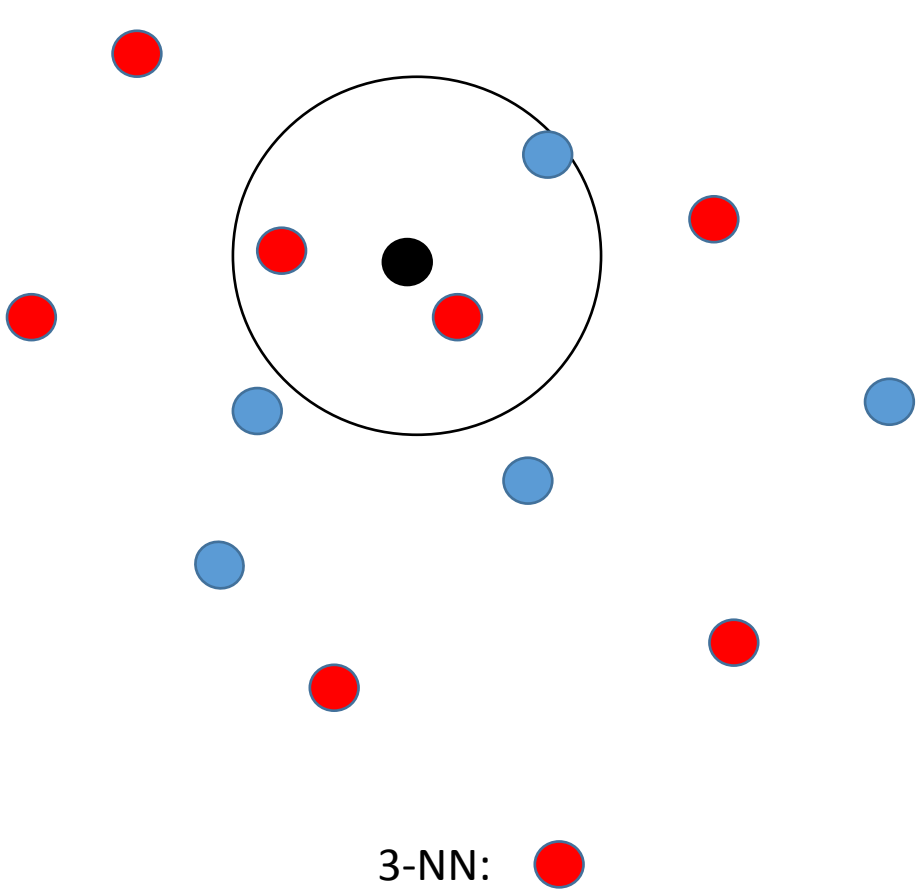


+ points

- points

Unknown point

k-NN Visualization

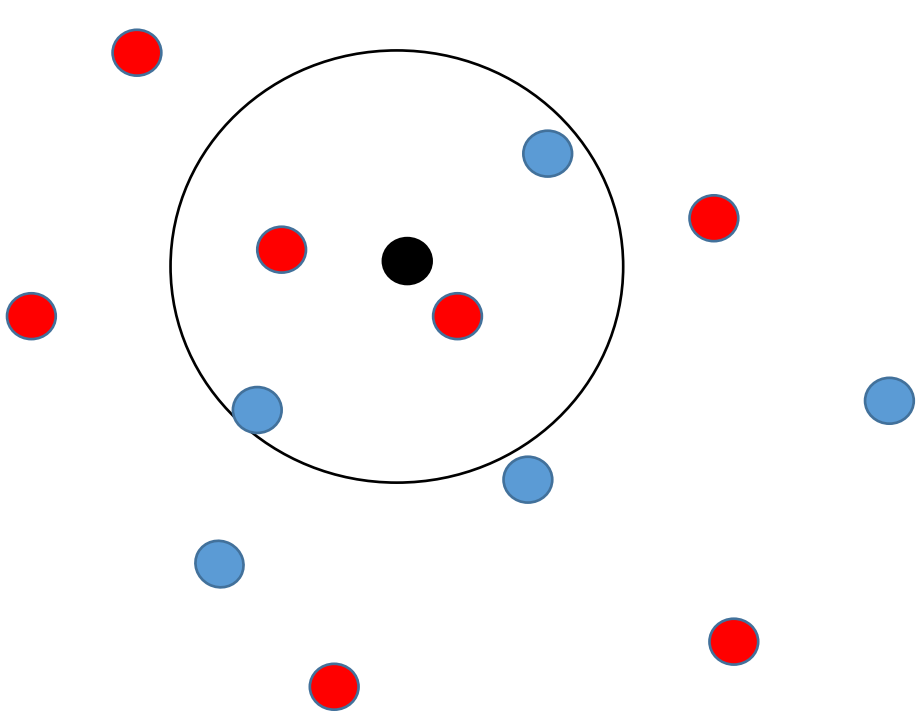


+ points

- points

Unknown point

k-NN Visualization



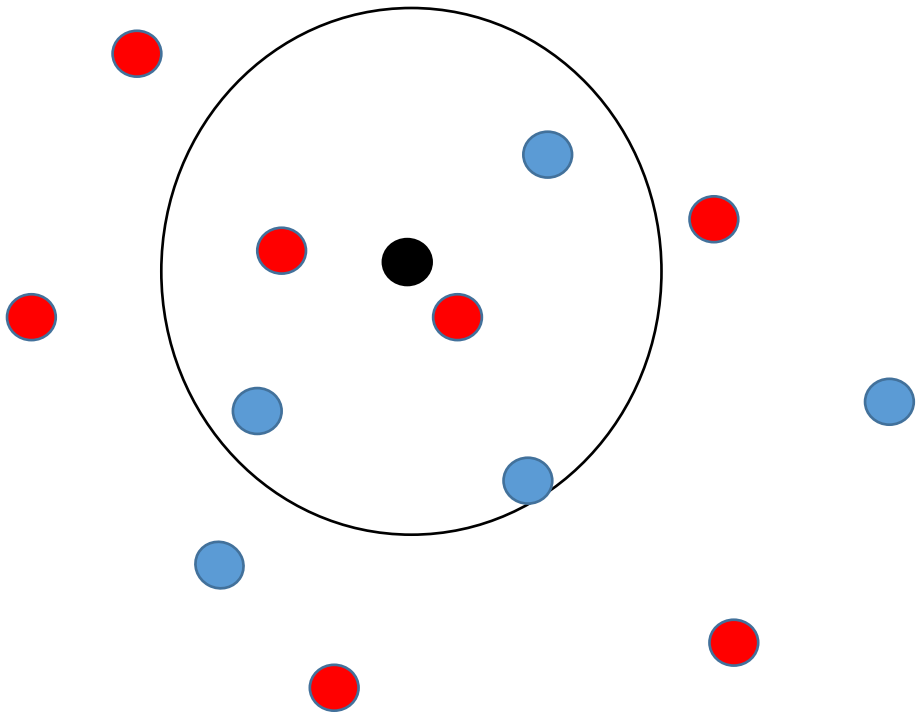
+ points

- points

Unknown point

4-NN: /

k-NN Visualization



+ points

- points

Unknown point

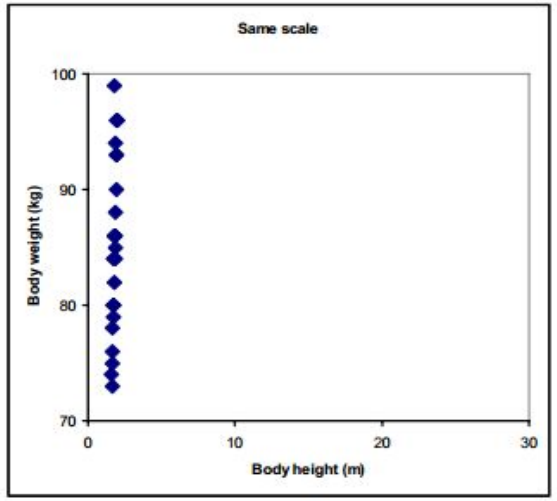
5-NN:

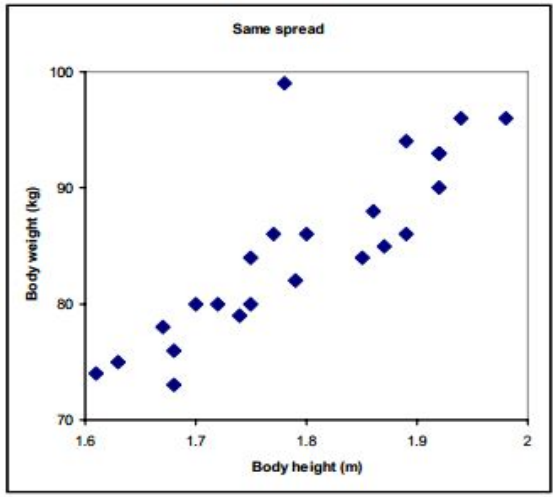
Standardization

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
- Example:
  - height of a person may vary from 1.5m to 1.8m
  - weight of a person may vary from 60 KG to 100KG
  - income of a person may vary from Rs10K to Rs 2 Lakh
- Transform raw feature values into z-scores

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

  - x_{ij} is the value for the *i*th sample and *j*th feature
  - μ_j is the average of *j*th feature
  - σ_j is the standard deviation of *j*th feature





KNN Example

	Food (3)	Chat (2)	Fast (2)	Price (3)	Bar (2)	BigTip
1	great	yes	yes	normal	no	yes
2	great	no	yes	normal	no	yes
3	mediocre	yes	no	high	no	no
4	great	yes	yes	normal	yes	yes

- **Example 1 (great, no, no, normal, no) : yes**
  - **Most similar: number 2 (1 mismatch, 4 match): yes**
  - **Second most similar example: number 1 (2 mismatch, 3 match): yes**

Example 2 (mediocre, yes, no, normal, no): yes/no

- **Most similar: number 3 (1 mismatch, 4 match): no**
- **Second most similar example: number 1 (2 mismatch, 3 match): yes**

Source: CS 4700: Foundations of Artificial Intelligence, Carla P. Gomes

k-NN Time Complexity

- Suppose there are n instances and d features in the dataset
- Nearest neighbor algorithm requires computing n distances
- Each distance computation involves scanning through each feature value
- Running time complexity is proportional to $n \times d$

k-NN variations

- Value of k
  - Larger k increases confidence in prediction
  - Note that if k is too large, decision may be skewed
  - Smaller k leads to unstable decision boundary
- Weighted evaluation of nearest neighbors
  - Plain majority may unfairly skew decision
  - Revise algorithm so that closer neighbors have greater “vote weight”
- Other distance measures

Other distance measures

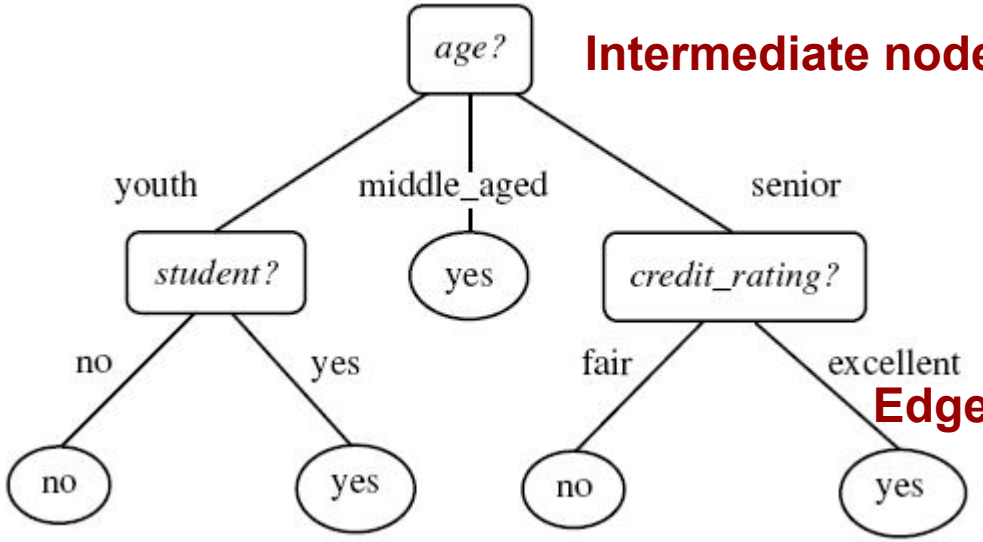
- City-block distance (Manhattan dist)
  - Add absolute value of differences
- Cosine similarity
  - Measure angle formed by the two samples (with the origin)
- Jaccard distance
  - Determine percentage of exact matches between the samples (not including unavailable data)
- Others

Some Remarks

- ***k*-NN works well on many practical problems and is fairly noise tolerant (depending on the value of *k*)**
- ***k*-NN is subject to the curse of dimensionality (i.e., presence of many irrelevant attributes)**
- ***k*-NN needs adequate distance measure**

Decision Tree

Example tree



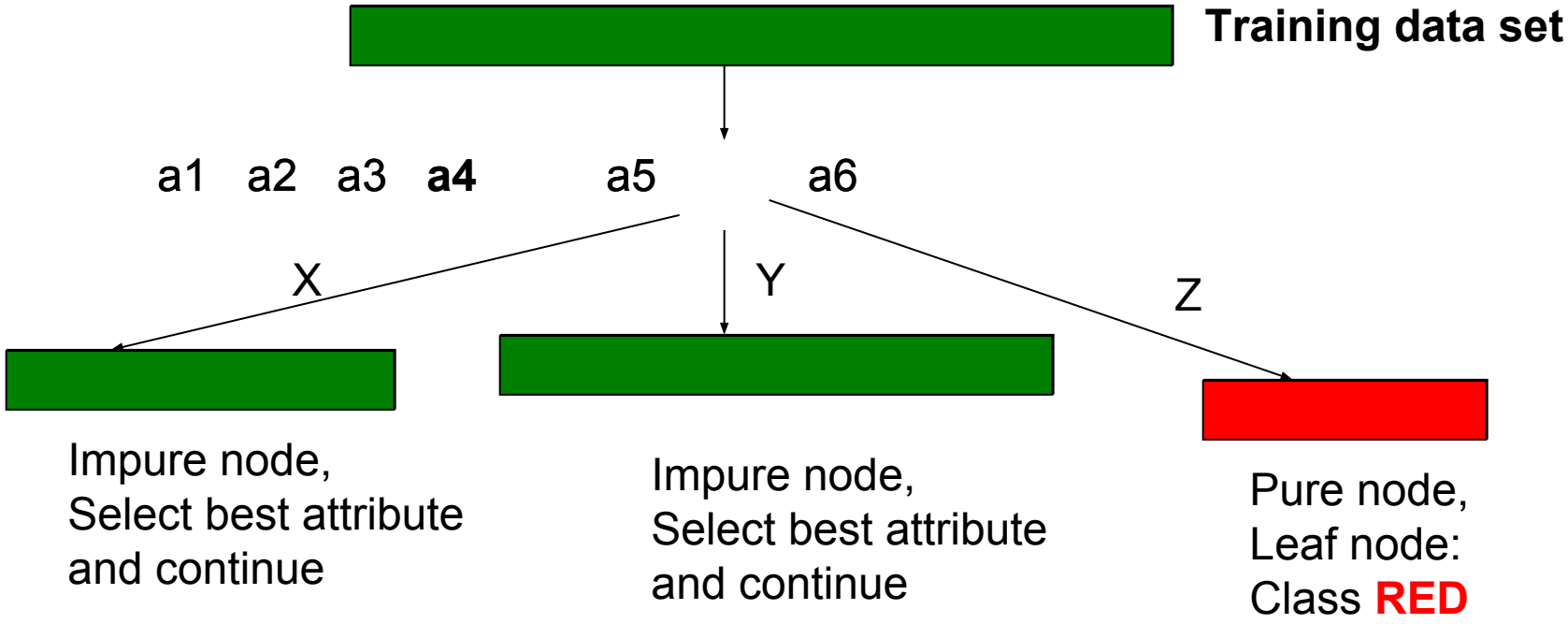
Intermediate nodes : Attributes value tests

Edges : Attribute value

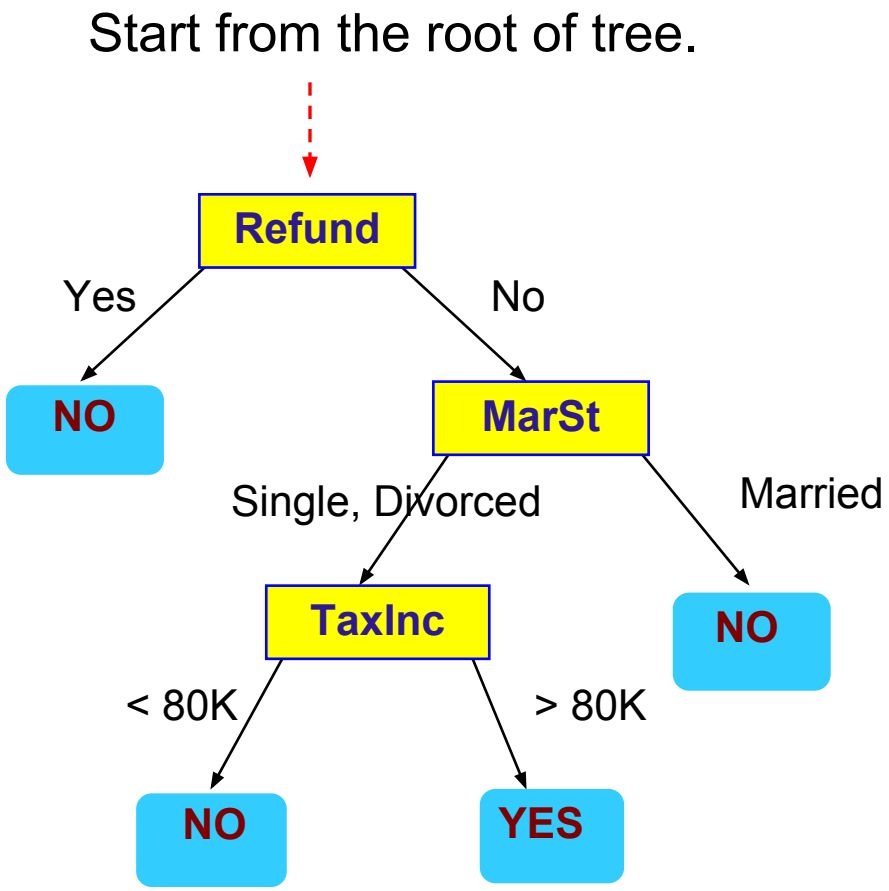
Leaf nodes : Class predictions

Example algorithms: ID3, SPRINT, CART

Decision Tree schematic



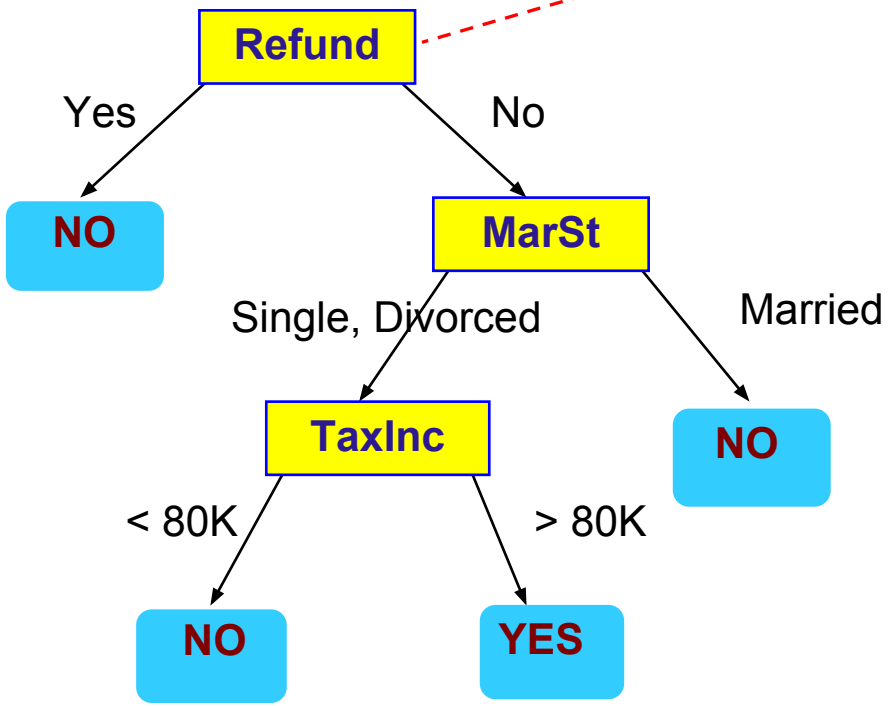
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

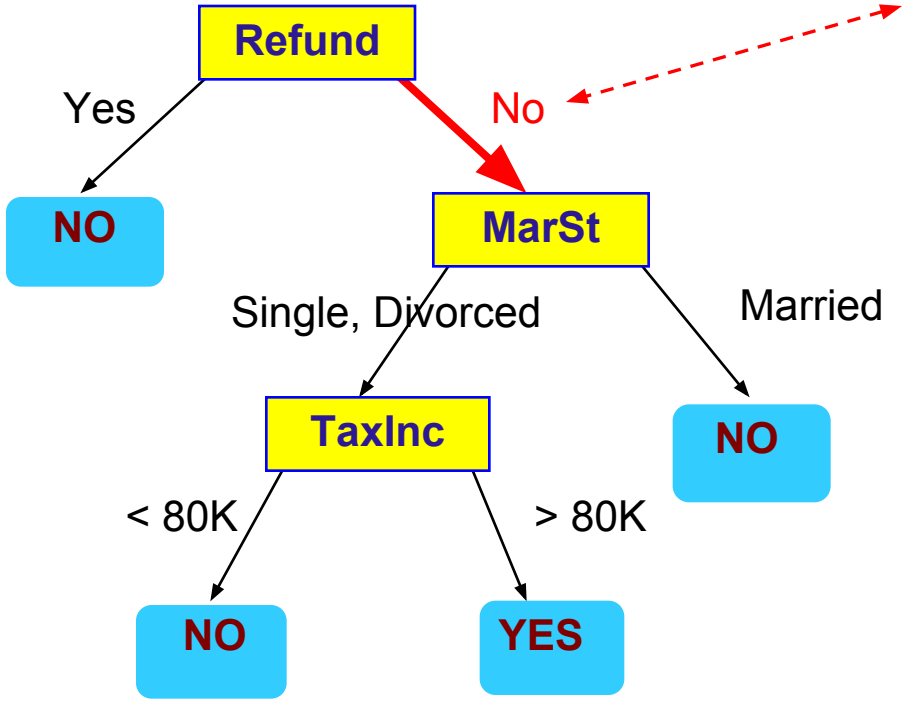
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

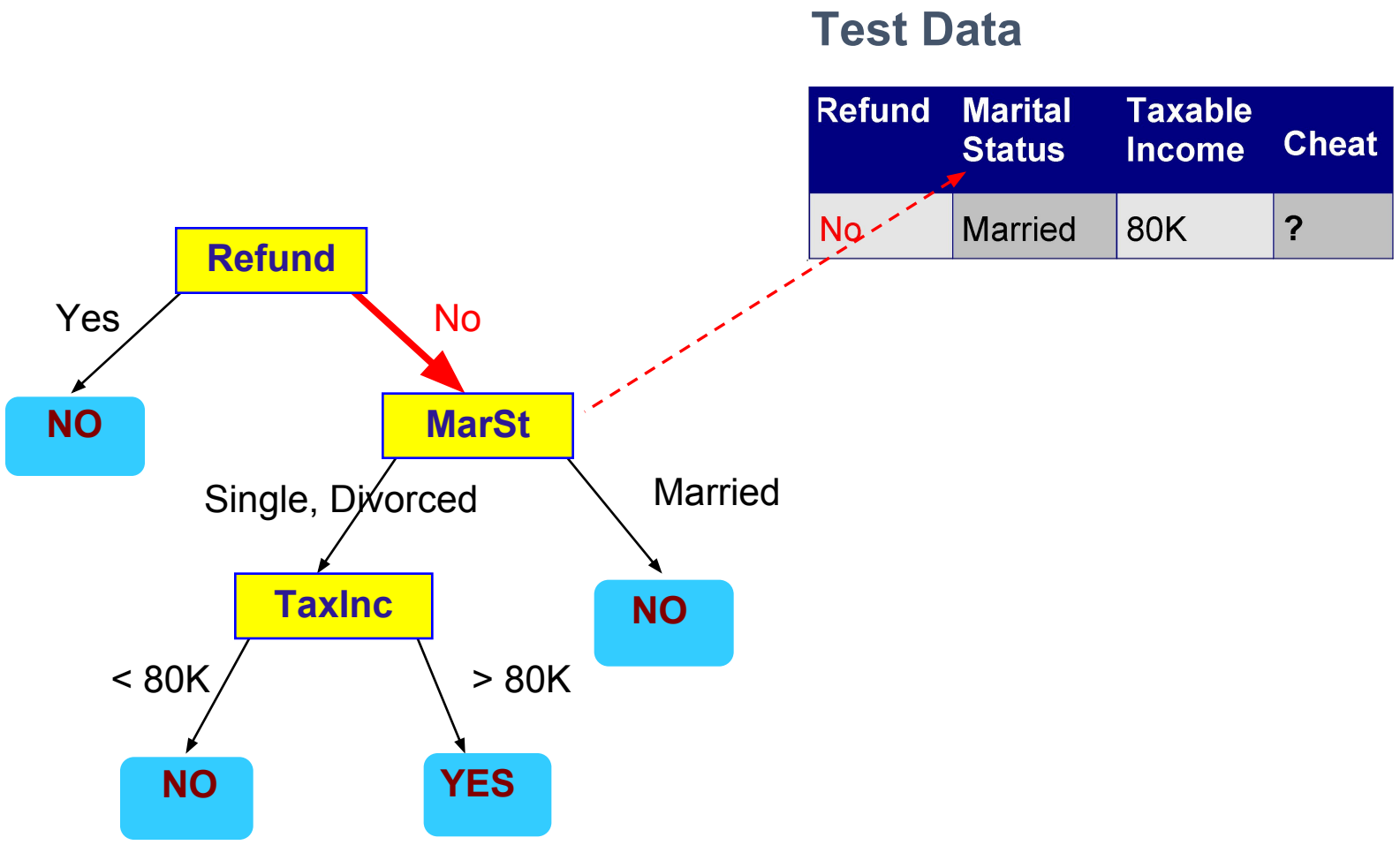
Apply Model to Test Data



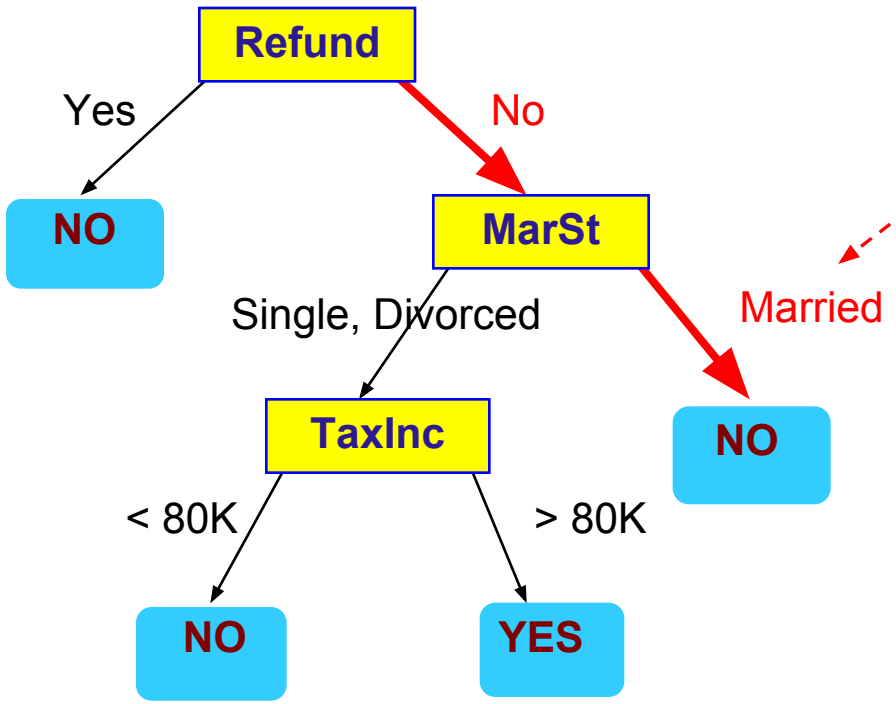
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



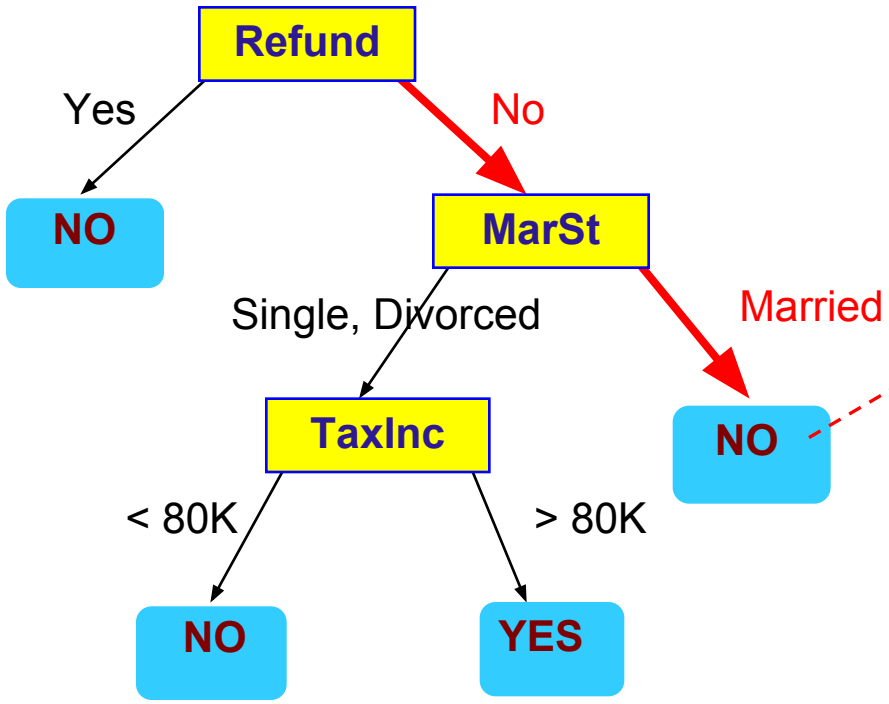
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to “No”

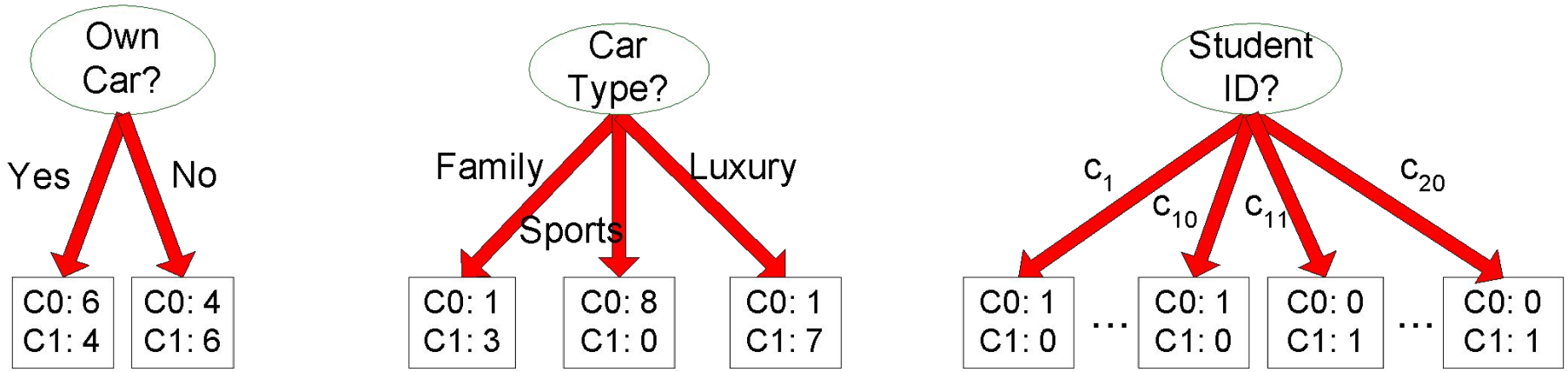
How to Build Decision Trees

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

How to determine the Best Split

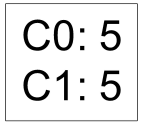
Before Splitting: 10 records of class 0,
10 records of class 1



Which test condition is the best?

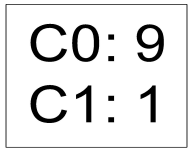
How to determine the Best Split

- Greedy approach:
  - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:



Non-homogeneous,

High degree of impurity



Homogeneous,

Low degree of impurity

Measures of Node Impurity

- Entropy
- Gini Index
- Misclassification error

Entropy

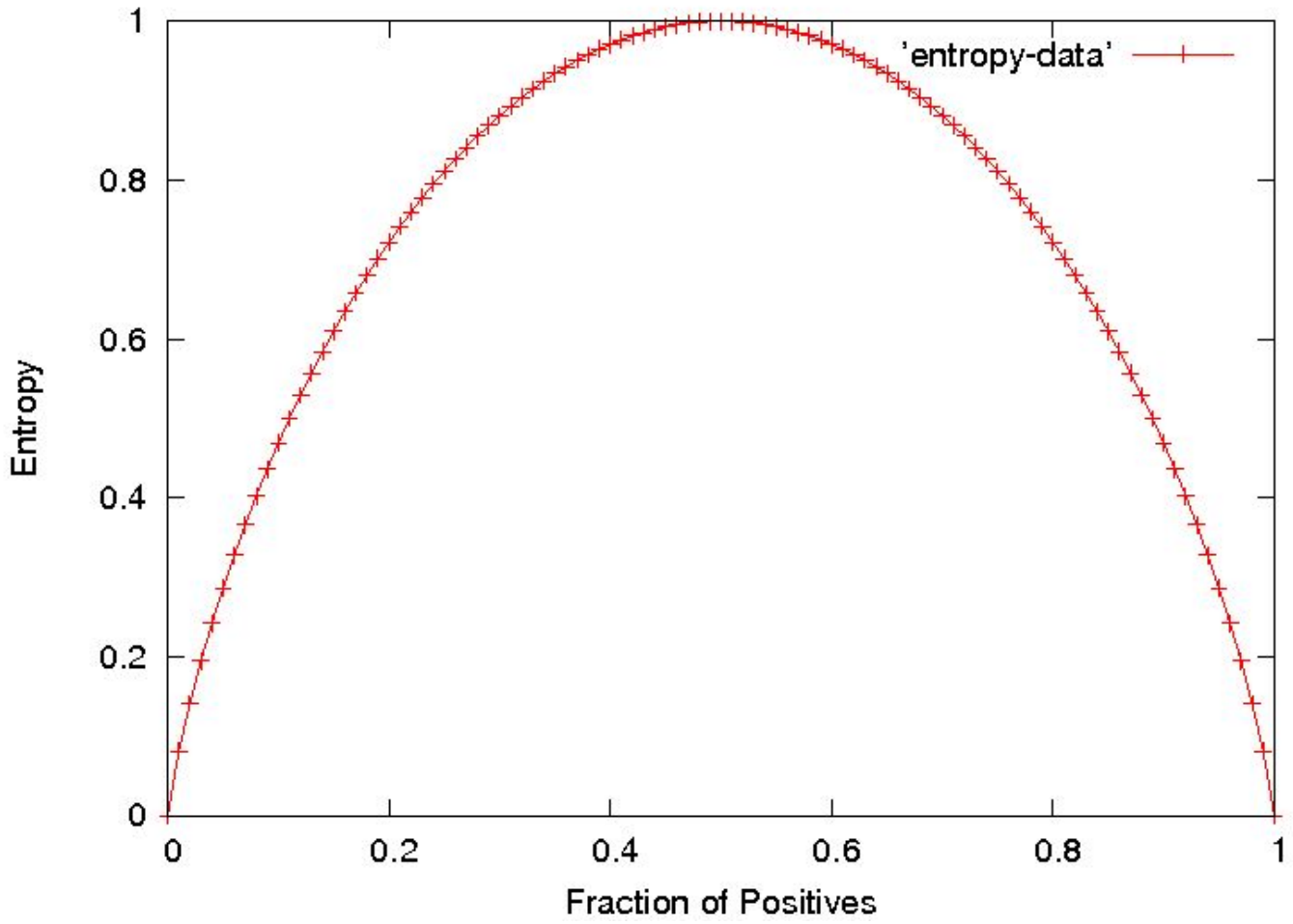
- Entropy (impurity) of a set of examples S, relative to a binary classification is:

$$Entropy(S) = -p_1 \log_2(p_1) - p_0 \log_2(p_0)$$

- where p_1 is the fraction of positive examples in S and p_0 is the fraction of negatives.
- If all examples are in one category, entropy will be zero
- If examples are equally mixed $(p_1 = p_0 = 0.5)$, entropy is a maximum of 1.
- For multi-class problems with c categories, entropy generalizes to:

$$Entropy(S) = \sum_{i=1}^{c} - p_i \log_2(p_i)$$

Entropy Plot for Binary Classification



Information Gain

- The information gain of an attribute *F* is the expected reduction in entropy resulting from splitting on this feature.

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

where S_v is the subset of *S* having value *v* for feature *F*.

- Entropy of each resulting subset weighted by its relative size.

Example

$$E(S) = -P_{\oplus} \log_2 P_{\oplus} - P_{\ominus} \log_2 P_{\ominus}$$

$$E(S) = [9+, 5-] = -\left(\frac{9}{14}\right) \log_2 \left(\frac{9}{14}\right) - \left(\frac{5}{14}\right) \log_2 \left(\frac{5}{14}\right)$$

$$E(S) = 0.94$$

$$E(\text{Outlook}_{\text{sunny}}) = [2+, 3-] = -\left(\frac{2}{5}\right) \log_2 \left(\frac{2}{5}\right) - \left(\frac{3}{5}\right) \log_2 \left(\frac{3}{5}\right) = 0.971$$

$$E(\text{Outlook}_{\text{overcast}}) = [4+, 0-] = -\left(\frac{4}{4}\right) \log_2 \left(\frac{4}{4}\right) - \left(\frac{0}{4}\right) \log_2 \left(\frac{0}{4}\right) = 0$$

$$E(\text{Outlook}_{\text{rainy}}) = [3+, 2-] = -\left(\frac{3}{5}\right) \log_2 \left(\frac{3}{5}\right) - \left(\frac{2}{5}\right) \log_2 \left(\frac{2}{5}\right) = 0.971$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

S: Data set

Example

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

$$Gain(S, \text{Outlook}) = 0.94 - \frac{|S_{sunny}|}{|S|} Entropy(\text{Outlook}_{\text{sunny}}) - \frac{|S_{overcast}|}{|S|} Entropy(\text{Outlook}_{overcast}) - \frac{|S_{rainy}|}{|S|} Entropy(\text{Outlook}_{\text{rainy}})$$

$$Gain(S, \text{Outlook}) = 0.94 - \frac{5}{14} * 0.971 - \frac{4}{14} * 0 - \frac{5}{14} * 0.971$$

$$= 0.2465$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

S: Data set

Example

$$E(\text{Temperature}_{\text{hot}}) = [2+, 2-] = -\left(\frac{2}{4}\right)\log_2\left(\frac{2}{4}\right) - \left(\frac{2}{4}\right)\log_2\left(\frac{2}{4}\right) = 1$$

$$E(\text{Temperature}_{mild}) = [4+, 2-] = -\left(\frac{4}{6}\right)\log_2\left(\frac{4}{6}\right) - \left(\frac{2}{6}\right)\log_2\left(\frac{2}{6}\right) = 0.918$$

$$E(\text{Temperature}_{\text{cold}}) = [3+, 1-] = -\left(\frac{3}{4}\right)\log_2\left(\frac{3}{4}\right) - \left(\frac{1}{4}\right)\log_2\left(\frac{1}{4}\right) = 0.811$$

$$Gain(S, \text{Temperature}) = 0.94 - \frac{4}{14} * 1 - \frac{6}{14} * 0.918 - \frac{4}{14} * 0.811$$
$$= 0.0291$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

S: Data set

Example

$$E(\text{Humidity}_{\text{high}}) = [3+, 4-] = -\left(\frac{3}{7}\right)\log_2\left(\frac{3}{7}\right) - \left(\frac{4}{7}\right)\log_2\left(\frac{4}{7}\right) = 0.985$$

$$E(\text{Humidity}_{\text{normal}}) = [6+, 1-] = -\left(\frac{6}{7}\right)\log_2\left(\frac{6}{7}\right) - \left(\frac{1}{7}\right)\log_2\left(\frac{1}{7}\right) = 0.592$$

$$Gain(S, \text{Humidity}) = 0.94 - \frac{7}{14} * 0.985 - \frac{7}{14} * 0.592$$
$$= 0.155$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

S: Data set

Example

$$E(\text{Wind}_{\text{weak}}) = [6+, 2-] = -\left(\frac{6}{8}\right)\log_2\left(\frac{6}{8}\right) - \left(\frac{2}{8}\right)\log_2\left(\frac{2}{8}\right) = 0.811$$

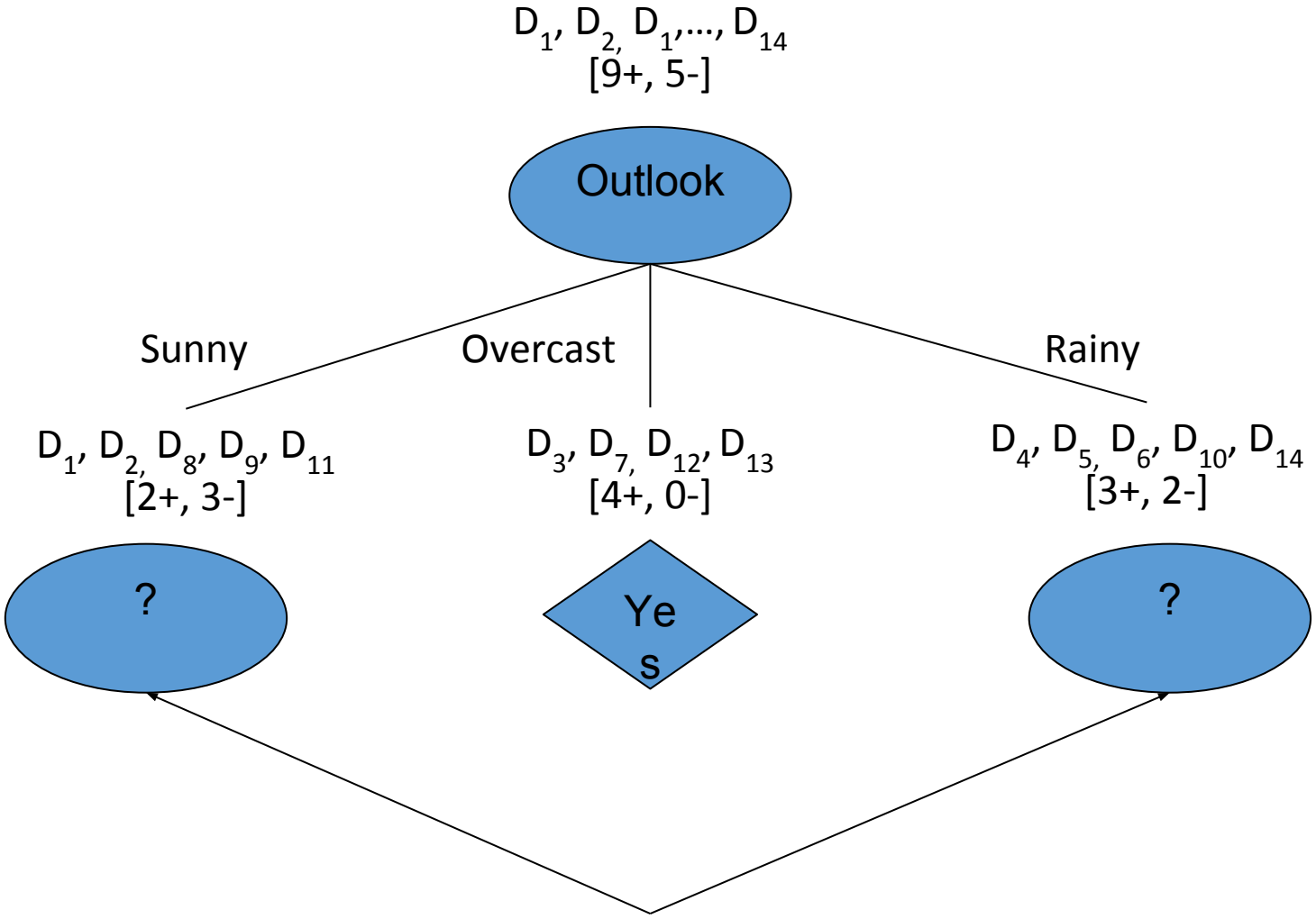
$$E(\text{Wind}_{\text{strong}}) = [3+, 3-] = -\left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) - \left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) = 1$$

$$Gain(S, \text{Wind}) = 0.94 - \frac{8}{14} * 0.811 - \frac{6}{14} * 1 = 0.0480$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

S: Data set

Example



Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

S: Data set

Example

$$\mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}}) = [2+, 3-] = -\left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) - \left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) = 0.971$$

$$\mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}} \wedge \mathrm{Temperature}_{\mathrm{hot}}) = [0+, 2-] = 0$$

$$\mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}} \wedge \mathrm{Temperature}_{\mathrm{Mild}}) = [1+, 1-] = 1$$

$$\mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}} \wedge \mathrm{Temperature}_{\mathrm{cool}}) = [1+, 0-] = 0$$

$$Gain(\mathrm{Outlook}_{\mathrm{sunny}}, Temperature) = \mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}})$$

$$-\frac{|Outlook_{sunny} \wedge \mathrm{Temperature}_{\mathrm{hot}}|}{|Outlook_{sunny}|} Entropy(\mathrm{Outlook}_{\mathrm{sunny}} \wedge \mathrm{Temperature}_{\mathrm{hot}})$$

$$-\frac{|Outlook_{sunny} \wedge \mathrm{Temperature}_{\mathrm{mild}}|}{|Outlook_{sunny}|} Entropy(\mathrm{Outlook}_{\mathrm{sunny}} \wedge \mathrm{Temperature}_{\mathrm{mild}})$$

$$-\frac{|Outlook_{sunny} \wedge \mathrm{Temperature}_{\mathrm{cool}}|}{|Outlook_{sunny}|} Entropy(\mathrm{Outlook}_{\mathrm{sunny}} \wedge \mathrm{Temperature}_{\mathrm{cool}})$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

S: Data set

Example

$$Gain(\text{Outlook}_{\text{sunny}}, Temperature) =$$

$$= 0.971 - \frac{2}{5} * 0 - \frac{2}{5} * 1 - \frac{1}{5} * 0 = 0.571$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

S: Data set

Example

$$E(\text{Outlook}_{\text{sunny}}) = [2+, 3-] = -\left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) - \left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) = 0.971$$

$$E(\text{Outlook}_{\text{sunny}} \wedge \textit{Humidity}_{\textit{high}}) = [0+, 3-] = 0$$

$$E(\text{Outlook}_{\text{sunny}} \wedge \text{Humidity}_{\text{normal}}) = [2+, 0-] = 0$$

$$\textit{Gain}(\text{Outlook}_{\text{sunny}}, \text{Humidity}) = E(\text{Outlook}_{\text{sunny}})$$

$$-\frac{|\textit{Outlook}_{\textit{sunny}} \wedge \textit{Humidity}_{\textit{high}}|}{|\textit{Outlook}_{\textit{sunny}}|}\textit{Entropy}(\text{Outlook}_{\text{sunny}} \wedge \textit{Humidity}_{\textit{high}})$$

$$-\frac{|\textit{Outlook}_{\textit{sunny}} \wedge \text{Humidity}_{\text{normal}}|}{|\textit{Outlook}_{\textit{sunny}}|}\textit{Entropy}(\text{Outlook}_{\text{sunny}} \wedge \text{Humidity}_{\text{normal}})$$

$$\textit{Gain}(\text{Outlook}_{\text{sunny}}, \text{Humidity}) = 0.971 - \frac{3}{5} * 0 - \frac{2}{5} * 0 = 0.971$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

S: Data set

Example

$$\mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}}) = [2+, 3-] = -\left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) - \left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) = 0.971$$

$$\mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}} \wedge Wind_{weak}) = [1+, 2-] = 0.918$$

$$\mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}} \wedge Wind_{strong}) = [1+, 1-] = 1$$

$$Gain(\mathrm{Outlook}_{\mathrm{sunny}}, \mathrm{wind}) = \mathrm{E}(\mathrm{Outlook}_{\mathrm{sunny}})$$

$$-\frac{|Outlook_{sunny} \wedge Wind_{weak}|}{|Outlook_{sunny}|} Entropy(\mathrm{Outlook}_{\mathrm{sunny}} \wedge Wind_{weak})$$

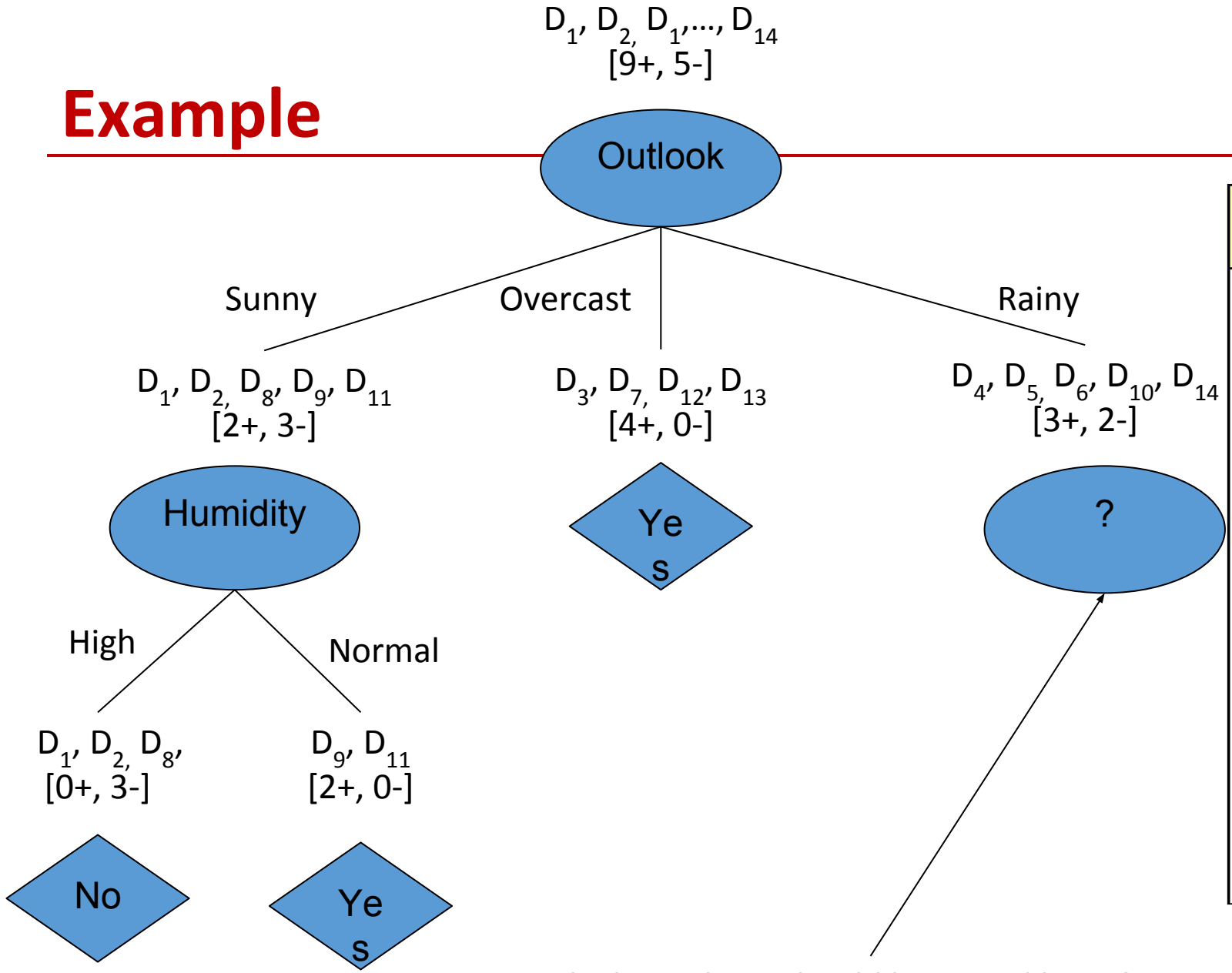
$$-\frac{|Outlook_{sunny} \wedge Wind_{strong}|}{|Outlook_{sunny}|} Entropy(\mathrm{Outlook}_{\mathrm{sunny}} \wedge Wind_{strong})$$

$$\mathrm{Gain}(\mathrm{Outlook}_{\mathrm{sunny}}, \mathrm{Wind}) = 0.971 - \frac{3}{5} * 0.918 - \frac{2}{5} * 1 = 0.020$$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

S: Data set

Example



Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

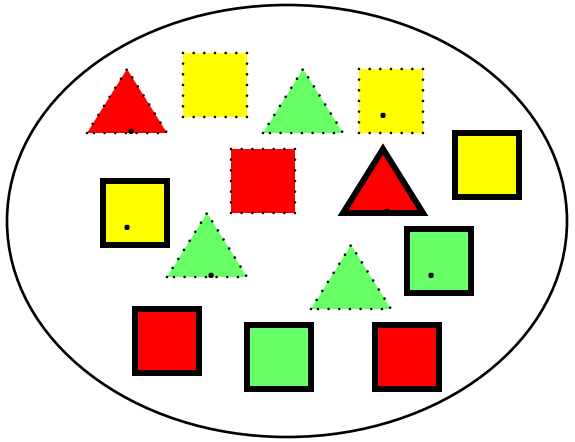
S: Data set

Which attribute should be tested here ?

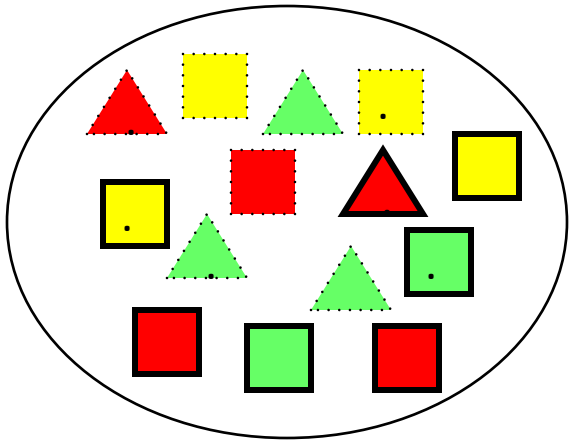
Triangles and Squares

#	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



Entropy



- 5 triangles
- 9 squares
- class probabilities

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

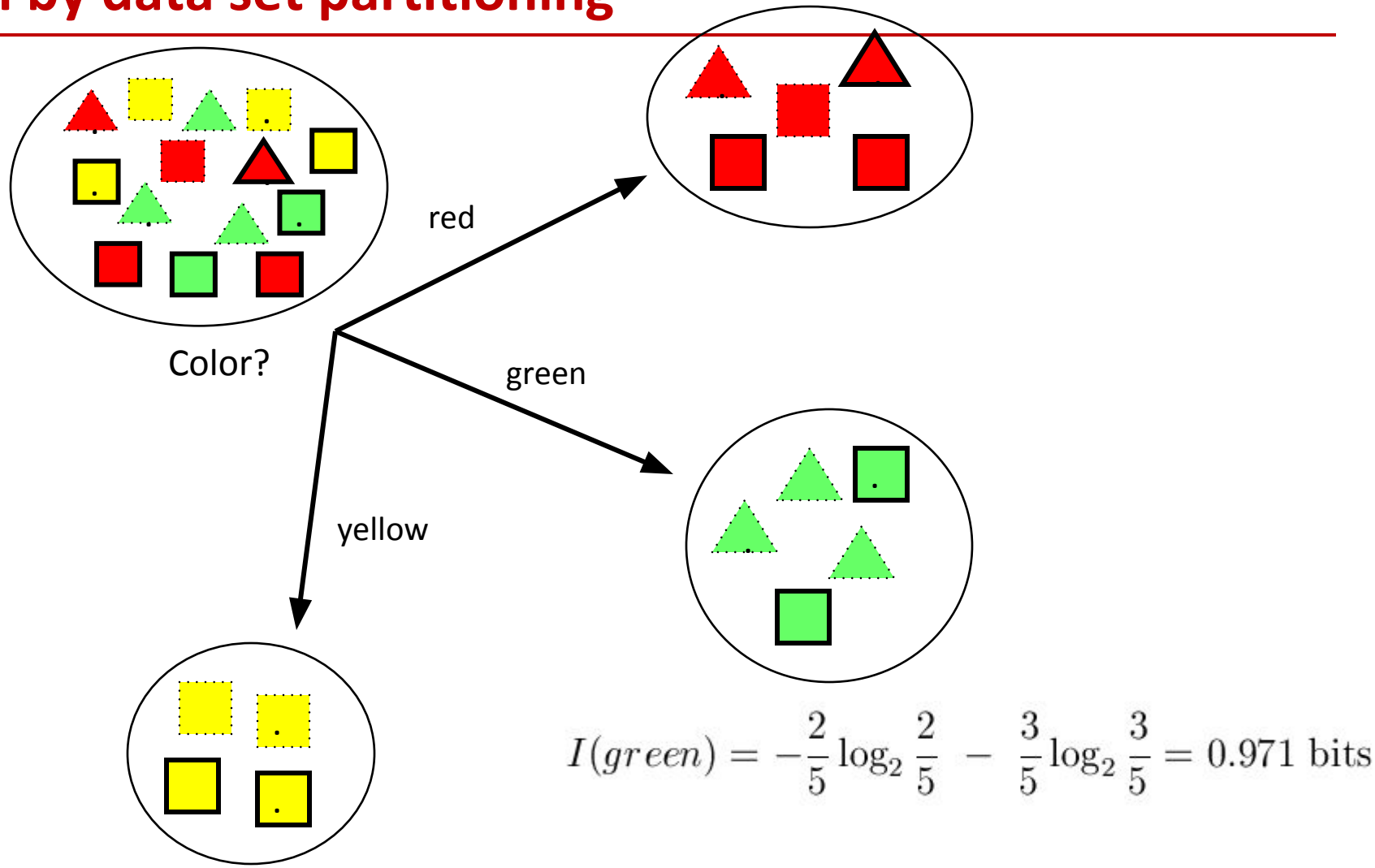
$$p(\triangle) = \frac{5}{14}$$

- 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

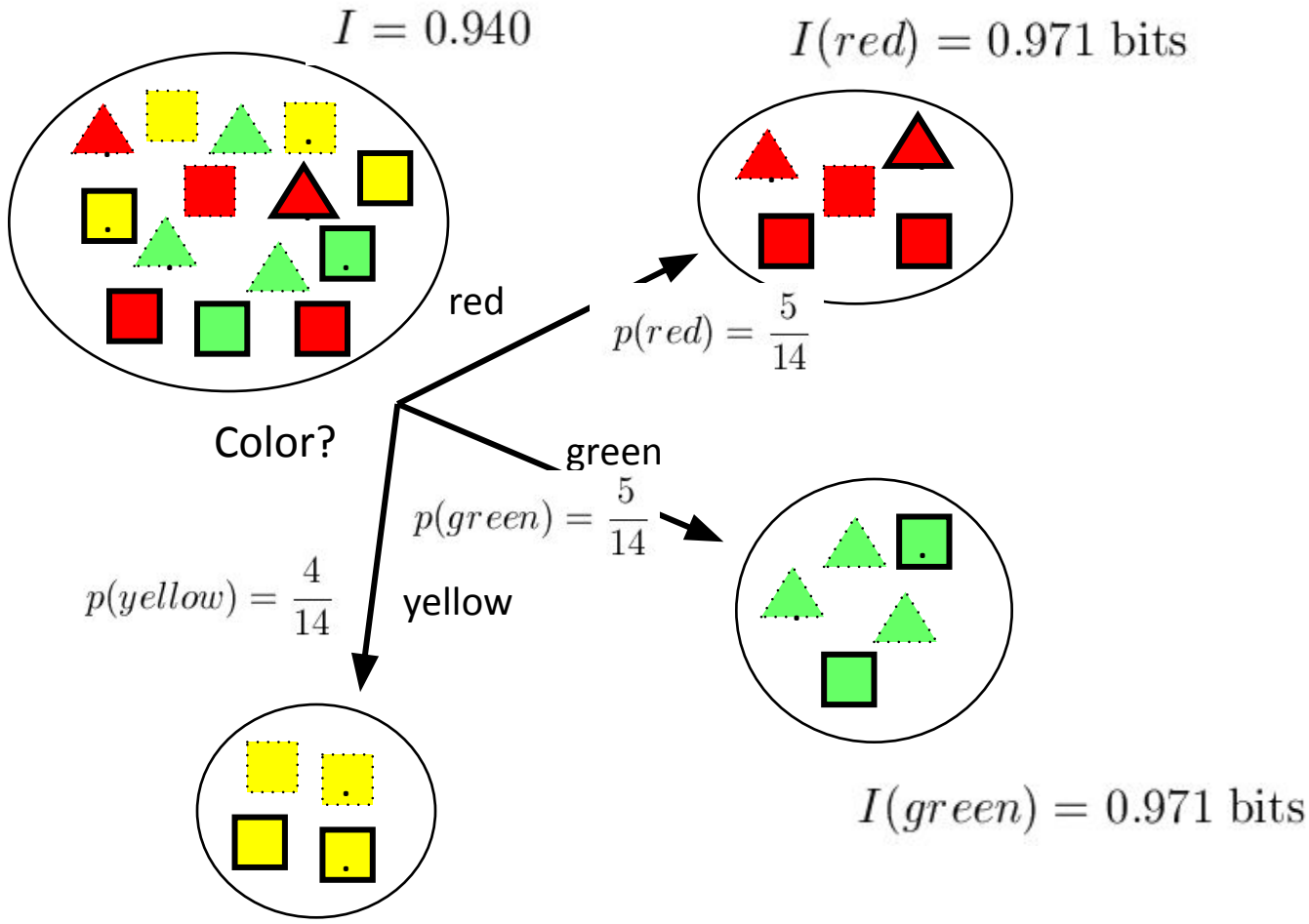
$$I(red) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.971 \text{ bits}$$



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

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

Entropy reduction by data set partitioning

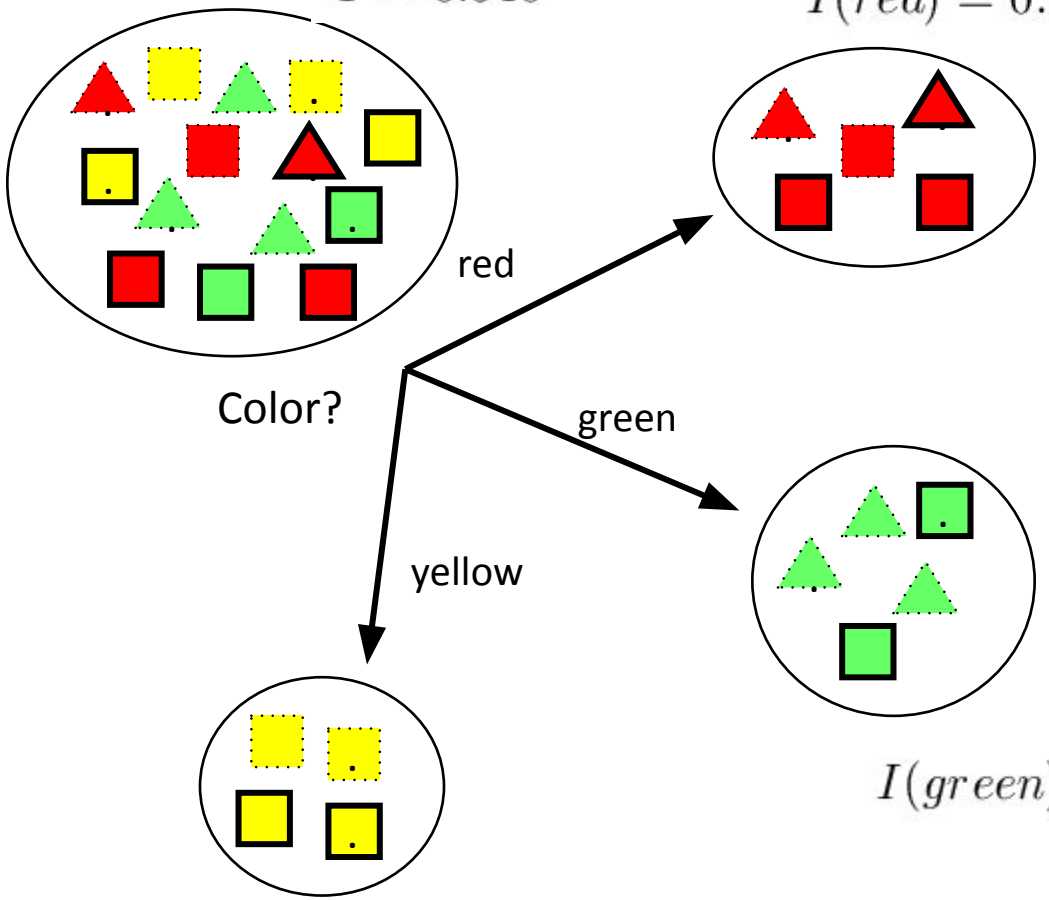


$$I_{res}(\text{Color}) = \sum p(v)I(v) = \frac{5}{14}0.971 + \frac{5}{14}0.971 + \frac{4}{14}0.0 = 0.694 \; bits$$

Information Gain

$I = 0.940$

$I(red) = 0.971$ bits



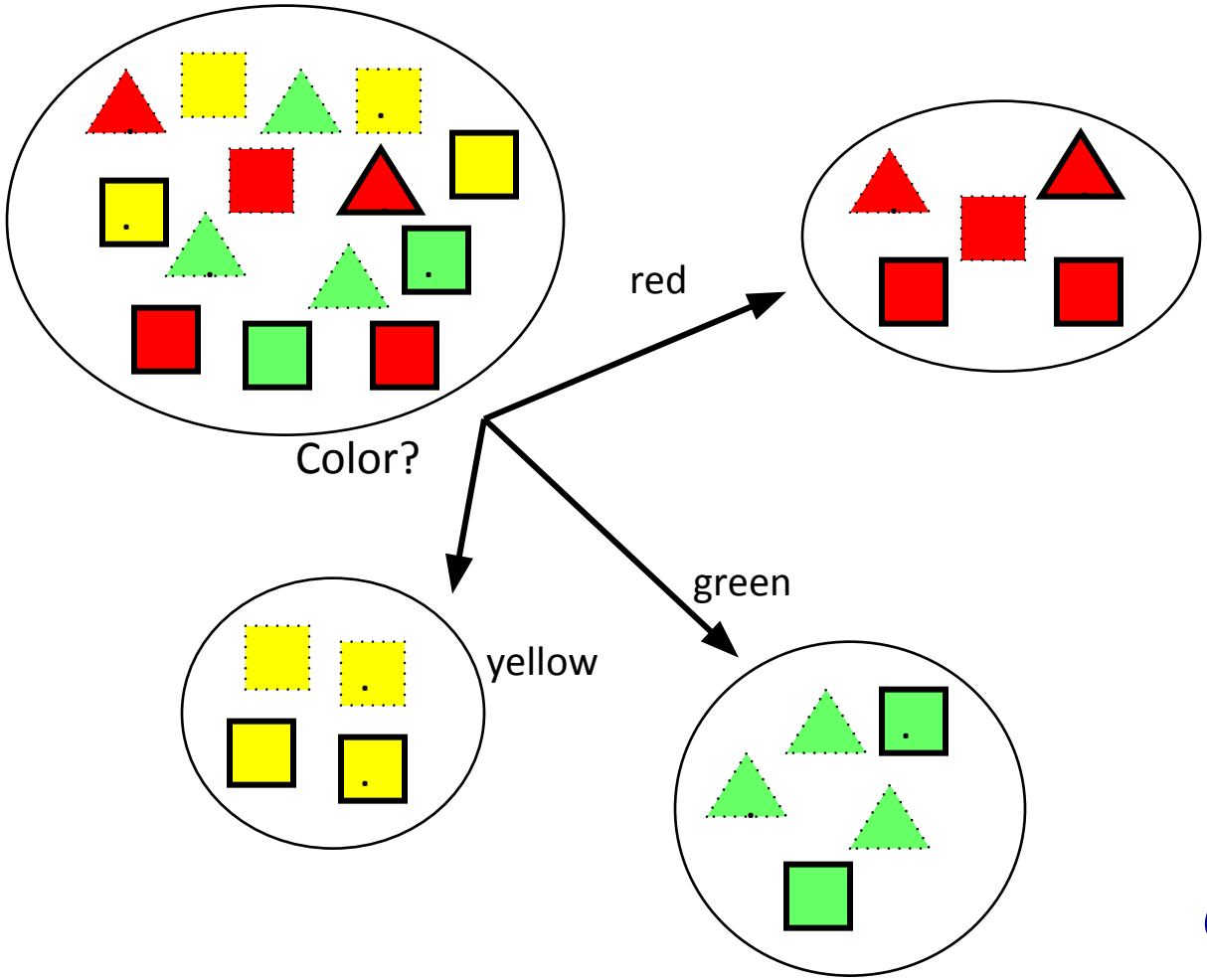
$I(green) = 0.971$ bits

$I(yellow) = 0.0$ bits

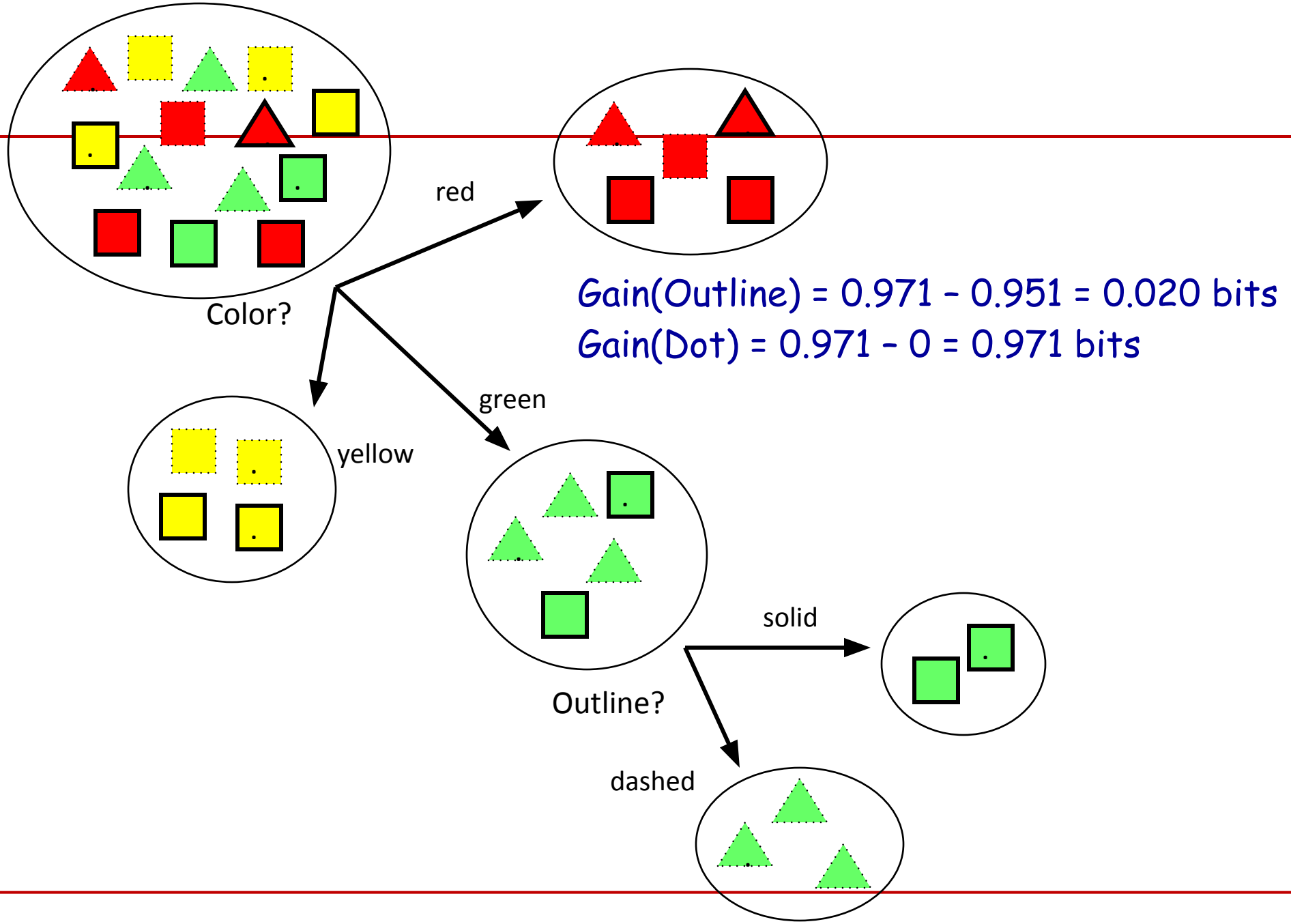
$$Gain(\text{Color}) = I - I_{res}(\text{Color}) = 0.940 - 0.694 = 0.246\ bits$$

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)

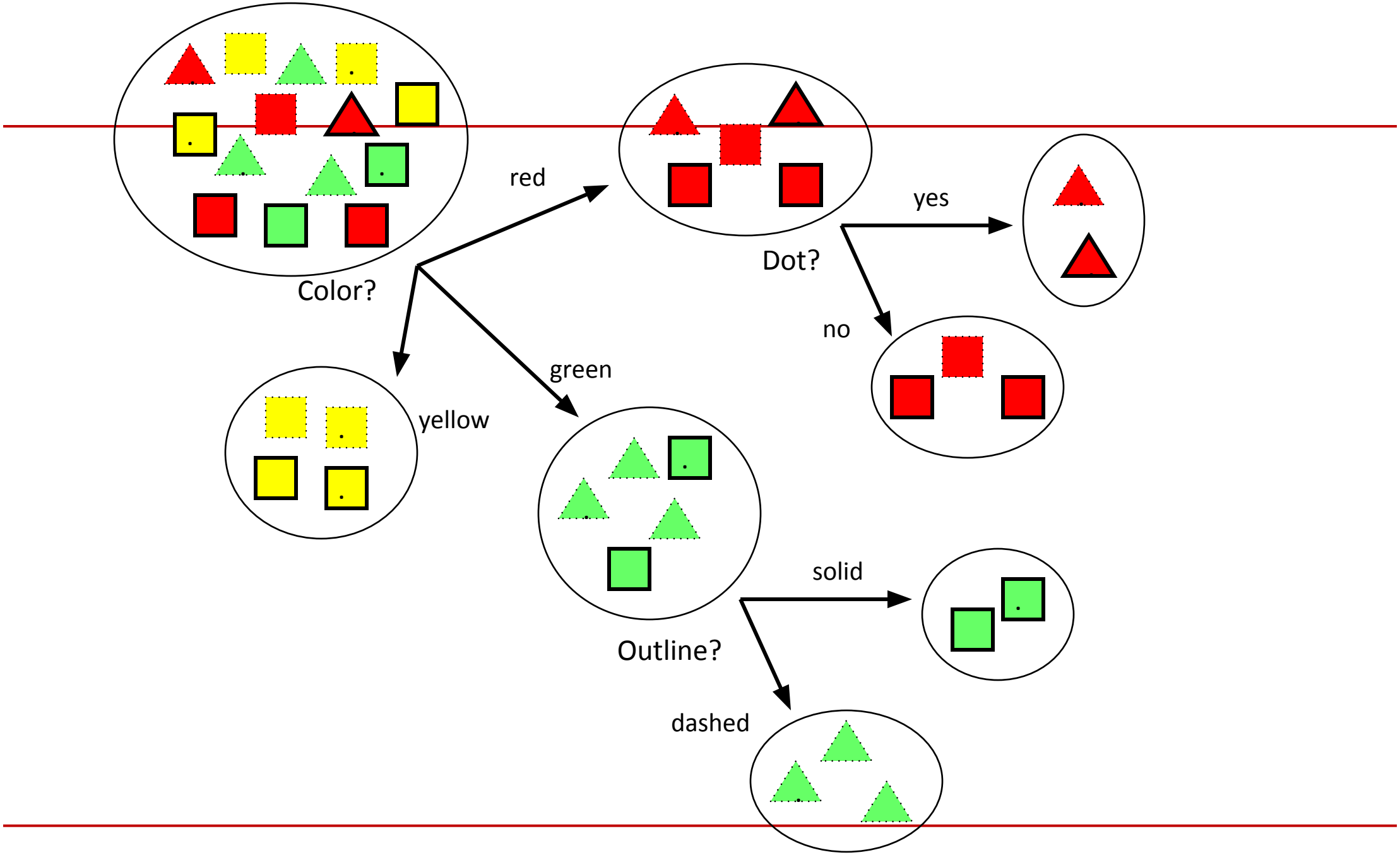


Gain(Outline) = 0.971 – 0 = 0.971 bits
Gain(Dot) = 0.971 – 0.951 = 0.020 bits

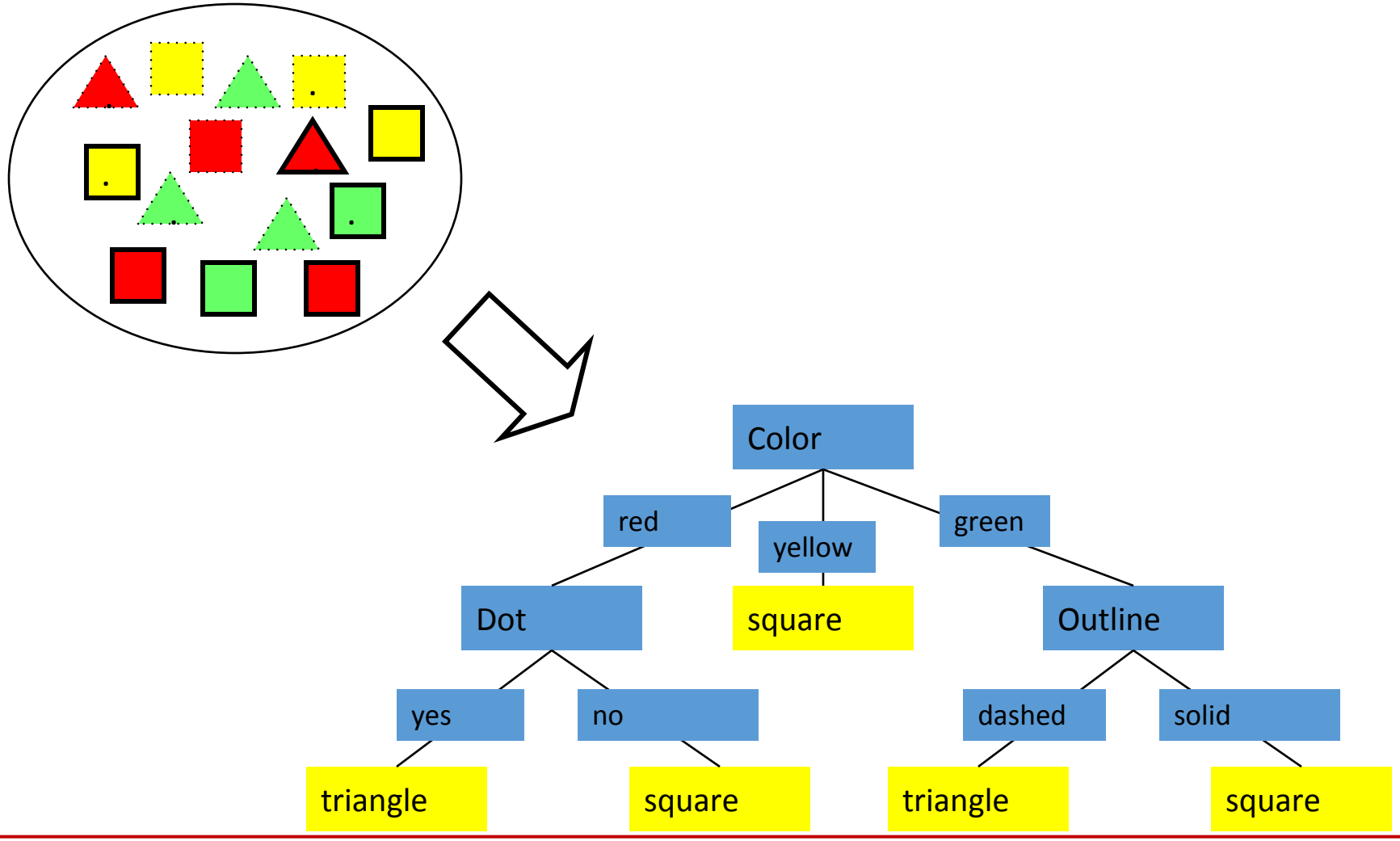


Gain(Outline) = 0.971 – 0.951 = 0.020 bits

Gain(Dot) = 0.971 – 0 = 0.971 bits


Color?
red
yellow
green
Dot?
yes
no
Outline?
solid
dashed

Decision Tree



Definition

- Decision tree is a classifier in the form of a tree structure
  - Decision node: specifies a test on a single attribute
  - Leaf node: indicates the value of the target attribute
  - Arc/edge: split of one attribute
- Decision trees classify instances or examples by starting at the root of the tree and moving through it until a leaf node.

key requirements

- **Attribute-value description:** object or case must be expressible in terms of a fixed collection of properties or attributes (e.g., hot, mild, cold).
- **Predefined classes (target values):** the target function has discrete output values (Boolean or Multiclass)
- **Sufficient data:** enough training cases should be provided to learn the model.

Splitting Based on Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the best cut
    - can be more compute intensive

Binary Decision

- For continuous attribute
  - Partition the continuous value of attribute A into a discrete set of intervals
  - Create a new Boolean attribute A_c , looking for a threshold c,

$$A_c = \begin{cases} true & \text{if } A_c < c \\ false & \text{otherwise} \end{cases}$$

How to choose c ?

Continuous Attributes

- For efficient computation: for each attribute,
  - Sort the attribute on values (static or dynamic)
  - Linearly scan these values
  - Choose the split position which gives you the in the purity

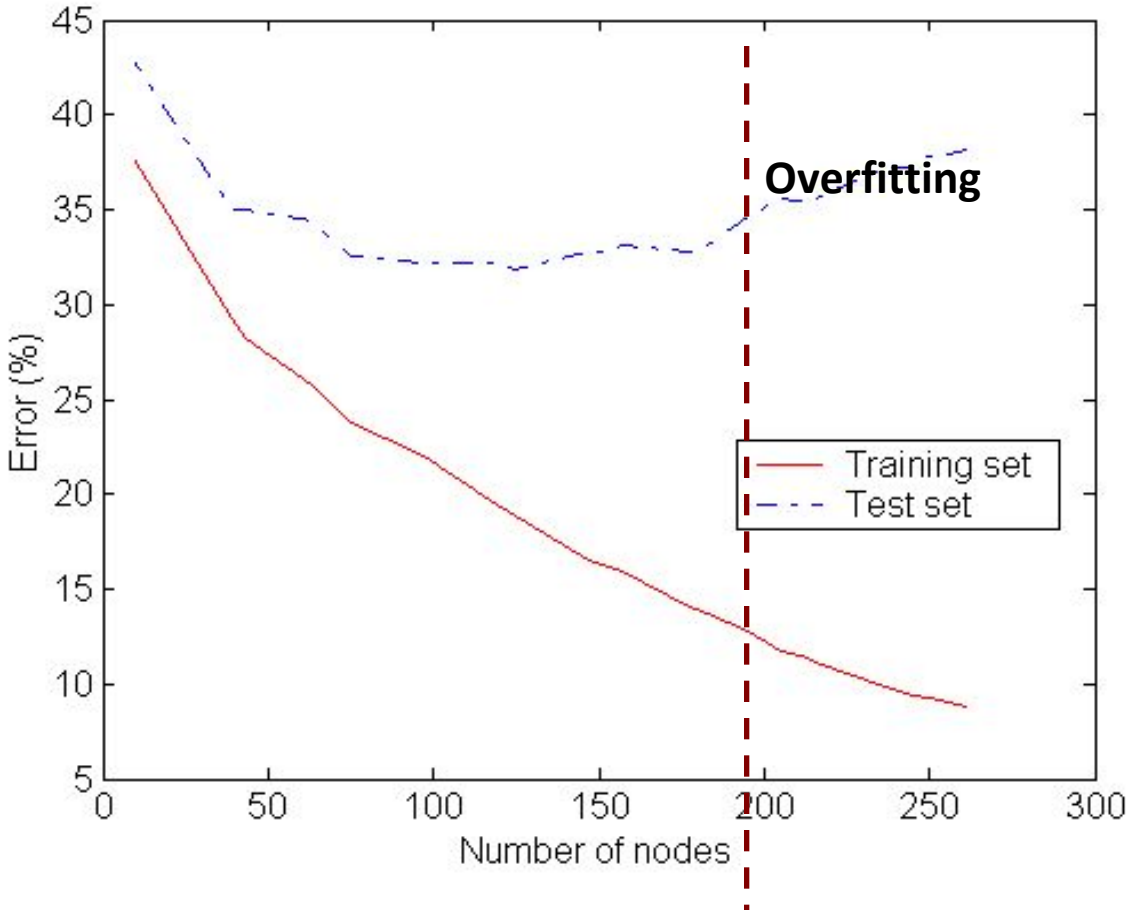
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Cheat	No		No		No		Yes		Yes		Yes		No		No		No		No		
Sorted Values →	Taxable Income																				
	60		70		75		85		90		95		100		120		125		220		
Split Positions →	55		65		72		80		87		92		97		110		122		172		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
Purity	?		?		?		?		?		?		?		?		?		?		?	

Practical Issues of Classification

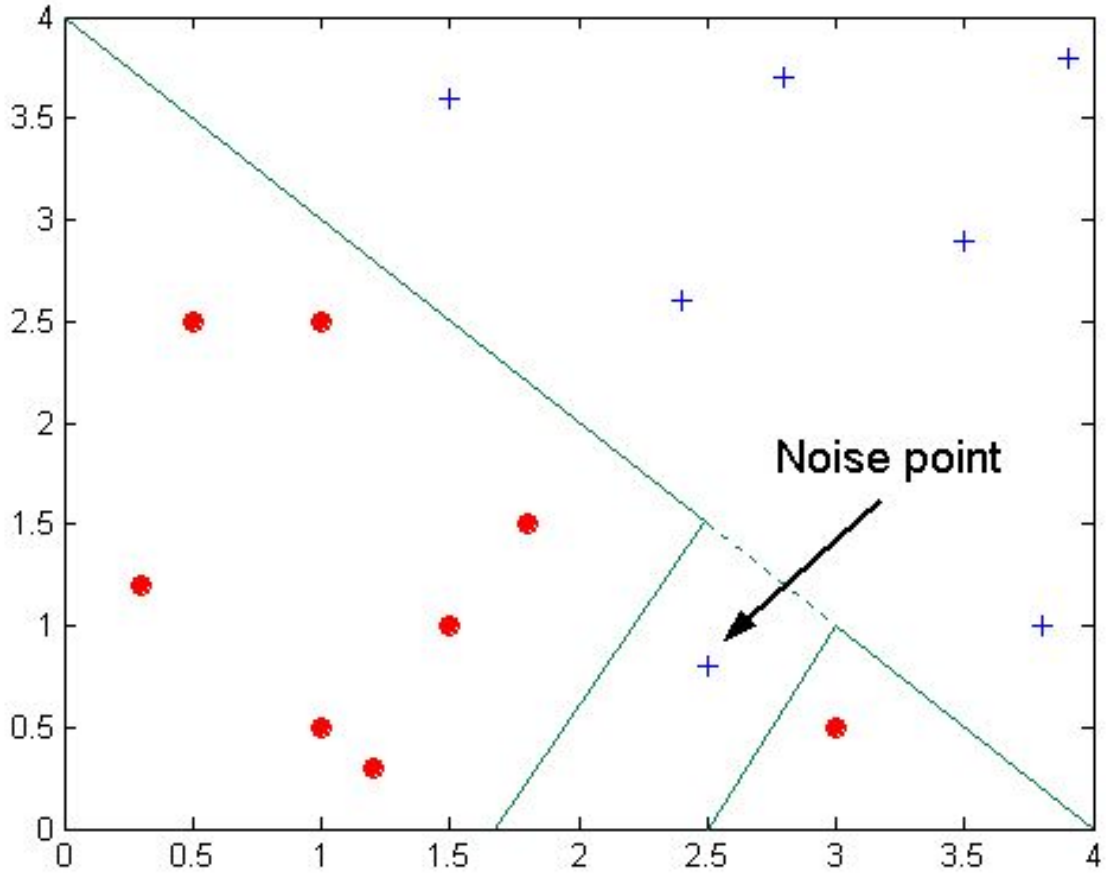
- Underfitting and Overfitting
- Missing Values

Underfitting and Overfitting



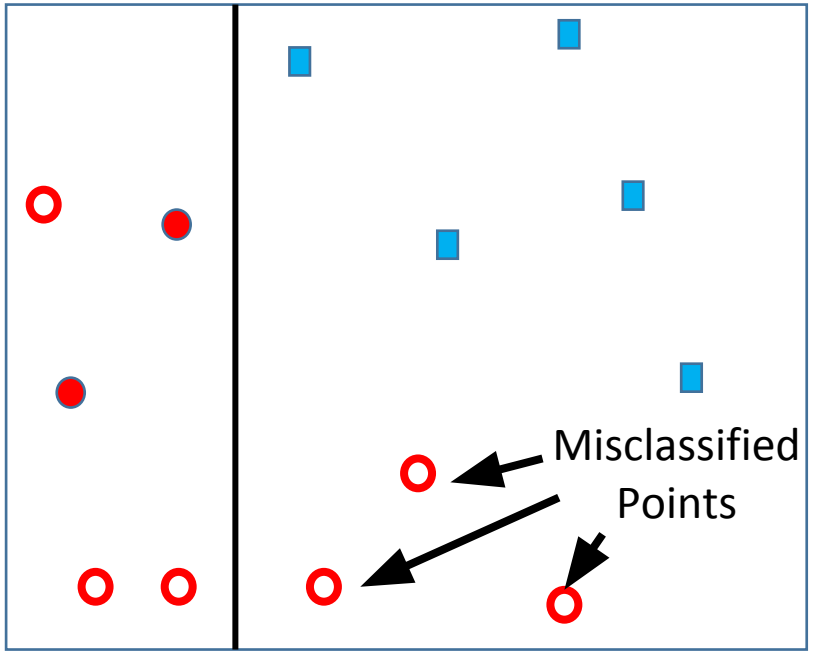
Underfitting: when model is too simple, both training and test errors are large

Overfitting due to Noise



Decision boundary is distorted by noise point

Overfitting due to Insufficient Examples



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records

Occam's Razor

- Given two models of similar generalization errors, one should prefer the simpler model over the more complex model
- For complex models, there is a greater chance that it was fitted accidentally by errors in data
- Therefore, one should include model complexity when evaluating a model

How to Address Overfitting

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical 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:
    - Stop if number of instances is less than some user-specified threshold
    - Stop if expanding the current node does not improve purity measures.

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 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

Handling Missing Attribute Values

- Assume an attribute can take the value “blank”.
- Assign most common value of A among training data at node n.
- Assign most common value of A among training data at node n which have the same target class.

Strengths

- Can generate understandable rules
- Perform classification without much computation
- Can handle continuous and categorical variables
- Provide a clear indication of which fields are most important for prediction or classification

Weakness

- Not suitable for prediction of continuous attribute.
- Perform poorly with many class and small data.
- Computationally expensive to train.
  - At each node, each candidate splitting field must be sorted before its best split can be found.
  - In some algorithms, combinations of fields are used and a search must be made for optimal combining weights.
  - Pruning algorithms can also be expensive since many candidate sub-trees must be formed and compared.

Measure of impurity: Gini index

- **Gini index for a given node *t* :**

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

p(j | t) **is the relative frequency of class *j* at node *t***

- **Maximum (1 – 1 / n_c) when records are equally distributed among all classes, implying least amount of information (n_c = number of classes).**
- **Minimum (0.0) when all records belong to one class, implying most amount of information.**

C1	0
C2	6
Gini=0.000	

C1	1
C2	5
Gini=0.278	

C1	2
C2	4
Gini=0.444	

C1	3
C2	3
Gini=0.500	

Wish list for a purity measure

- **Properties we require from a impurity measure:**
  - **When node is pure, measure should be zero**
  - **When impurity is maximal (i.e. all classes equally likely), measure should be maximal**
  - **Measure should obey *multistage property* (i.e. decisions can be made in several stages):**

$$\text{measure}([2,3,4]) = \text{measure}([2,7]) + (7/9) \times \text{measure}([3,4])$$

- **Entropy is a function that satisfies all three properties!**

Issue with Information Gain

- There is a natural bias in the information gain measure that favors attributes with many values over those with few values.
- As an extreme example, consider the attribute Date, which has a very large number of possible values, it would have the highest information gain of any of the attributes.
- This is because Date alone perfectly predicts the target attribute over the training data.
- Thus, it would be selected as the decision attribute for the root node of the tree and lead to a (quite broad) tree of depth one, which perfectly classifies the training data.
- Of course, this decision tree would fare poorly on subsequent examples, because it is not a useful predictor despite the fact that it perfectly separates the training data.

The gain ratio

- *Gain ratio*: a modification of the information gain that reduces its bias on high-branch attributes
- Gain ratio takes number and size of branches into account when choosing an attribute
  - It corrects the information gain by taking the *intrinsic information* of a split into account. Also called split ratio
- Intrinsic information: entropy of distribution of instances into branches
  - (i.e. how much info do we need to tell which branch an instance belongs to)

Gain Ratio

- ***Gain ratio*** **normalizes info gain by this reduction:**

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

where S1 through S, are the k subsets of examples resulting from partitioning S by the k-valued attribute A.

- **Note that SplitInfomzation is actually the entropy of S with respect to the values of attribute A.**

Gain Ratio

- **The Gain Ratio measure is defined in terms of the earlier Gain measure, as well as this SplitInfomzation, as follows**

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

- **Notice that the SplitInfomzation term discourages the selection of attributes with many uniformly distributed values.**
- **For example, consider a collection of n examples that are completely separated by attribute A (e.g., Day). In this case, the SplitInfomzation value will be $\log_2 n$. In contrast, a boolean attribute B that splits the same n examples exactly in half will have SplitInfomzation of 1. If attributes A and B produce the same information gain, then clearly B will score higher according to the Gain Ratio measure.**

CART: Classification and Regression Tree

- Perform binary split at each node
- For Classification tree Gini index is used as an impurity measure
- For regression trees CART uses the following criteria function:

$$\textbf{Equal variances}(CART)\textbf{: h} = \frac{N_L\hat{\sigma}_L^2 + N_R\hat{\sigma}_R^2}{N_L + N_R}$$

CART

1. Assign all objects to root node.
2. Split each attribute at all its possible split points (that is in between all the values observed for that variable in the considered node).
3. For each split point, split the parent node into two child nodes by separating the objects with values lower and higher than the split point for the considered explanatory variable.
4. Select the variable and split point with the highest reduction of impurity.
5. Perform the split of the parent node into the two child nodes according to the selected split point.
6. Repeat steps 2–5, using each node as a new parent node, until the tree has maximum size.
7. Prune the tree back using cross-validation to select the optimal sized tree

Acknowledgement

- Classifiers, Prof. Pushpak Bhattacharyya and Aditya M Joshi
- Nearest-Neighbor Classifier MTL 782 IIT DELHI
- CS 4700: Foundations of Artificial Intelligence, Carla P. Gomes