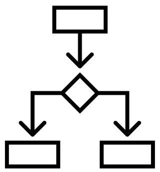
Decision Tree Classifier



This Session

- Overview
- General algorithm
- Decision Tree use cases
  - Entropy
  - Information gain

Decision Tree Classifier

- A decision tree (aka **prediction tree**) uses a **tree structure** to **specify sequences** of **decisions** and **consequences**

- The **prediction** can be achieved by through **test points** and **branches**

    - **Test:** Each (test) point in a decision tree **involves testing** a particular **input variable** (or **attribute**)

    - **Consequence**: a decision is made to **pick a specific branch** and traverse down the tree

    - **Branch** represents the **decision** being **made**

    - **Prediction:** Eventually, a final point is reached, and a prediction can be made

- Due to its flexibility and easy visualization, decision trees are commonly deployed in data mining applications.

Decision Tree Classifier – Analogy

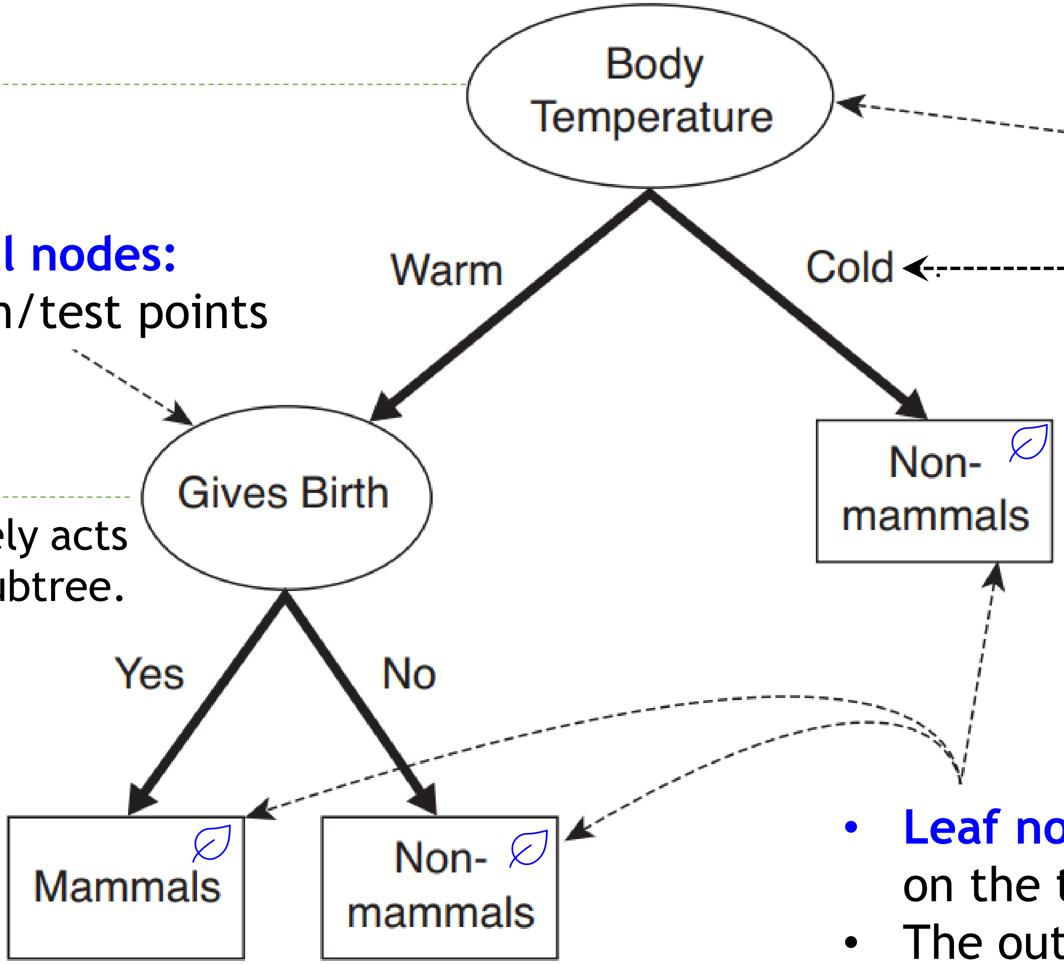
Root node - Top Internal node

Branch - outcome of a decision/test

Depth = 1

Internal nodes:
decision/test points

Each internal node effectively acts as the root of a subtree.



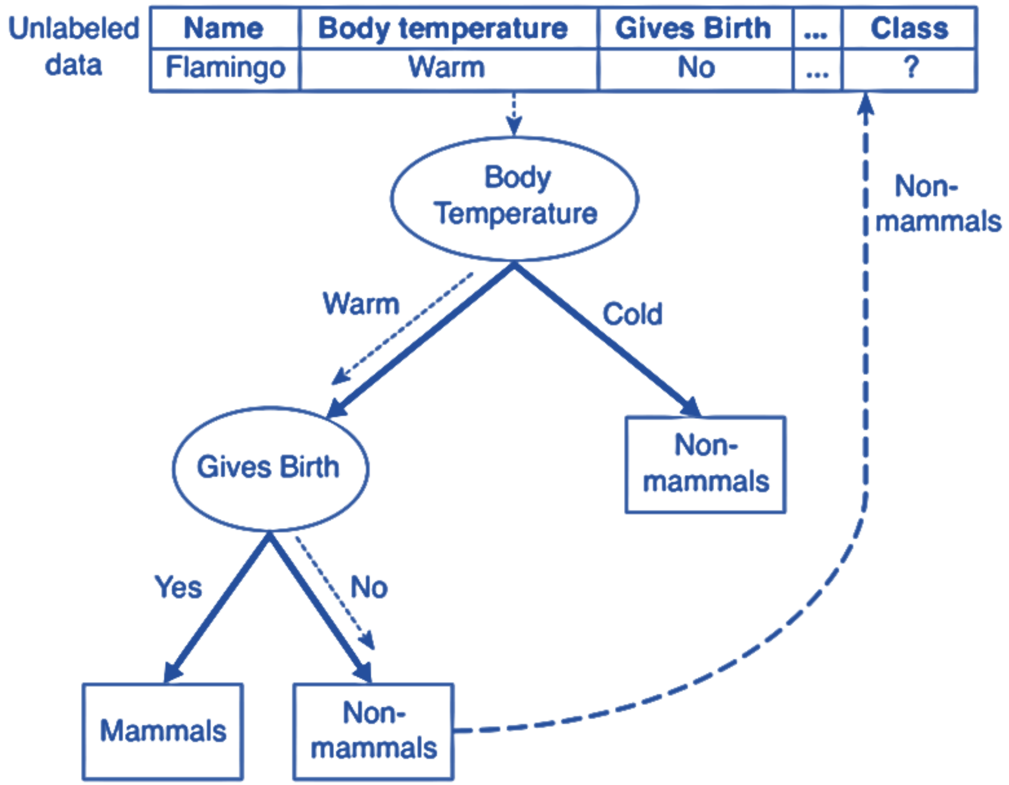
- **Leaf nodes** are at the end of the last branches on the tree.
- The outcome of all the prior decisions representing class labels.

- The **path from the root to a leaf** node contains **a series of decisions made at various internal nodes**.

Decision Tree Classifier – Analogy Cont.

- **DT Classifier for Species Classification:**

```
if (BT == cold):
   Species = non-mammal
else:
   if (gives birth):
         Species = non-mammal
   else:
          Species = mammal
```



- Another example is a **checklist of symptoms** during a doctor's evaluation of a patient.

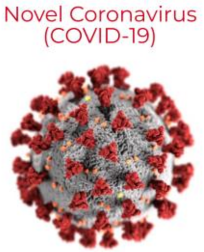


Image: https://www.scottcountyiowa.gov/

Decision Tree – The General Algorithm

- **Given:** **training set, S**

```
if (all samples in S belong to a specific class c_i ∈ C, or S is sufficiently pure)
 make a leaf labeled c_i
else{
  - select the most informative attribute A
  - partition S according to A's values
  - recursively construct sub-trees T_1, T_2, …, T_m for the subsets
    of S until one of the following conditions is met:
    Case 1: All the leaf nodes in the tree satisfy the minimum purity threshold

    Case 2: The tree cannot be further split

    Case 3: Any other Early Stopping criterion is satisfied (max depth of the tree)
}
```

Decision Tree – The General Algorithm

- It can be summarized in three important steps:

Step 1: Choose the most informative attribute with lowest Entropy

Step 2: Find the partition with the highest InfoGain

Step 3: At each resulting node, repeat Steps 1 and 2

- until node is "pure enough“
- Pure nodes => no information gain by splitting on other attributes

Decision Tree – The General Algorithm

Step 1: Choose the most informative attribute

- A **common way** to identify the most **informative attribute** is to use **entropy-based methods**.
- The **entropy methods select** the most informative attribute based on **two basic measures**:
  - **Entropy (E)** - measures the **impurity of an attribute**
  - **Information gain (IG)** - measures the **purity of an attribute**

- Family of Decision Tree algorithms:
  - **ID3** (Iterative Dichotomiser 3), **C4.5**, **C5.0**, **CART** (Classification and Regression Tree)
  - Different algorithms use different measures build the DT.

Decision Tree – The General Algorithm

- **Entropy:** if a class C and its label $c \in C$, let $p(c)$ be the probability of c. Then H_C the entropy of c, is defined as:

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

- **Example:** Consider a dataset with 1 blue, 2 greens, and 3 reds: ●●●●●●. Then:

$$H_C = -\sum_{c}^{●●●} p(c)\log_2 p(c)$$

$$= -(p_b \log_2 p_b + p_g \log_2 p_g + p_r \log_2 p_r)$$

From training data, $p_b = 1/6$, $p_g = 2/6$, and $p_r = 3/6$.

Thus,

$$H_C = -(\frac{1}{6}\log_2(\frac{1}{6}) + \frac{2}{6}\log_2(\frac{2}{6}) + \frac{3}{6}\log_2(\frac{3}{6}))$$
$$= \boxed{1.46}$$

Decision Tree – The General Algorithm

- **Entropy:** a class C and its label $c \in C$, let $p(c)$ be the probability of c. H_C the entropy of c, is defined as:

$$H_C = -\sum_{c \in \boldsymbol{C}} p(c) \log_2 p(c)$$

- **Question:** If a subset has all single class label? Example, consider 3 blues: ●●● What would be the entropy?

$$H_C = -(1 \log_2 1) = \boxed{0}$$

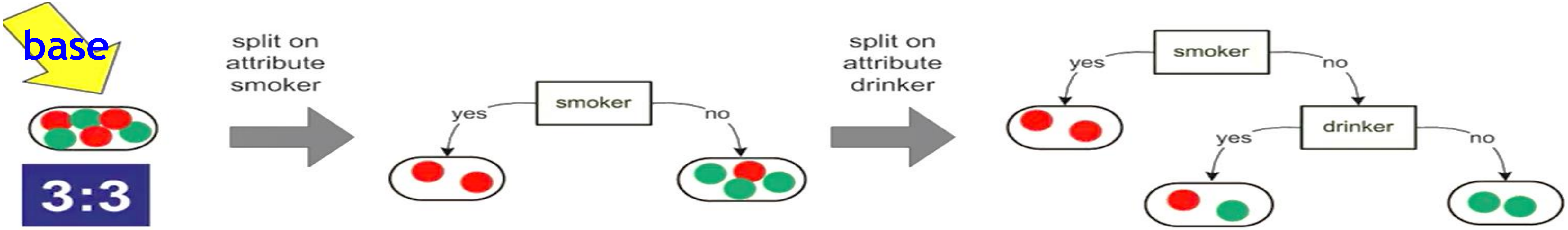
Decision Tree – The General Algorithm

- **Information gain (IG):**
  Information gain is defined as the difference between the base entropy and the conditional entropy of an attribute.

- Indicates the purity of an attribute - it compares the degree of purity of the parent node before a split with the degree of purity of the child node after a split.

- At each split, an attribute with the greatest IG is considered the most informative attribute.

$$\mathrm{InfoGain}_{attr} = H_{base} - H_{attr}$$

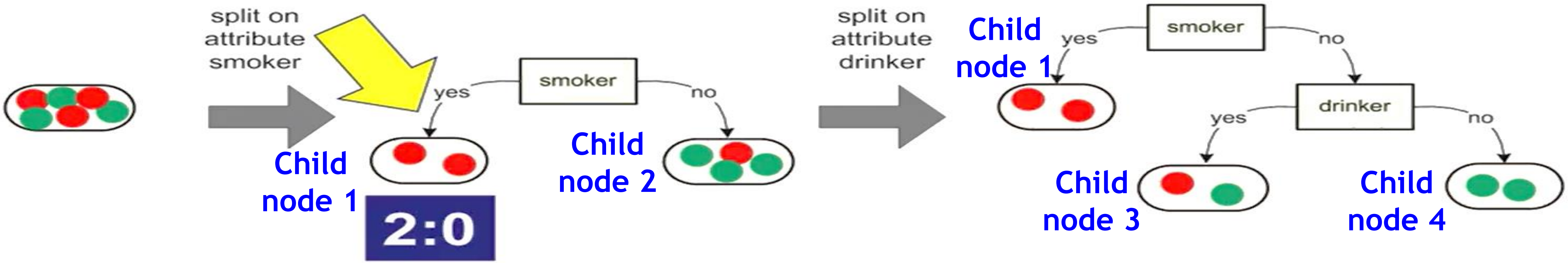


Base entropy:

$$E = -\sum_{i=1}^{k} p_i \log_2(p_i) = -(\underbrace{\frac{3}{6}\log_2(\frac{3}{6})}_{\text{First Class}} + \underbrace{\frac{3}{6}\log_2(\frac{3}{6})}_{\text{Second Class}}) = -(\frac{1}{2} \times -1 + \frac{1}{2} \times -1) = 1$$

Decision Tree – Example # 1 Cont.

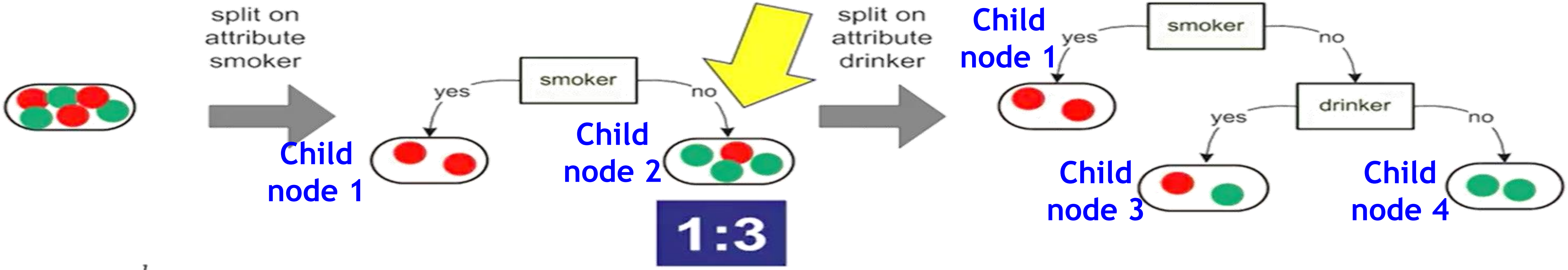
- **Conditional entropy of the attribute "smoker"**



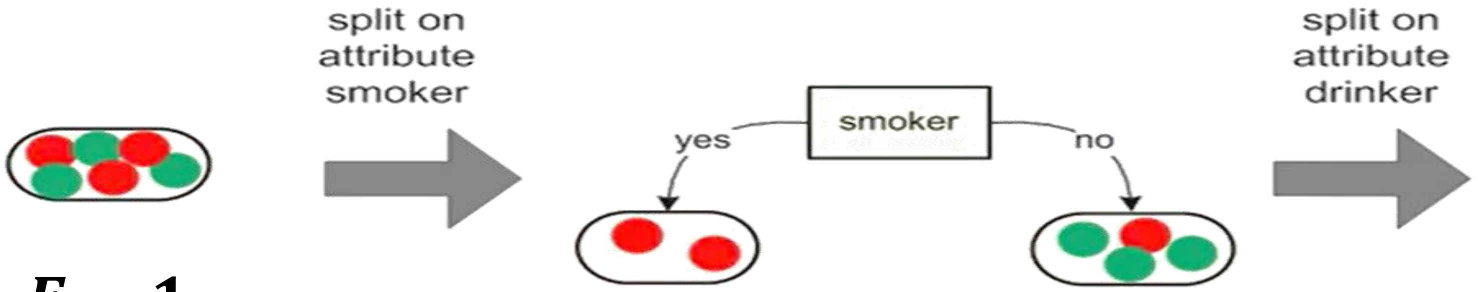
$$E = -\sum_{i=1}^{k} p_i \log_2(p_i) = -(\frac{2}{2}\log_2(\frac{2}{2})) = -(1 \times 0) = 0$$

Decision Tree – Example # 1 Cont.

- **Conditional entropy of the attribute "smoker" Cont.**



$$E = -\sum_{i=1}^{k} p_i \log_2(p_i) = -(\frac{1}{4}\log_2(\frac{1}{4}) + \frac{3}{4}\log_2(\frac{3}{4})) = -(\frac{1}{4}\times -2 + \frac{3}{4}\times -0.415) = 0.811$$



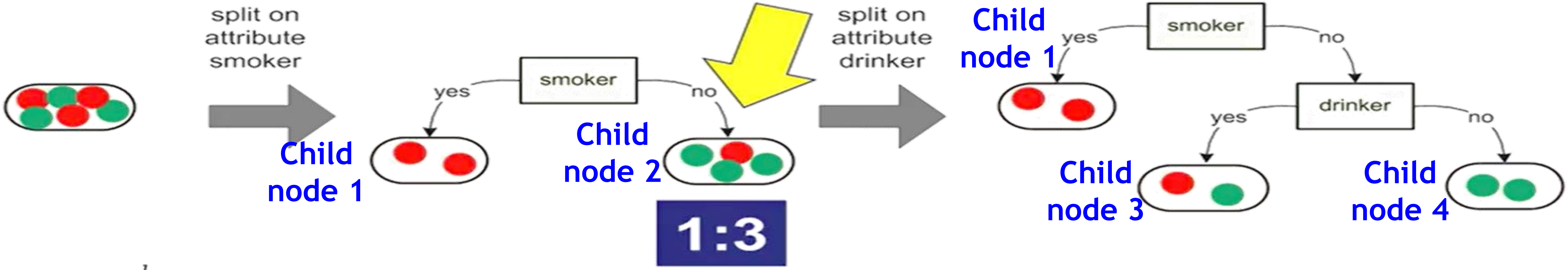
$\boldsymbol{E = 1}$

$\boldsymbol{E = 0}$ $\boldsymbol{E = 0.811}$

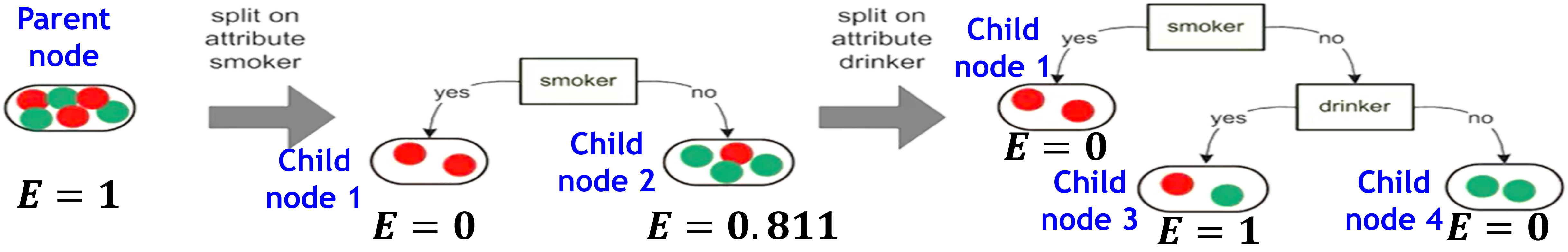
- **Question:** Compute the conditional entropy of child node 3 child node 4.

Decision Tree – Example # 1 Cont.

- Conditional entropy of the attribute “smoker” Cont.



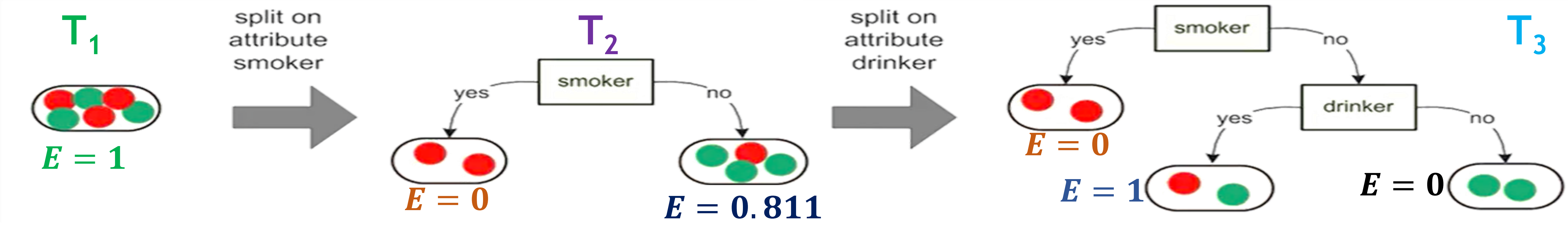
$$E = -\sum_{i=1}^{k} p_i \log_2(p_i) = -(\frac{1}{4}\log_2(\frac{1}{4})+\frac{3}{4}\log_2(\frac{3}{4})) = -(\frac{1}{4}\times -2+\frac{3}{4}\times -0.415) = 0.811$$



Decision Tree – Example # 1 Cont.

- **Weighted average of the Entropy (E_w)**
  - The entropy for each child (c_i) is **weighted by the proportion** of instances belonging to that child, $p(c_i)$.

$$[\, p(c_1) \times entropy(c_1) + p(c_2) \times entropy(c_2) + \cdots \,]$$



T_1 Ew = $\frac{6}{6} \times 1 = 1$ T_2 Ew = $\frac{2}{6} \times 0 + \frac{4}{6} \times 0.811 = 0.54$ T_3 Ew = $\frac{2}{6} \times 1 + \frac{2}{6} \times 0 = 0.33$