Learning Decision Trees

COURSE: CS40002

Pallab Dasgupta
Professor,
Dept. of Computer Sc & Engg

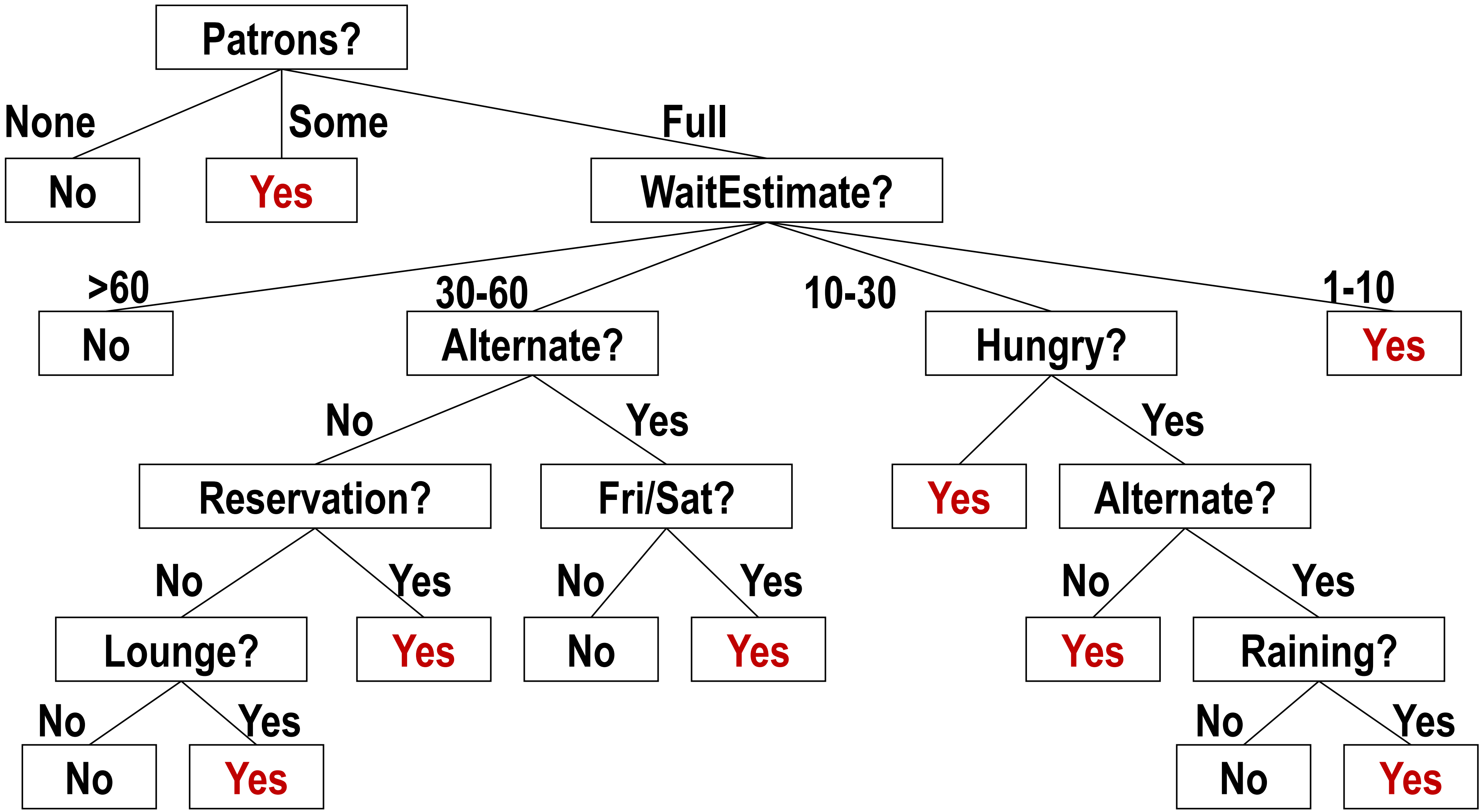
INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Decision Trees

- A decision tree takes as input an object or situation described by a set of properties, and outputs a yes/no “decision”.
- A list of variables which potentially affect the decision on *whether to wait for a table at a restaurant.*

1. *Alternate*: whether there is a suitable alternative restaurant
2. *Lounge*: whether the restaurant has a lounge for waiting customers
3. *Fri/Sat*: true on Fridays and Saturdays
4. *Hungry*: whether we are hungry
5. *Patrons*: how many people are in it (None, Some, Full)
6. *Price*: the restaurant’s rating (★, ★★, ★★★)
7. *Raining*: whether it is raining outside
8. *Reservation*: whether we made a reservation
9. *Type*: the kind of restaurant (Indian, Chinese, Thai, Fastfood)
10. *WaitEstimate*: 0-10 mins, 10-30, 30-60, >60.

Sample Decision Tree



Decision Tree Learning

Aim: find a <u>small</u> tree consistent with the training examples

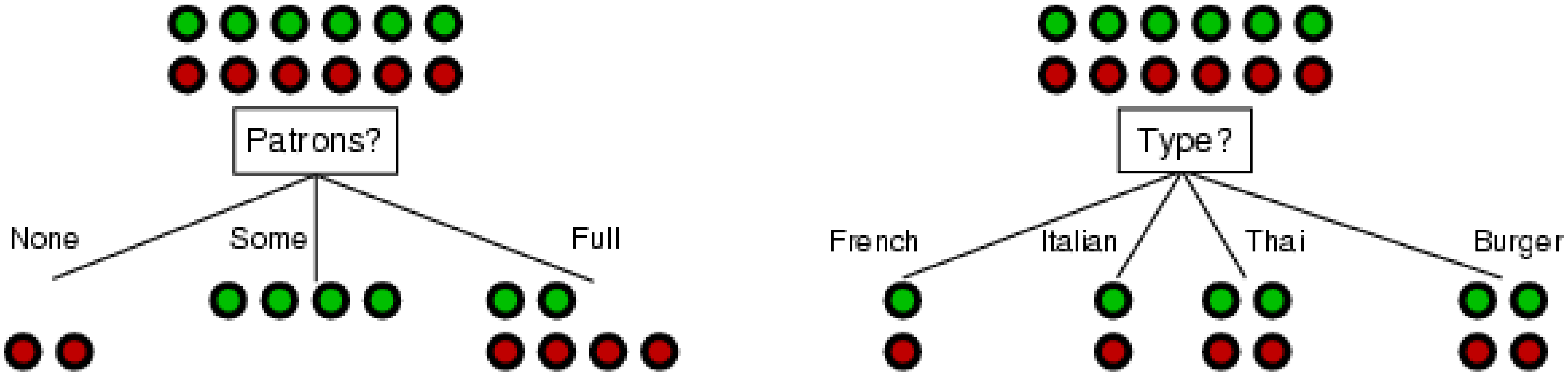
Idea: (recursively) choose "most significant" attribute as root of (sub) tree

```
function DTL(examples, attributes, default) returns a decision tree

    if examples is empty then return default
    else if all examples have the same classification then return the classification
    else if attributes is empty then return MODE(examples)
    else
        best ← CHOOSE-ATTRIBUTE(attributes, examples)
        tree ← a new decision tree with root test best
        for each value v_i of best do
            examples_i ← {elements of examples with best = v_i}
            subtree ← DTL(examples_i, attributes − best, MODE(examples))
            add a branch to tree with label v_i and subtree subtree
        return tree
```

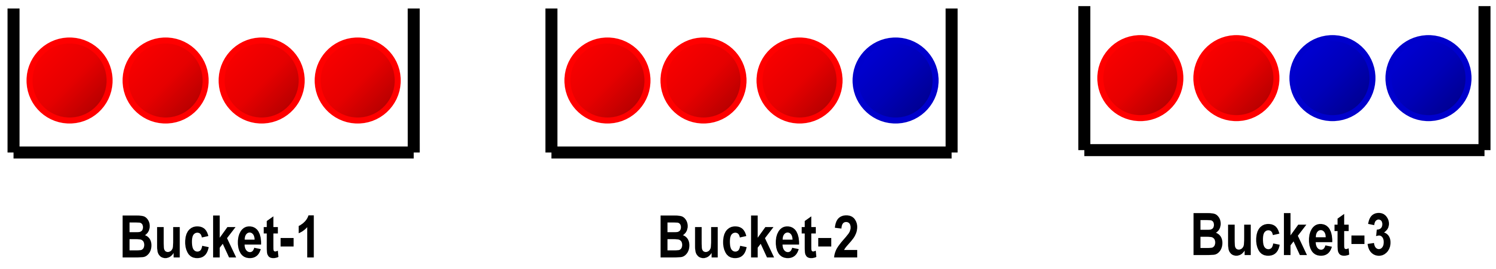
Choosing an attribute

Idea: A good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"



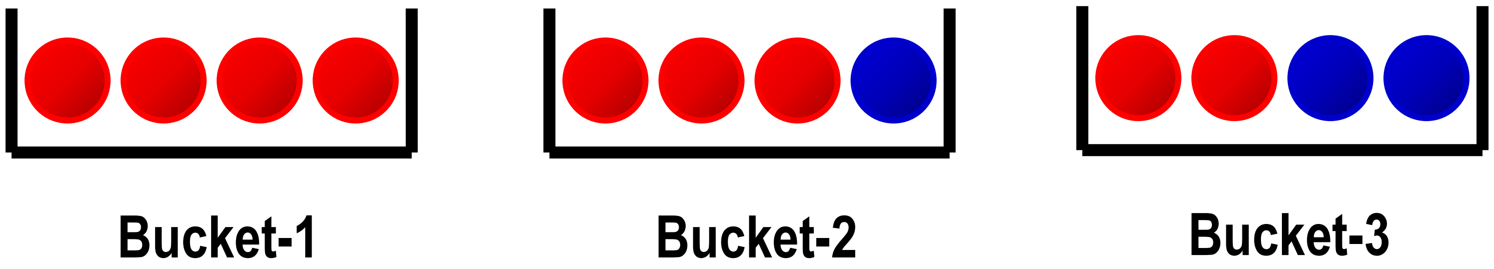
Patrons? **is a better choice**

Entropy and Knowledge



- How much information do we have on the color of a ball drawn at random?
  - In the first bucket we are sure that the ball will be red
  - In the second bucket we know with 75% certainty that the ball will be red
  - In the third bucket we know with 50% certainty that the ball will be red
- Bucket-1 gives us the most amount of knowledge about the color of the ball
- Entropy is the opposite of knowledge
  - Bucket-1 has the least amount of entropy and Bucket-3 has the highest entropy

Entropy and Probability



- How many distinct arrangements of the balls are possible?
  - For the first bucket we have only one arrangement: RRRR
  - For the second bucket we have four arrangements: RRRB, RRBR, RBRR, BRRR
  - For the third bucket we have six arrangements: RRBB, RBBR, BBRR, RBRB, BRBR, BRRB

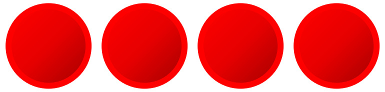
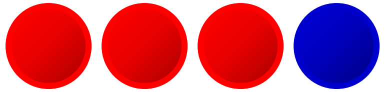
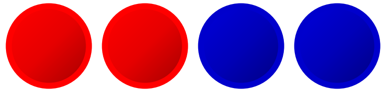
- The probability of finding a specific arrangement in four draws of balls is less for the third bucket because the number of possible arrangements is larger.

An interesting game for understanding entropy

We're given, again, the three buckets to choose. The rules go as follows:

- We choose one of the three buckets.
- We are shown the balls in the bucket, in some order. Then, the balls go back in the bucket.
- We then pick one ball out of the bucket, at a time, record the color, and return the ball back to the bucket.
- If the colors recorded make the same sequence than the sequence of balls that we were shown at the beginning, then we win. If not, then we lose.

Example

Pattern	P(red)	P(blue)	P(win)
	1	0	1 x 1 x 1 x 1 = 1
	0.75	0.25	0.75 x 0.75 x 0.75 x 0.25 = 0.105
	0.5	0.5	0.5 x 0.5 x 0.5 x 0.5 = 0.0625

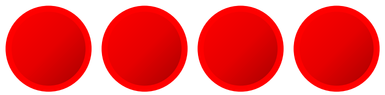
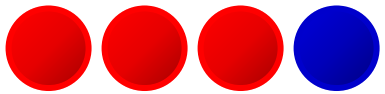
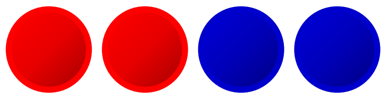
- **Products of many probability terms will make the metric very small and create precision problems**
- **Instead, we can take the logarithm of P(win), which will convert the product into a sum. Since probability terms are fractional, the logarithm will be negative and hence we take its negation**
- **For example, for Bucket-2 we compute:**

$$-\log_2(0.75) - \log_2(0.75) - \log_2(0.75) - \log_2(0.25) = 3.245$$

- **Finally we take the average in order to normalize:**

$$\frac{1}{4}(-\log_2(0.75) - \log_2(0.75) - \log_2(0.75) - \log_2(0.25)) = 0.81125$$

Example

Pattern	P(red)	P(blue)	P(win)
	1	0	1 x 1 x 1 x 1 = 1
	0.75	0.25	0.75 x 0.75 x 0.75 x 0.25 = 0.105
	0.5	0.5	0.5 x 0.5 x 0.5 x 0.5 = 0.0625

Entropy = $\frac{-m}{m+n}\log_2\left(\frac{m}{m+n}\right)+\frac{-n}{m+n}\log_2\left(\frac{n}{m+n}\right)$

- **Entropy for Bucket-3:** $\frac{-2}{2+2}\log_2\left(\frac{2}{2+2}\right)+\frac{-2}{2+2}\log_2\left(\frac{2}{2+2}\right)=\frac{1}{2}+\frac{1}{2}=1$
- **Entropy for Bucket-1:** $\frac{-4}{4+0}\log_2\left(\frac{4}{4+0}\right)+\frac{-0}{0+4}\log_2\left(\frac{0}{4+0}\right)=0+0=0$
- **Entropy for Bucket-2:** $\frac{-3}{3+1}\log_2\left(\frac{3}{3+1}\right)+\frac{-1}{1+3}\log_2\left(\frac{1}{1+3}\right)=0.81125$

Returning to the Decision Tree Learning Algorithm

To implement `Choose-Attribute` in the DTL algorithm

Information Content (Entropy):

$$I\big(P(v_1), \dots, P(v_n)\big) = \sum_{j=1}^{n} -P(v_j) \log_2 P(v_j)$$

For a training set containing *p* positive examples and *n* negative examples:

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

Information Gain

A chosen attribute *A* divides the training set *E* into subsets $E_1, \ldots, E_v$ according to their values for *A*, where *A* has v distinct values.

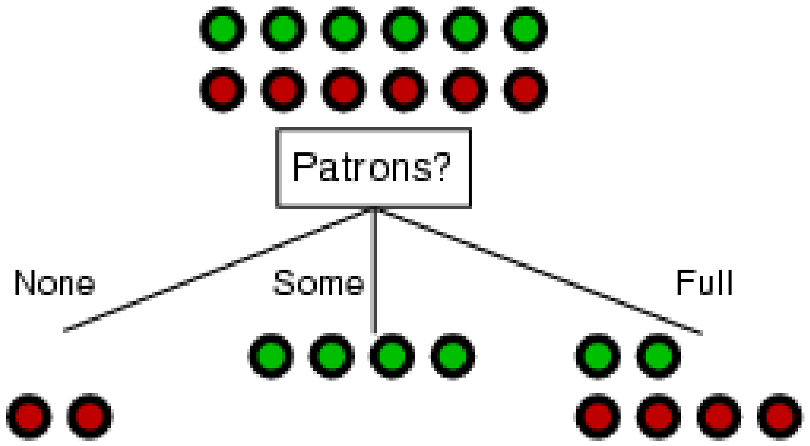
$$\text{remainder}(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

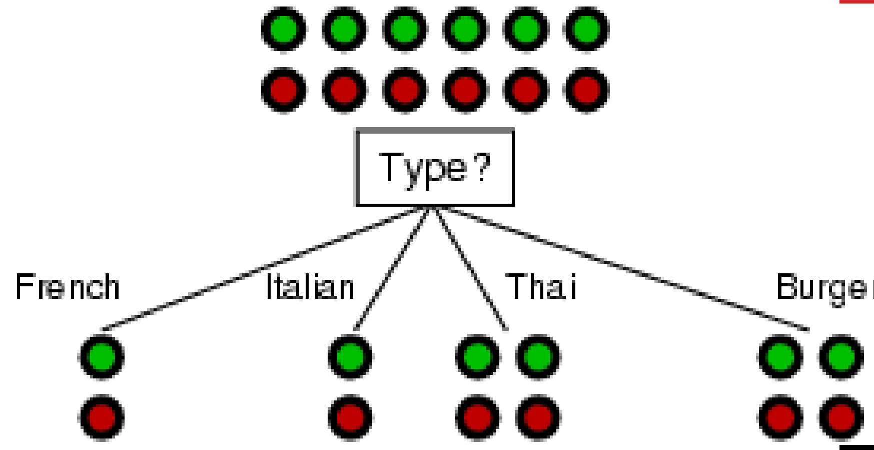
Information Gain (IG) or reduction in entropy from the attribute test:

$$IG(A) = I\left(\frac{p}{p + n}, \frac{n}{p + n}\right) - \text{remainder}(A)$$

Choose the attribute with the largest IG

Information gain





For the training set, $p = n = 6$, *I(6/12, 6/12)* = *1* bit

Consider the attributes *Patrons* and *Type* (and others too):

$$IG(Patrons) = 1 - [\frac{2}{12}I(0,1) + \frac{4}{12}I(1,0) + \frac{6}{12}I(\frac{2}{6},\frac{4}{6})] = .0541 \text{ bits}$$

$$IG(Type) = 1 - [\frac{2}{12}I(\frac{1}{2},\frac{1}{2}) + \frac{2}{12}I(\frac{1}{2},\frac{1}{2}) + \frac{4}{12}I(\frac{2}{4},\frac{2}{4}) + \frac{4}{12}I(\frac{2}{4},\frac{2}{4})] = 0 \text{ bits}$$

***Patrons* has the highest IG of all attributes and so is chosen by the DTL algorithm as the root**