Learning Decision Trees

Decision making based on information

COURSE: CS60045

Pallab Dasgupta
Professor,
Dept. of Computer Sc & Engg

Decision Trees

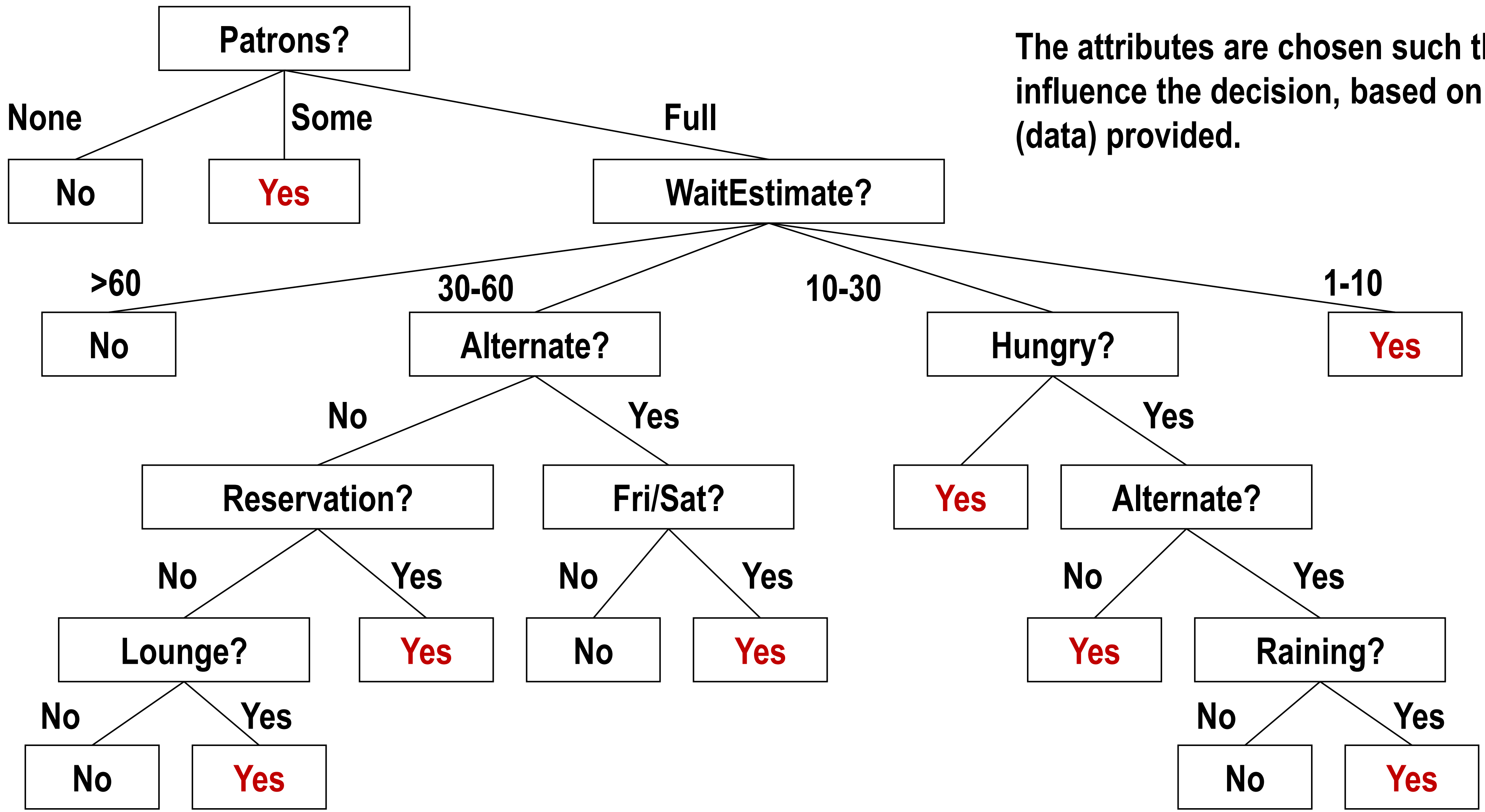
- A decision tree takes as input an object or situation described by a set of properties, and outputs a yes/no "decision".
- A list of variables which potentially affect the decision on *whether to wait for a table at a restaurant.*

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.

A portion of the given data

Alternate	Lounge	Fri / Sat	Hungry	Patrons	Price	Raining	Reservation	Type	Wait Estimate	Decision
Yes	Yes	Yes	No	Full	★★	No	Yes	French	10-30	Yes
No	Yes	No	Yes	Full	★★★	Yes	Yes	Italian	1-10	Yes
Yes	No	Yes	No	Full	★	Yes	No	Burger	> 60	No
Yes	Yes	Yes	Yes	Full	★★★	No	Yes	French	10-30	No
Yes	No	No	Yes	Full	★★	No	No	Thai	30-60	No
No	No	No	No	Full	★	No	No	Thai	30-60	No
Yes	No	Yes	Yes	Some	★	No	No	Burger	1-10	Yes
No	No	No	No	Some	★	No	No	Thai	10-30	Yes
No	Yes	No	No	Some	★★	Yes	No	Thai	30-60	Yes
Yes	No	Yes	No	Some	★	Yes	No	Burger	10-30	Yes
No	No	Yes	Yes	None	★★	No	Yes	Italian	>60	No
Yes	No	No	No	None	★	Yes	No	Burger	1-10	No

This is how a Decision Tree looks like

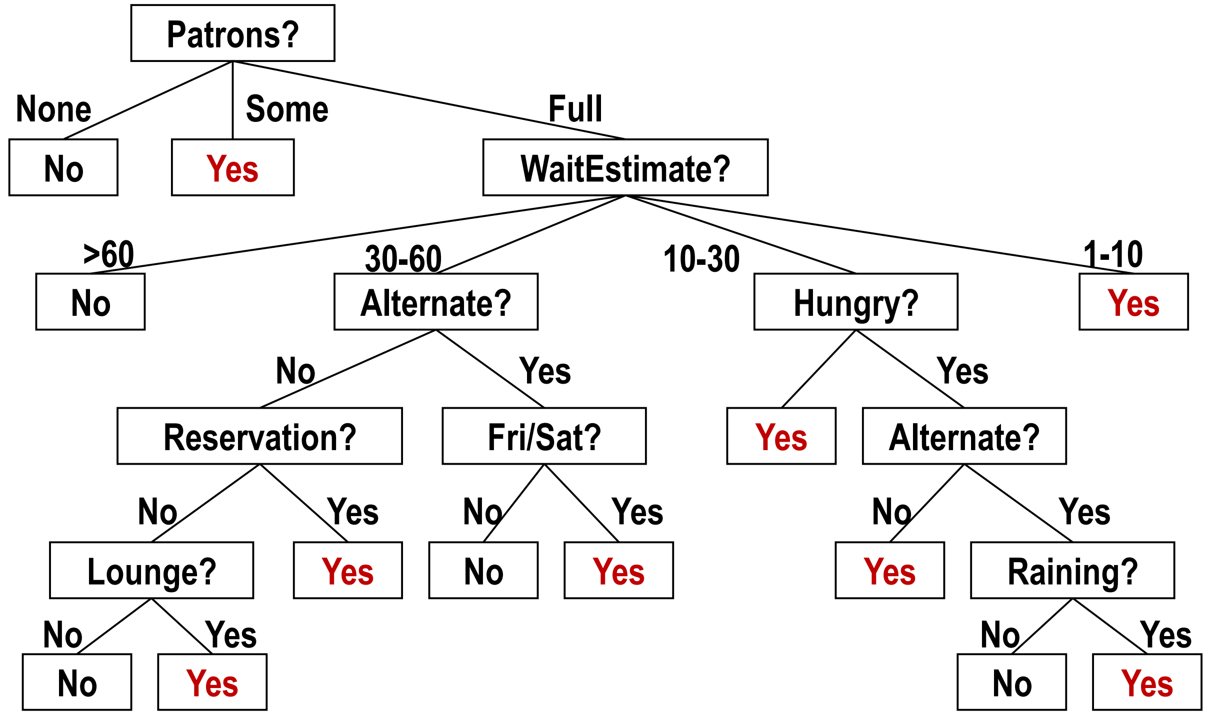


Decision variable: Does the customer wait for a table?

The attributes are chosen such that they influence the decision, based on the evidence (data) provided.

Basic Idea of Constructing Decision Trees

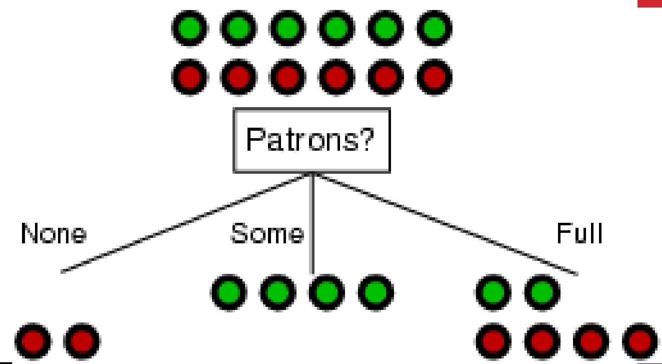
- Find an attribute which helps in separating the Yes answers from the No answers, and make it the root
- Then split the data based on the values of the chosen feature (what does this mean ??)
- For each subset thus created:
  - If the subset contains all Yes or all No, then create a leaf node with that decision
  - Otherwise, the subset contains some Yes and some No, and we recursively use the first two steps
- We may terminate the recursion early to avoid overfitting (will discuss this later)



Key Question:

Which attribute is a good discriminator between the Yes and No decisions?

How good is Patrons?



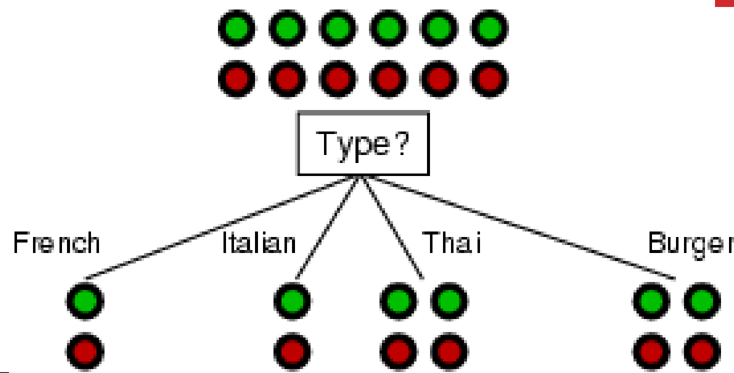
Alternate	Lounge	Fri / Sat	Hungry	Patrons	Price	Raining	Reservation	Type	Wait Estimate	Decision
Yes	Yes	Yes	No	Full	★★	No	Yes	French	10-30	Yes
No	Yes	No	Yes	Full	★★★	Yes	Yes	Italian	1-10	Yes
Yes	No	Yes	No	Full	★	Yes	No	Burger	> 60	No
Yes	Yes	Yes	Yes	Full	★★★	No	Yes	French	10-30	No
Yes	No	No	Yes	Full	★★	No	No	Thai	30-60	No
No	No	No	No	Full	★	No	No	Thai	30-60	No
Yes	No	Yes	Yes	Some	★	No	No	Burger	1-10	Yes
No	No	No	No	Some	★	No	No	Thai	10-30	Yes
No	Yes	No	No	Some	★★	Yes	No	Thai	30-60	Yes
Yes	No	Yes	No	Some	★	Yes	No	Burger	10-30	Yes
No	No	Yes	Yes	None	★★	No	Yes	Italian	>60	No
Yes	No	No	No	None	★	Yes	No	Burger	1-10	No

Some Yes and some No means more uncertainty, and high Entropy

All Yes

All No

How good is Type?



Alternate	Lounge	Fri / Sat	Hungry	Patrons	Price	Raining	Reservation	Type	Wait Estimate	Decision
Yes	Yes	Yes	No	Full	★★	No	Yes	French	10-30	Yes
No	Yes	No	Yes	Full	★★★	Yes	Yes	Italian	1-10	Yes
Yes	No	Yes	No	Full	★	Yes	No	Burger	> 60	No
Yes	Yes	Yes	Yes	Full	★★★	No	Yes	French	10-30	No
Yes	No	No	Yes	Full	★★	No	No	Thai	30-60	No
No	No	No	No	Full	★	No	No	Thai	30-60	No
Yes	No	Yes	Yes	Some	★	No	No	Burger	1-10	Yes
No	No	No	No	Some	★	No	No	Thai	10-30	Yes
No	Yes	No	No	Some	★★	Yes	No	Thai	30-60	Yes
Yes	No	Yes	No	Some	★	Yes	No	Burger	10-30	Yes
No	No	Yes	Yes	None	★★	No	Yes	Italian	>60	No
Yes	No	No	No	None	★	Yes	No	Burger	1-10	No

This is not a good discriminator.
All the subsets have high Entropy

The Decision Tree Learning Algorithm

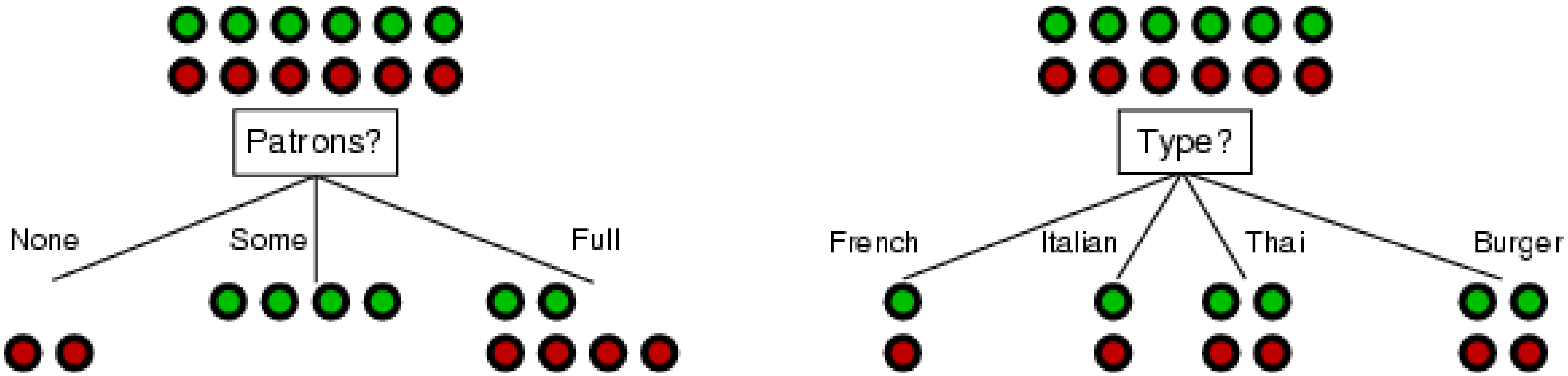
Aim: find a <u>small</u> tree consistent with the training examples

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
```

A formal metric for choosing an attribute

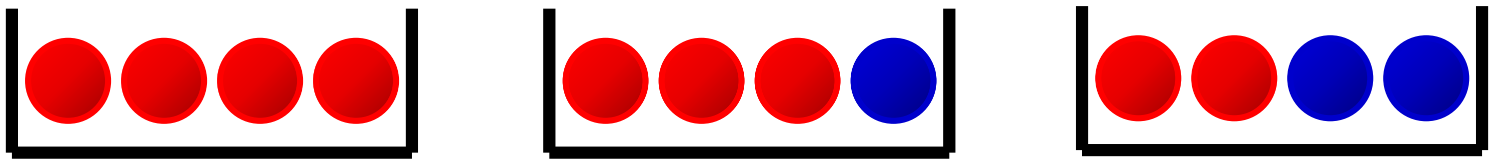
Idea: A good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"



Patrons **is a better choice apparently, but how can we mathematically compare such cases?**

Let's start with a primer on Entropy !!

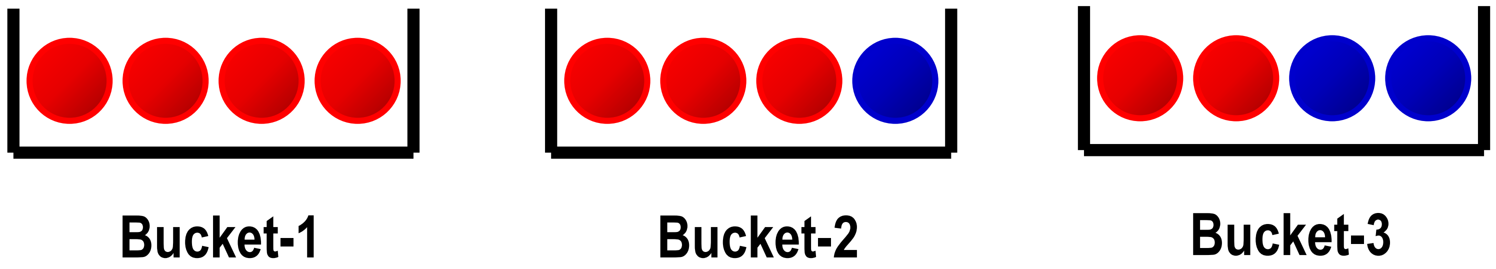
Entropy and Knowledge



Bucket-1 Bucket-2 Bucket-3

- 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

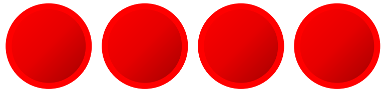
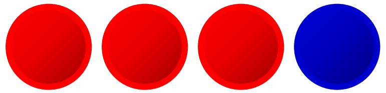
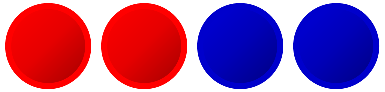
- The probability of finding a specific arrangement in four draws of balls is less for the third bucket because the number of possible arrangements is larger.

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

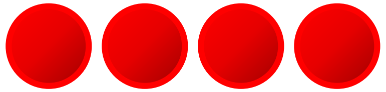
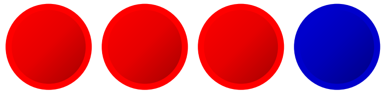
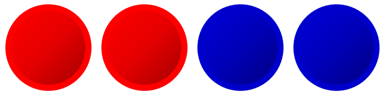
- **Products of many probability terms will make the metric very small and create precision problems**
- **Instead, we can take the logarithm of P(win), which will convert the product into a sum. Since probability terms are fractional, the logarithm will be negative and hence we take its negation**
-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

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

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.

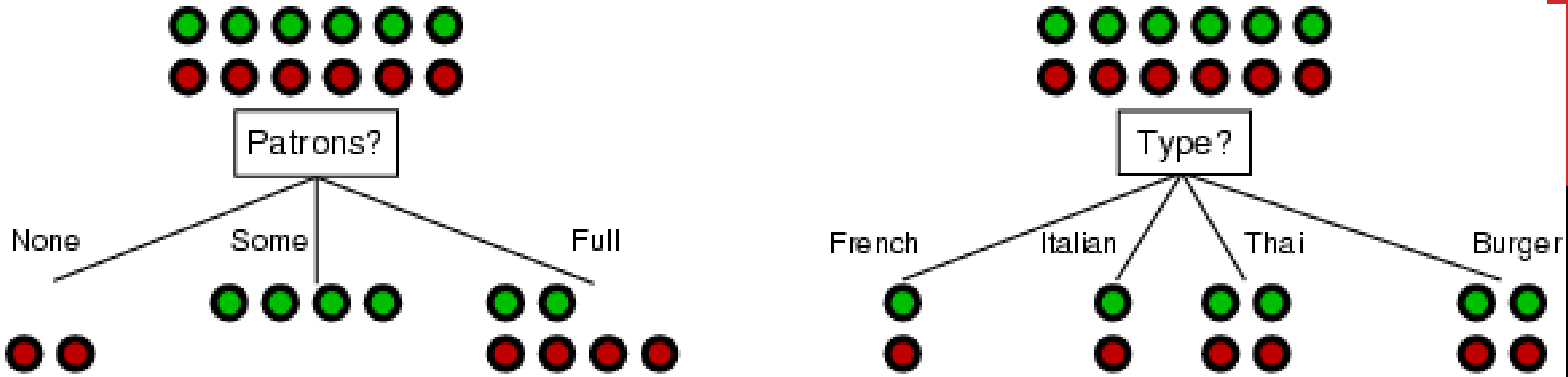
$$\text{remainder}(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

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**

The Problem of Overfitting

Alternate	Lounge	Fri / Sat	Hungry	Patrons	Price	Raining	Reservation	Type	Wait Estimate	Decision
Yes	Yes	Yes	No	Full	★★	No	Yes	French	10-30	Yes
No	Yes	No	Yes	Full	★★★	Yes	Yes	Italian	1-10	Yes
Yes	No	Yes	No	Full	★	Yes	No	Burger	> 60	No
Yes	Yes	Yes	Yes	Full	★★★	No	Yes	French	10-30	No
Yes	No	No	Yes	Full	★★	No	No	Thai	30-60	No
No	No	No	No	Full	★	No	No	Thai	30-60	No
Yes	No	Yes	Yes	Some	★	No	No	Burger	1-10	Yes
No	No	No	No	Some	★	No	No	Thai	10-30	Yes
No	Yes	No	No	Some	★★	Yes	No	Thai	30-60	Yes
Yes	No	Yes	No	Some	★	Yes	No	Burger	10-30	Yes
No	No	Yes	Yes	None	★★	No	Yes	Italian	>60	No
Yes	No	No	No	None	★	Yes	No	Burger	1-10	No
Yes	No	Yes	Yes	Some	★	No	No	Burger	1-10	No

This person waited in keeping with the trend for Patrons = Some

This person did not wait under similar circumstances because she had to pick up her child from school. **Should we then add “child waiting at school” as an attribute?**

Further Readings

- Decision trees are widely used in data sciences, and management. Decision tree learning enables mining of relationships between variables towards influencing targeted decisions.
- Some recent directions:
  - Bayesian Rule Lists
  - Decision Trees for mining Temporal Relations
  - Decision Trees for anomaly detection

Bayesian Rule List for Titanic

if male and adult then survival probability 21% (19%–23%)

else if 3rd class then survival probability 44% (38%–51%)

else if 1st class then survival probability 96% (92%–99%)

else survival probability 88% (82%–94%)

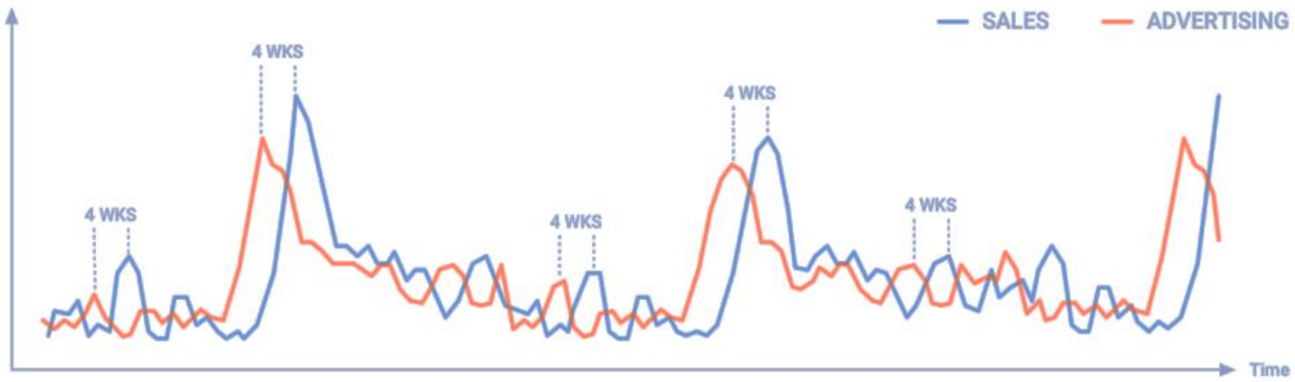
In parentheses is the 95% credible interval for the survival probability

Time Series: A time series is a sequence of observations recorded in time order.

Example: Simulation dumps, Sales and Advertising, industrial manufacturing processes, vehicle parameters, health monitoring, QOS of networks, etc.

We often need to find temporal causes of events that we see.

Antonio A Bruto da Costa, Goran Frehse, Pallab Dasgupta, Flexible Mining of Prefix Sequences from Time-Series Traces, https://arxiv.org/abs/1905.12262




```
LowSales && HighAdvt |-> ##[4:6wks] HighSales
```

The Mushroom Problem

You are stranded on a deserted island. Mushrooms of various types grow widely all over the island, but no other food is anywhere to be found. Some of the mushrooms have been determined as poisonous and others as not (determined by your former companions' trial and error). You are the only one remaining on the island. You have the following data to consider:

Example	Not Heavy	Smelly	Spotted	Smooth	Edible
A	1	0	0	0	1
B	1	0	1	0	1
C	0	1	0	1	1
D	0	0	0	1	0
E	1	1	1	0	0
F	1	0	1	1	0
G	1	0	0	1	0
H	0	1	0	0	0
U	0	1	1	1	?
V	1	1	0	1	?
W	1	1	0	0	?

You know whether or not mushrooms A through H are poisonous, but you do not know about U through W.

The Mushroom Problem

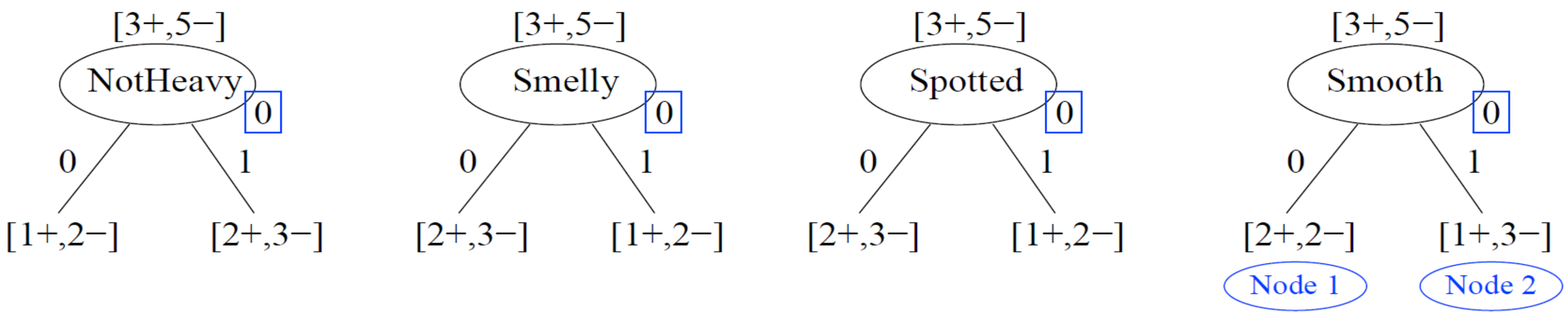
Considering only the data for mushrooms A through H, what is the entropy of Edible? Write the formula for entropy and then use it in your computation.

Example	Not Heavy	Smelly	Spotted	Smooth	Edible
A	1	0	0	0	1
B	1	0	1	0	1
C	0	1	0	1	1
D	0	0	0	1	0
E	1	1	1	0	0
F	1	0	1	1	0
G	1	0	0	1	0
H	0	1	0	0	0
U	0	1	1	1	?
V	1	1	0	1	?
W	1	1	0	0	?

$$\begin{aligned} H_{Edible} &= H[3+,5-] \stackrel{def.}{=} -\frac{3}{8}\cdot\log_2\frac{3}{8} - \frac{5}{8}\cdot\log_2\frac{5}{8} = \frac{3}{8}\cdot\log_2\frac{8}{3} + \frac{5}{8}\cdot\log_2\frac{8}{5} \\ &= \frac{3}{8}\cdot 3 - \frac{3}{8}\cdot\log_2 3 + \frac{5}{8}\cdot 3 - \frac{5}{8}\cdot\log_2 5 = 3 - \frac{3}{8}\cdot\log_2 3 - \frac{5}{8}\cdot\log_2 5 \\ &\approx 0.9544 \end{aligned}$$

The Mushroom Problem

Which attribute should you choose as the root of a decision tree?

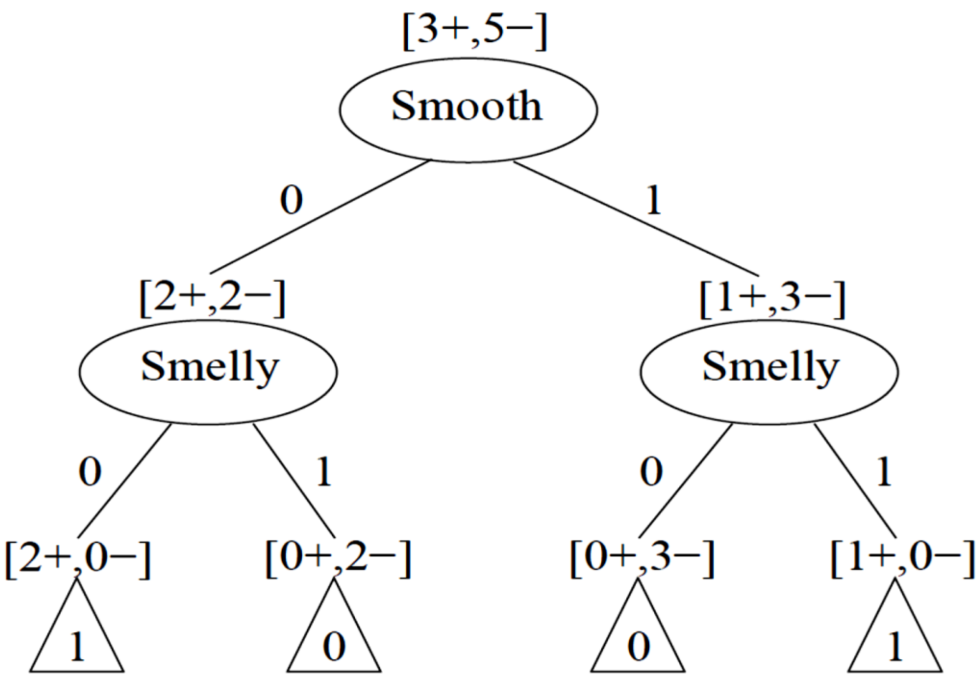


$$
\begin{aligned}
H_{0/Smooth} &\stackrel{def.}{=} \frac{4}{8} H[2+,2-] + \frac{4}{8} H[1+,3-] = \frac{1}{2}\cdot 1 + \frac{1}{2}\left(\frac{1}{4}\log_2\frac{4}{1} + \frac{3}{4}\log_2\frac{4}{3}\right) \\
&= \frac{1}{2} + \frac{1}{2}\left(\frac{1}{4}\cdot 2 + \frac{3}{4}\cdot 2 - \frac{3}{4}\log_2 3\right) = \frac{1}{2} + \frac{1}{2}\left(2 - \frac{3}{4}\log_2 3\right) \\
&= \frac{1}{2} + 1 - \frac{3}{8}\log_2 3 = \frac{3}{2} - \frac{3}{8}\log_2 3 \approx 0.9056
\end{aligned}
$$

$$
\begin{aligned}
IG_{0/Smooth} &\stackrel{def.}{=} H_{Edible} - H_{0/Smooth} \\
&= 0.9544 - 0.9056 = 0.0488
\end{aligned}
$$

The Mushroom Problem

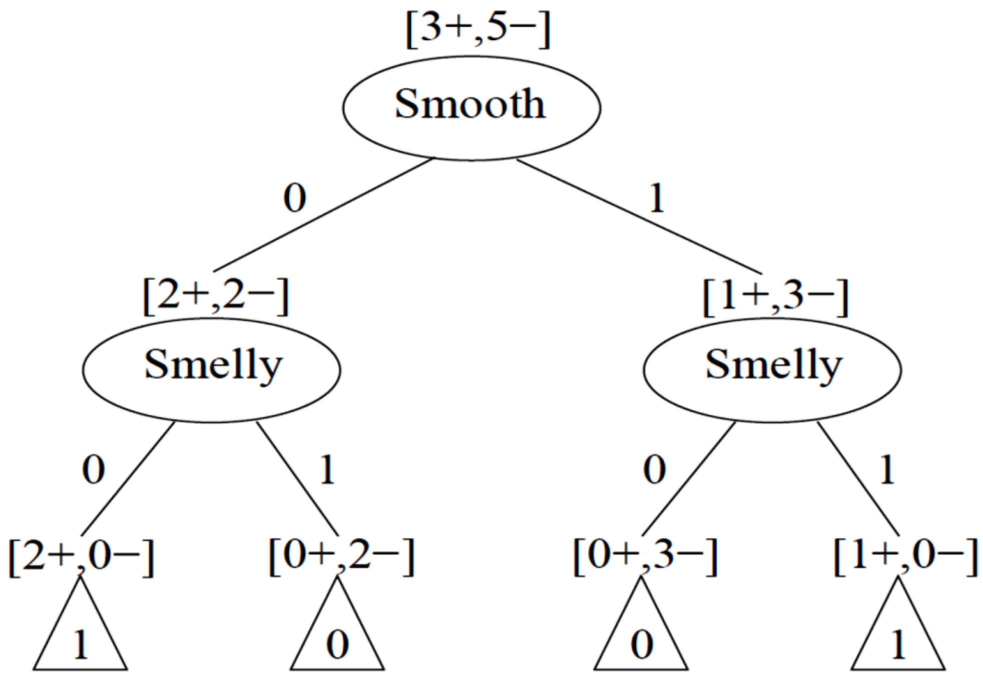
Build a decision tree to classify mushrooms as poisonous or not (not all attributes may be needed)



Example	Not Heavy	Smelly	Spotted	Smooth	Edible
A	1	0	0	0	1
B	1	0	1	0	1
C	0	1	0	1	1
D	0	0	0	1	0
E	1	1	1	0	0
F	1	0	1	1	0
G	1	0	0	1	0
H	0	1	0	0	0
U	0	1	1	1	?
V	1	1	0	1	?
W	1	1	0	0	?

The Mushroom Problem

Classify mushrooms U, V and W using the decision tree as poisonous or not poisonous.



Example	Not Heavy	Smelly	Spotted	Smooth	Edible
A	1	0	0	0	1
B	1	0	1	0	1
C	0	1	0	1	1
D	0	0	0	1	0
E	1	1	1	0	0
F	1	0	1	1	0
G	1	0	0	1	0
H	0	1	0	0	0
U	0	1	1	1	?
V	1	1	0	1	?
W	1	1	0	0	?

Classification of *test instances*:

U	$Smooth = 1,\ Smelly = 1 \Rightarrow Edible = 1$
V	$Smooth = 1,\ Smelly = 1 \Rightarrow Edible = 1$
W	$Smooth = 0,\ Smelly = 1 \Rightarrow Edible = 0$