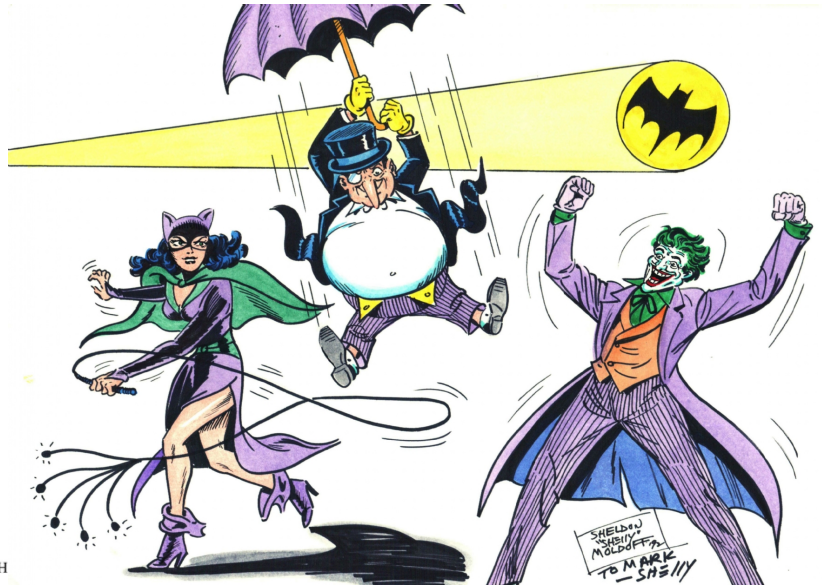
CS167: Machine Learning

Entropy Calculations
In-class Activity on Entropy + Information Gain

Monday, March 2nd, 2026

Drake
UNIVERSITY

Decision: Are these comic book characters good or evil?



Problem: Is a comic book character good or bad?

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

Training data

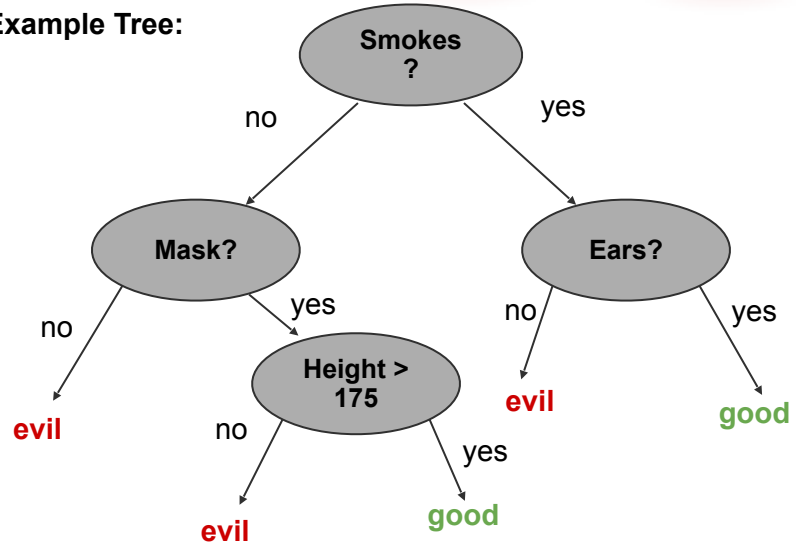
Test Data

Dataset and example from Dr. Kilian Weinberger @Cornell

Let's Make a Tree

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

Example Tree:



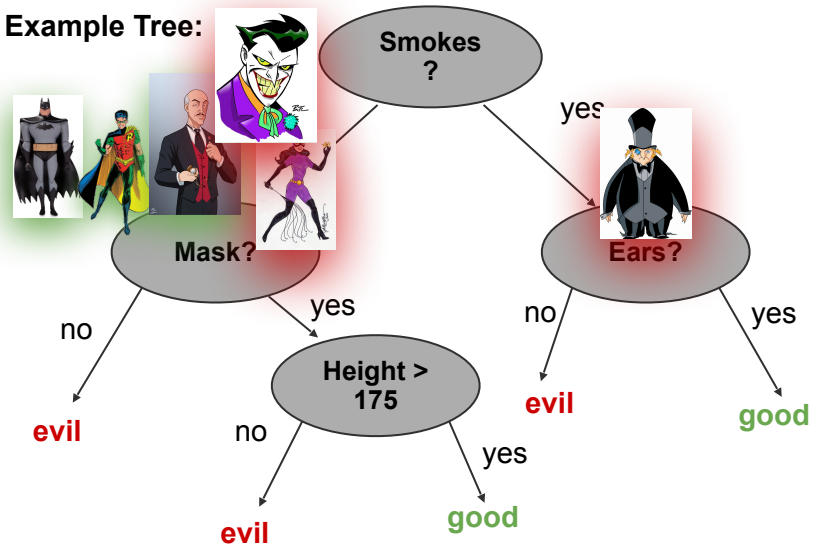
Question: Is this a good tree?

Is this tree **consistent:** would it classify everyone correctly?

Let's Make a Tree

Let's classify the characters based on the value of attribute 'Smokes'

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?



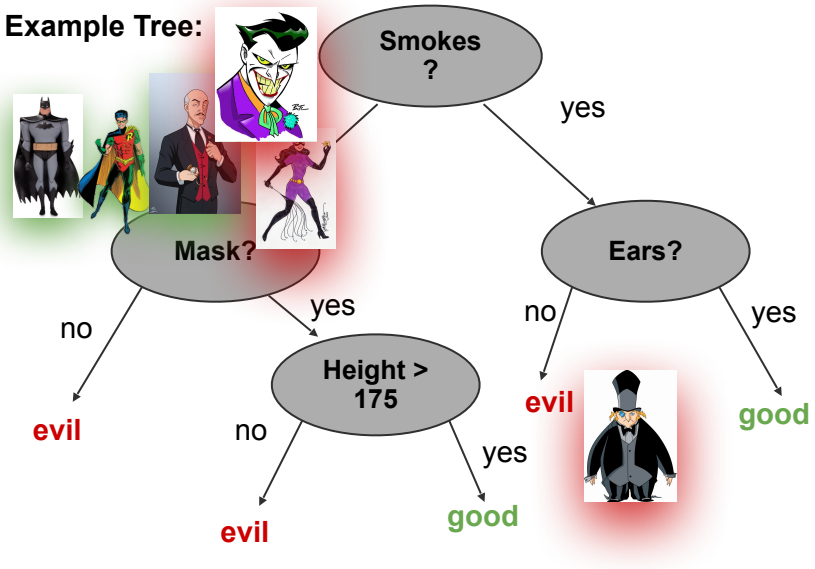
Question: Is this a good tree?

Is this tree **consistent:** would it classify everyone correctly?

Let's Make a Tree

Let's classify the characters based on the value of attribute 'Ears'

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?



Question: Is this a good tree?

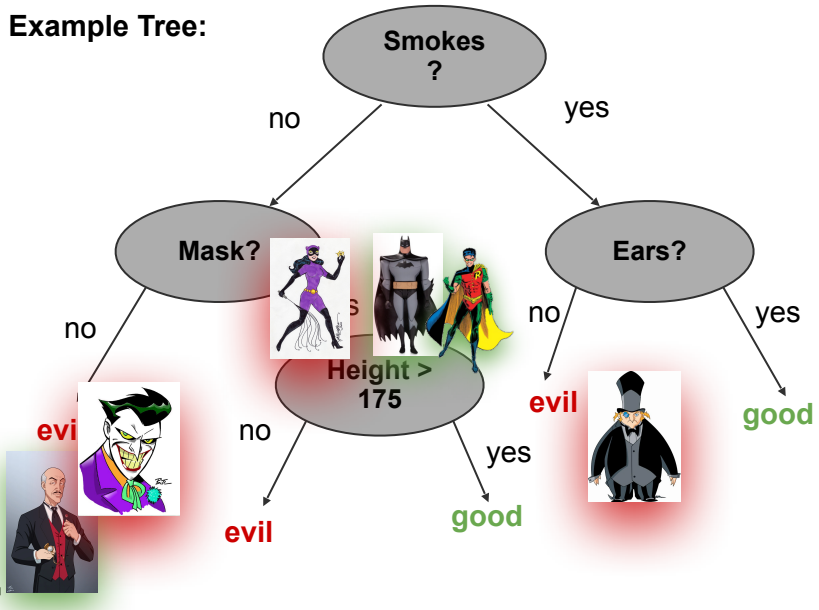
Is this tree **consistent:** would it classify everyone correctly?

Let's Make a Tree

Let's classify the characters based on the value of attribute 'Mask'

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

Example Tree:



Question: Is this a good tree?

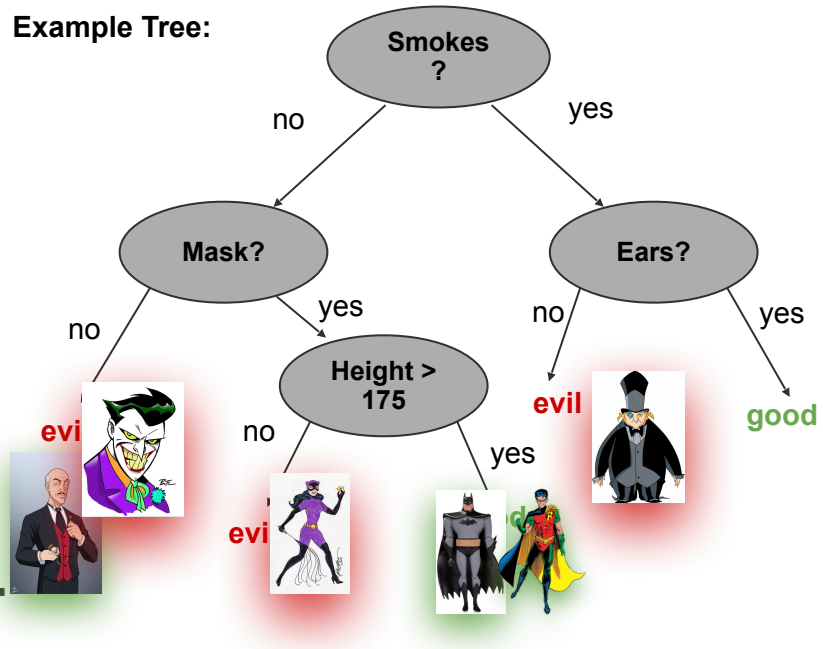
Is this tree **consistent:** would it classify everyone correctly?

Let's Make a Tree

Let's classify the characters based on the value of attribute 'Height'

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

Example Tree:



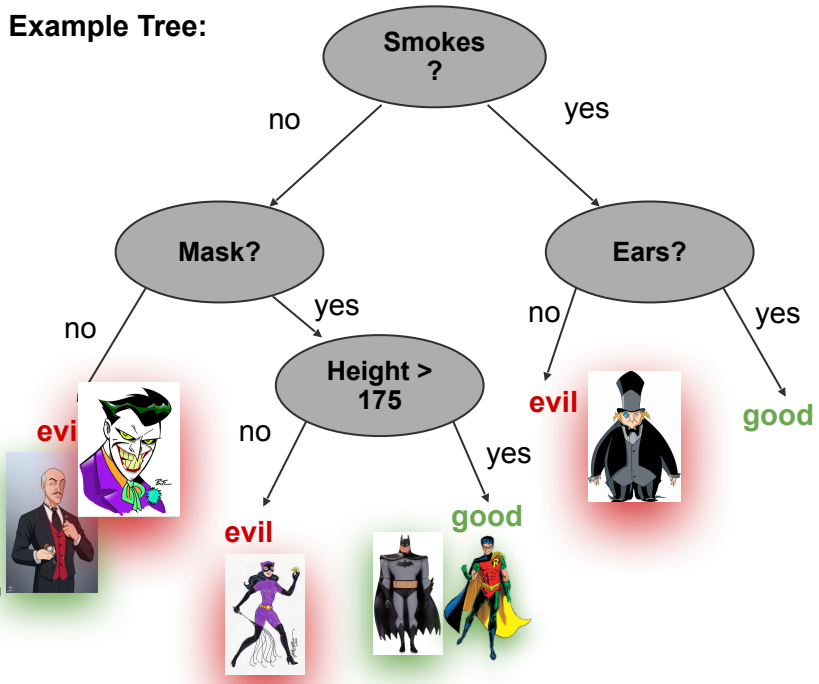
Answer:

- No, it is not consistent. It misclassified Alfred as evil

Let's Make a Tree

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

Example Tree:

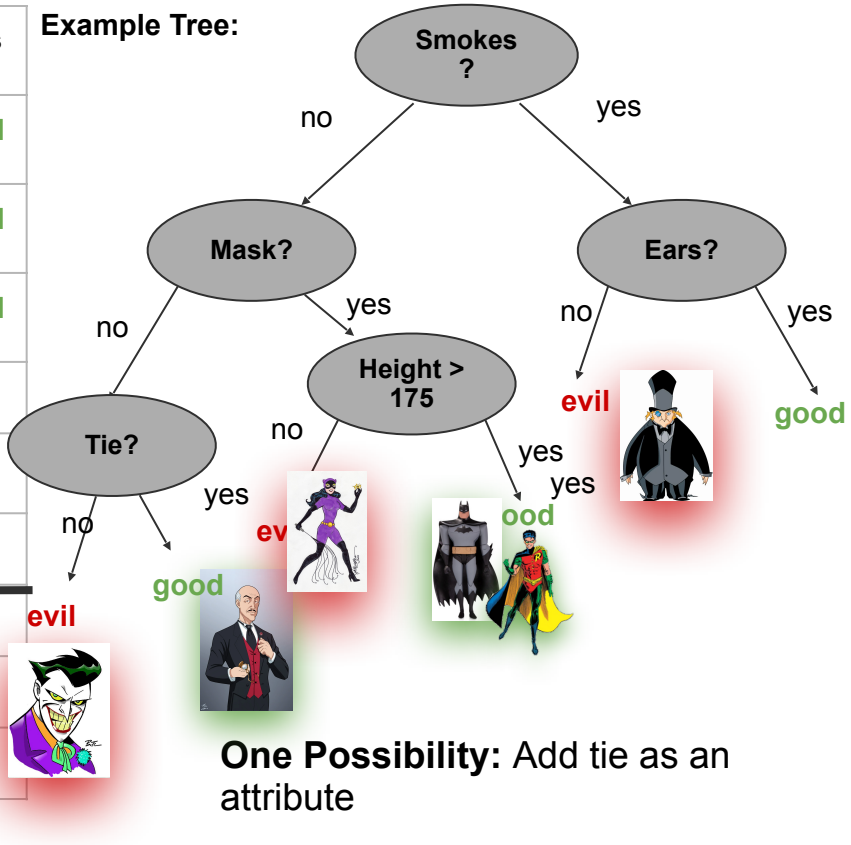


Question: What can we do to make this tree consistent?

Let's Make a Tree

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

Example Tree:



One Possibility: Add tie as an attribute

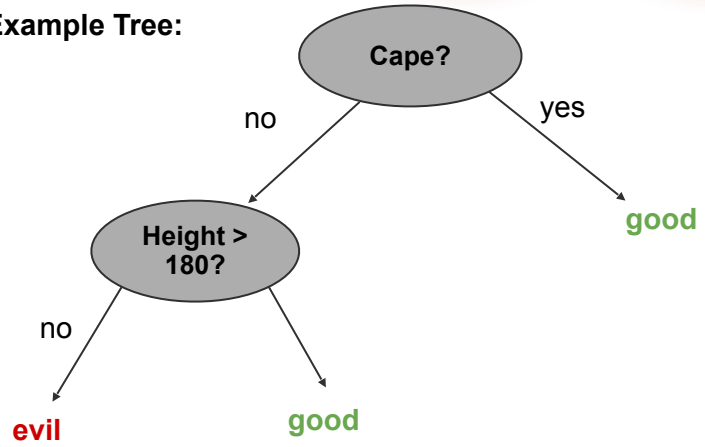
Activity: What is the smallest consistent tree?

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

Let's Make a Tree

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

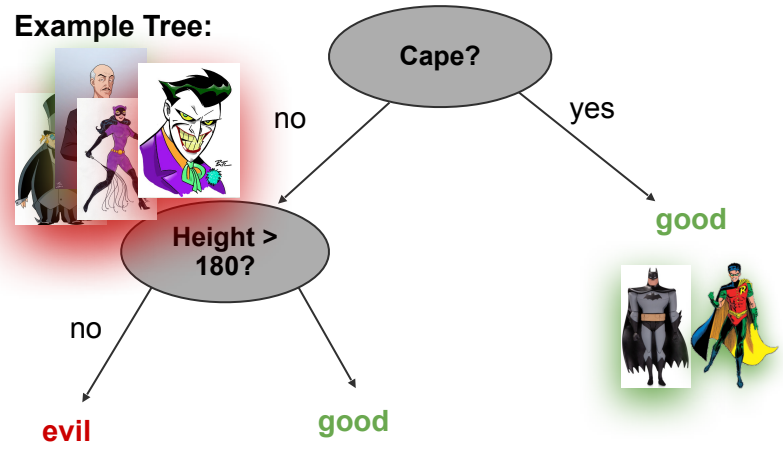
Example Tree:



Let's Make a Tree

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

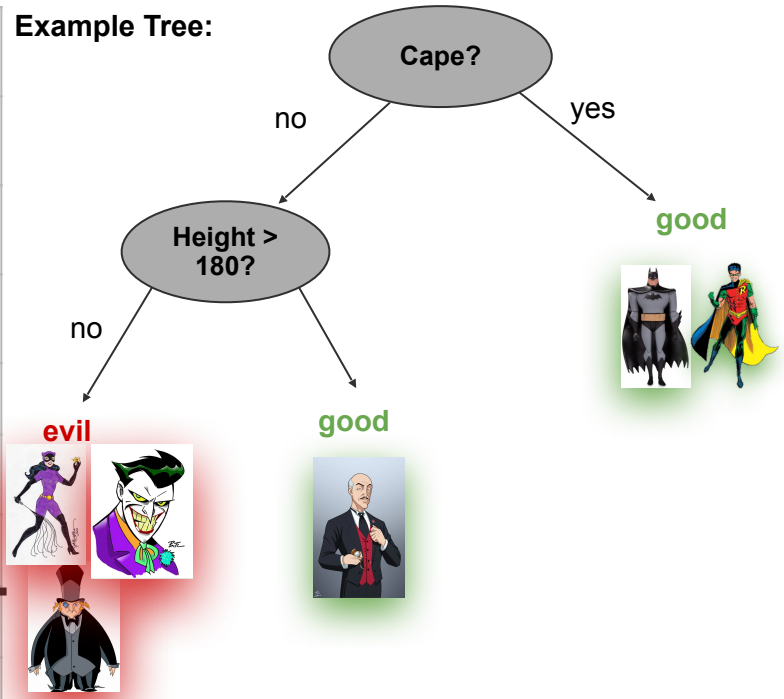
Example Tree:



Let's Make a Tree

	mask	cape	tie	ears	smokes	height	class
Batman	y	y	n	y	n	180	good
Robin	y	y	n	n	n	176	good
Alfred	n	n	y	n	n	185	good
Penguin	n	n	y	n	y	140	evil
Catwoman	y	n	n	y	n	170	evil
Joker	n	n	n	n	n	179	evil
Batgirl	y	y	n	y	n	165	?
Riddler	y	n	n	n	n	182	?
Your Date	n	y	y	y	y	181	?

Example Tree:



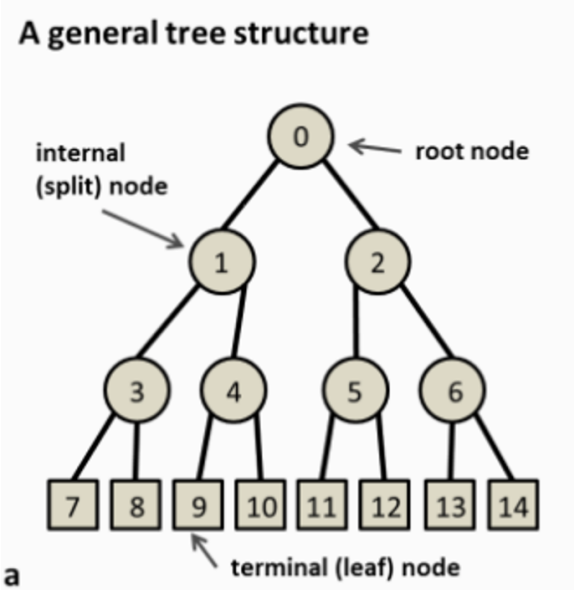
Today's Agenda

- Decision Tree

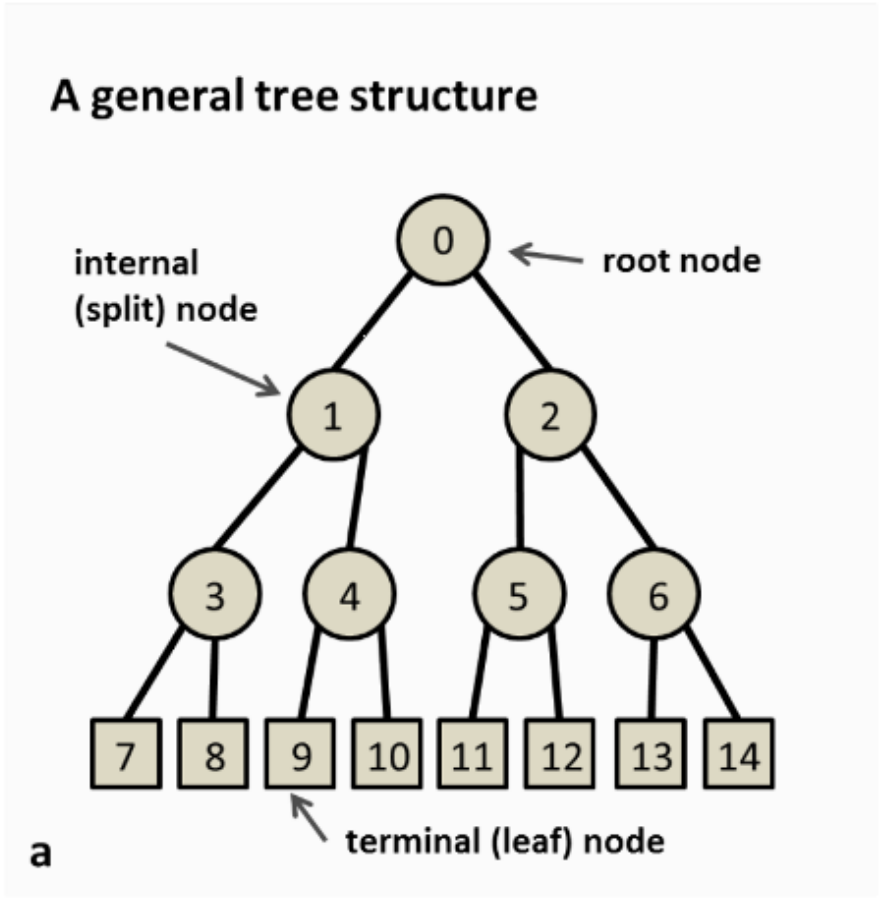
- Entropy

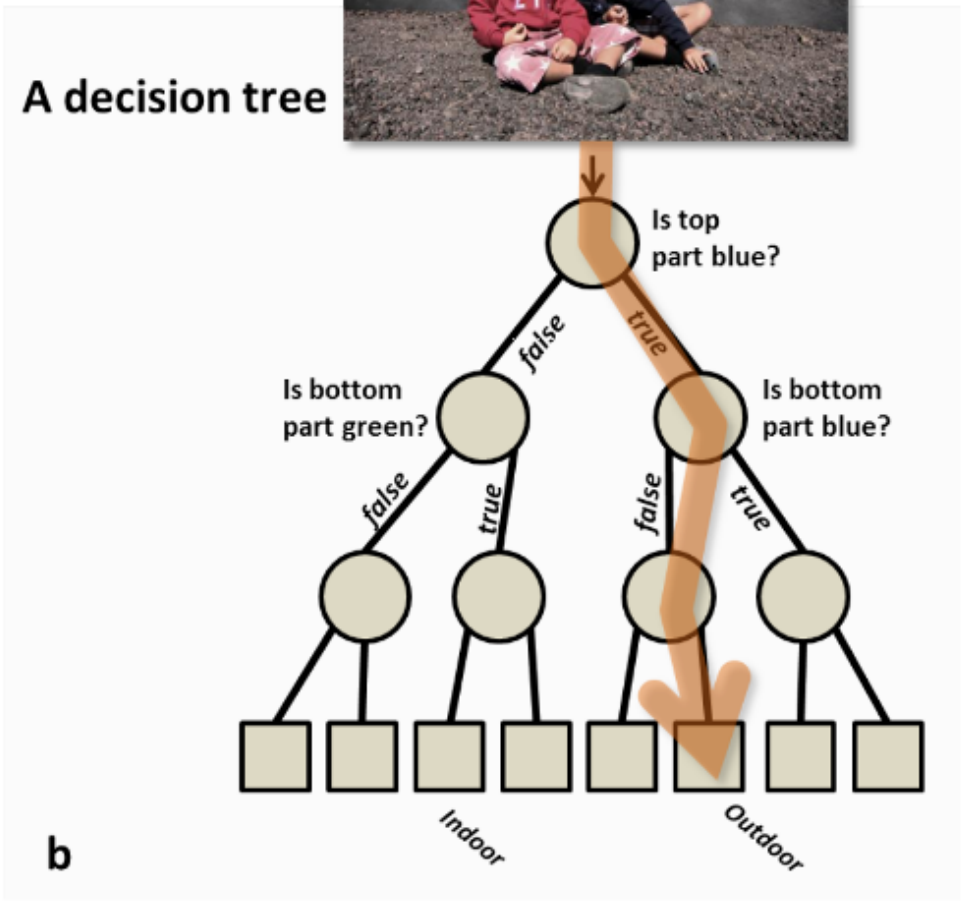
Decision Tree: Tree Data Structure

- **Tree**: a common data structure that simulates a hierarchical tree structure, with a root value and subtrees of children with a parent node, represented as a set of liked nodes.



Decision Tree: Tree Data Structure





Decision Tree: Another Example

Features or Attributes:

Target feature is whether or not a person will stay at a restaurant (T, F) with the following predictor features:

1. **Alternate:** whether there is a suitable alternative restaurant nearby.
2. **Bar:** whether the restaurant has a comfortable bar area to wait in.
3. **Fri/Sat:** true on Fridays and Saturdays.
4. **Hungry:** whether we are hungry.
5. **Patrons:** how many people are in the restaurant (values are None, Some, and Full).
6. **Price:** the restaurant's price range (one, two, or three $'s)
7. **Raining:** whether it is raining outside.
8. **Reservation:** whether we made a reservation.
9. **Type:** the kind of restaurant (French, Italian, Thai, or burger).
10. **Est:** the wait estimated by the host (0–10 minutes, 10–30, 30–60, or >60).

Decision Tree: Restaurant dataset

Predictors

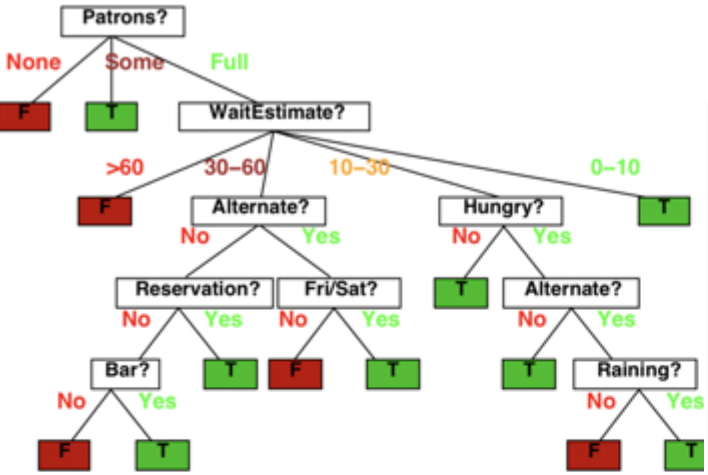
Target

Ex	*Alt*	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	Wait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Decision Tree: Restaurant dataset

How can we build this decision tree from the data below?

Example Tree



Predictors

Ex	*Alt*	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	Wait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Target

Consistency and Generalization

- Is this tree consistent with the training examples?
  - do all of the training examples get categorized appropriately?

- Will this tree generalize well to new examples?
  - how well will new examples (test set) perform?

Growing or Building a Decision Tree

- Great, now how do I build (grow) a tree?
  - One algorithm that builds a decision tree is called:
    - **ID3 Decision Tree Learning Algorithm**

ID3 Decision Tree Learning (Main Loop)

1. A ← select the "best" decision predictor for next node
2. Assign A as decision predictor for node
3. For each possible attribute (or value) of A, create new descendant of node
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

But… what does 'best' mean?

How would we go about deciding which node is the 'best?

Choosing a feature

Which of these features do you think is a better choice for putting at the root of the decision tree?

Ex	*Alt*	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	Wait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Today's Agenda

- Decision Tree

- Entropy

Choosing a feature

Patrons column is one of the candidate predictor for the root node

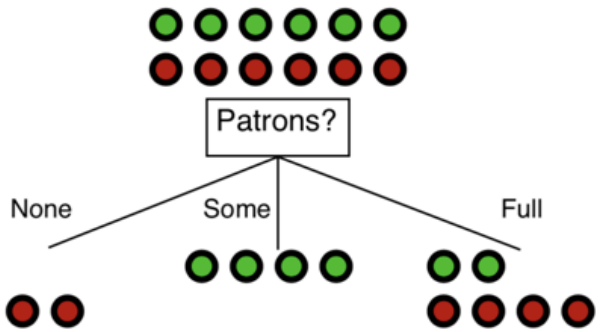
Type column is another candidate predictor for the root node

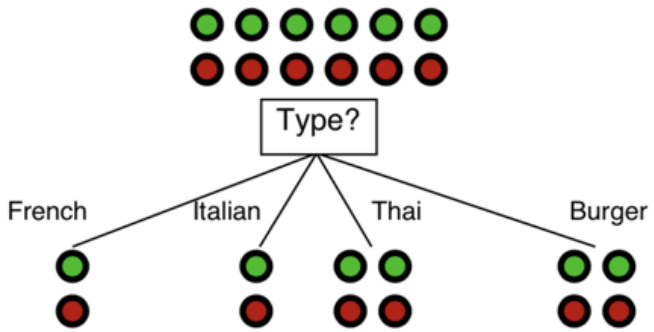
Wait column is target value

	Predictor feature										Target feature
Ex	*Alt*	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	Wait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Which of these features do you think is a better choice for putting at the root of the decision tree?

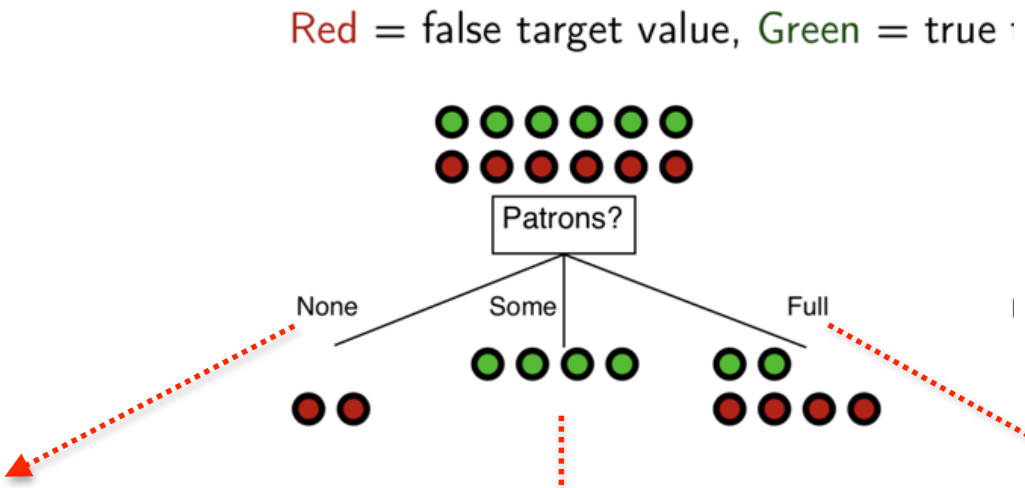
Red = false target value, Green = true target value





Choosing a feature

Let's consider Patrons column to be the candidate predictor for the root node



None patron value

p(red) = 2 red out of total 2 balls: 2/2 = 1.0
p(green) = 0 green out of total 2 balls: 0/2 = 0.0
<p_red, p_green> = <1.0, 0.0>

Full patron value

p(red) = 4 red out of total 6 balls: 4/6 = 0.67
p(green) = 2 green out of total 6 balls: 2/6 = 0.33
<p_red, p_green> = <0.67, 0.33>

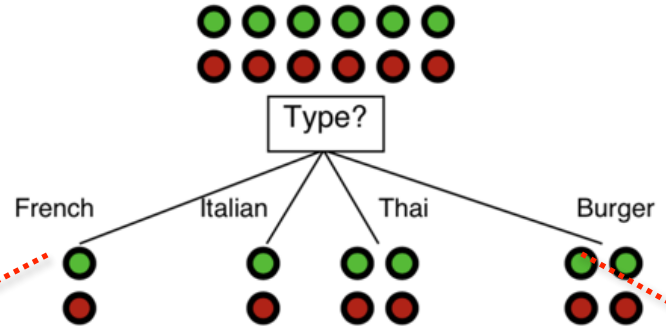
Some patron value

p(red) = 0 red out of total 4 balls: 0/4 = 0.0
p(green) = 4 green out of total 4 balls: 4/4 = 1.0
<p_red, p_green> = <0.0, 1.0>

Wait column is the target value

Choosing a feature

Let's consider Type column to be the candidate predictor for the root node



French *type* value

p(red) = 1 red out of total 2 balls: 1/2 = 0.5
p(green) = 1 green out of total 2 balls: 1/2 = 0.5
<p_red, p_green> = <0.5, 0.5>

Italian *type* value

p(red) = 1 red out of total 2 balls: 1/2 = 0.5
p(green) = 1 green out of total 2 balls: 1/2 = 0.5
<p_red, p_green> = <0.5, 0.5>

Thai *type* value

p(red) = 2 red out of total 4 balls: 2/4 = 0.5
p(green) = 2 green out of total 4 balls: 2/4 = 0.5
<p_red, p_green> = <0.5, 0.5>

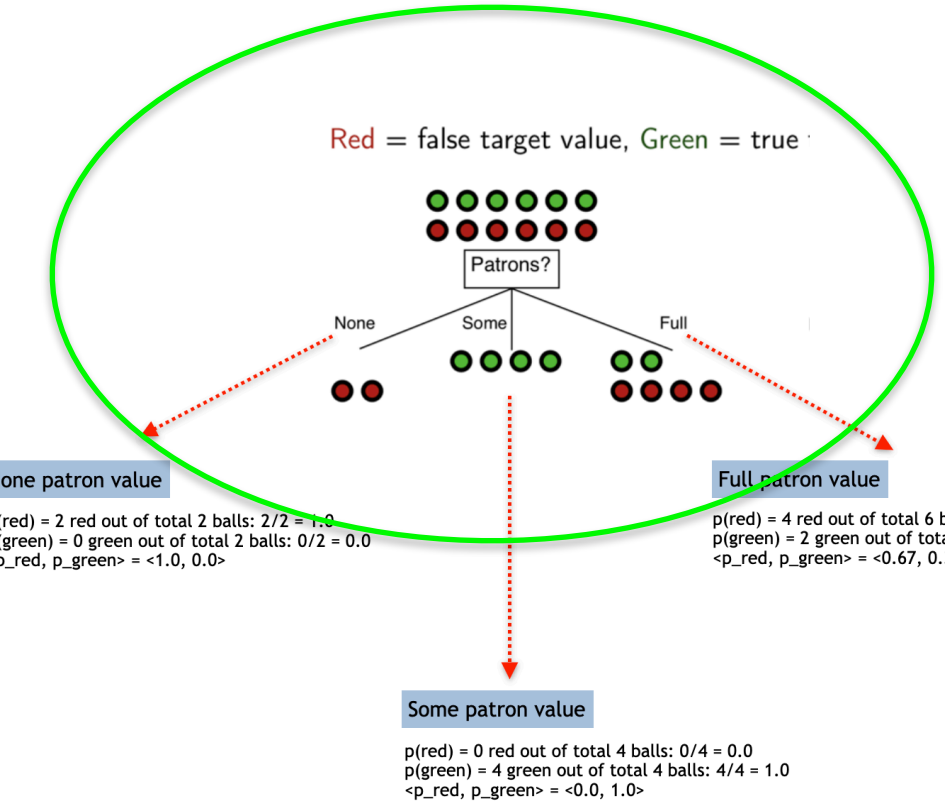
Burger *type* value

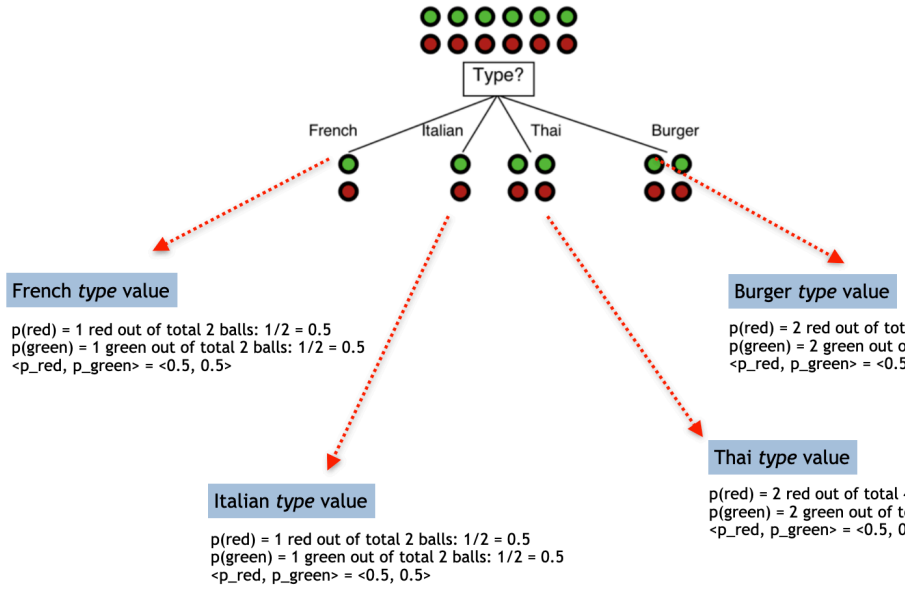
p(red) = 2 red out of total 4 balls: 2/4 = 0.5
p(green) = 2 green out of total 4 balls: 2/4 = 0.5
<p_red, p_green> = <0.5, 0.5>

Wait column is the target value

Choosing a feature

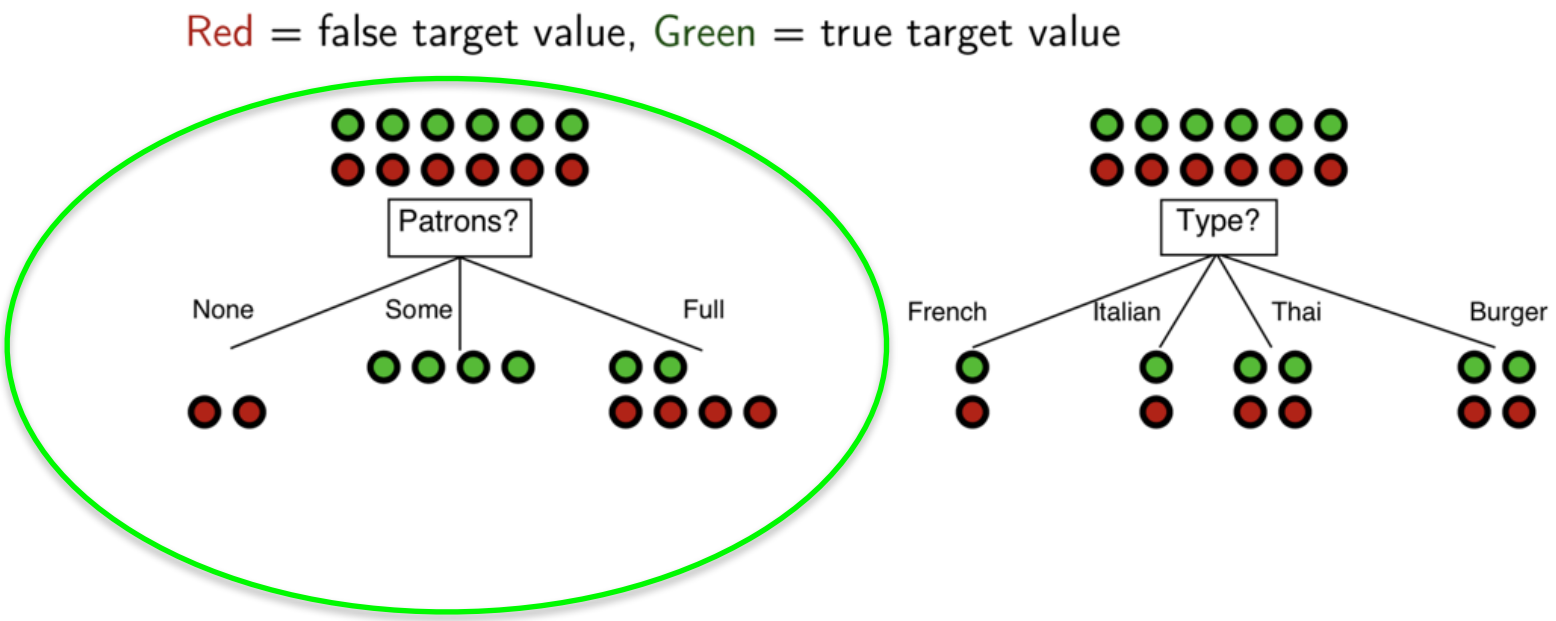
Patrons is a better choice for putting at the root of the decision tree — it gives more information about the classification





Decision Tree: Choosing a feature

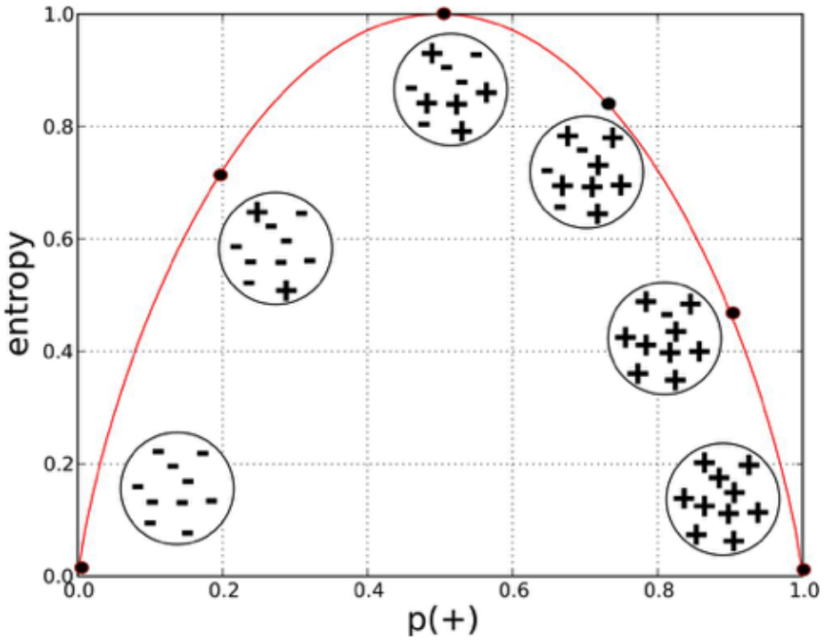
Idea: a good feature splits the examples into **subsets that are as pure as possible** (ideally) "all positive" or "all negative"



Decision Tree: Entropy

Entropy is a measure of impurity/randomness

- **High entropy**: more evenly split classes - highly **unpredictable**
- **Low entropy**: mostly one class - highly **predictable**



Calculating Entropy

Calculating the entropy when probability is $\langle P_1,...,P_c \rangle$:

$$Entropy(\langle P_1, \ldots, P_c \rangle) = \sum_{i=1}^{c} -P_i \log_2 P_i$$

- entropy $\langle 0.5, 0.5 \rangle$
  - $-0.5 \log_2 (0.5) - 0.5 \log_2 (0.5) = 1$ -{p_r*log2(p_r) + p_g*log2(p_g)}
- entropy $\langle 0.9, 0.1 \rangle$
  - $-0.9 \log_2 (0.9) - 0.1 \log_2 (0.1) \approx 0.47$ -{p_r*log2(p_r) + p_g*log2(p_g)}
- entropy $\langle 0.64, 0.36 \rangle$
  - $-0.64 \log_2 (0.64) - 0.36 \log_2 (0.36) \approx 0.94$ -{p_r*log2(p_r) + p_g*log2(p_g)}
- entropy $\langle 0.25, 0.25, 0.5 \rangle$
  - $-0.25 \log_2 (0.25) - 0.25 \log_2 (0.25) - 0.5 \log_2 (0.5) = 1.5$ -{p_r*log2(p_r) + p_g*log2(p_g)}

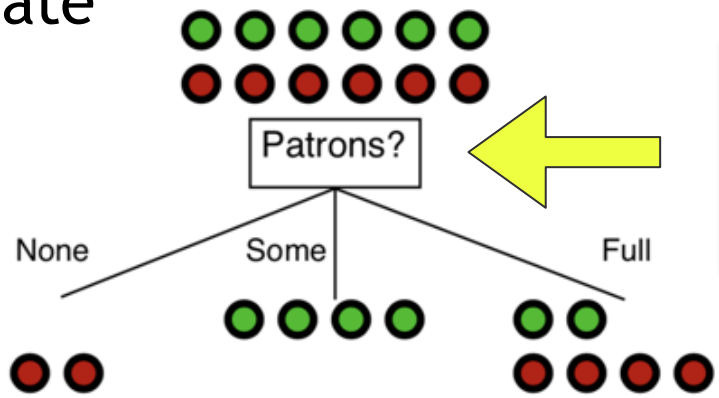
The maximum entropy is $\log_2(k)$ where k is the number of categories. It is not always bounded by 0 and 1

$$\log_2 (3) = 1.584962501$$

Decision Tree: Information Gain

- Information gain for a particular attribute (eg, *Patron*) can be computed in a 3 step process
  - **Step 1:** calculate entropy prior entropy_before
  - **Step 2:** calculate expected entropy entropy_after
  - **Step 3:** subtract expected entropy from prior entropy

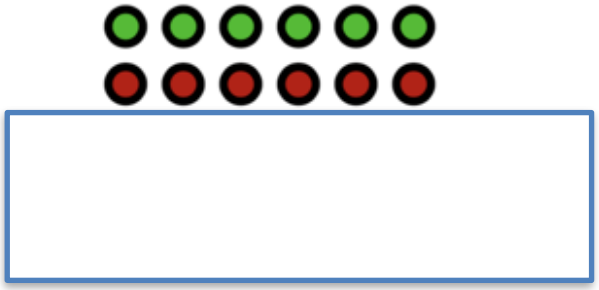
- Let's calculate Information gain for attribute *Patron* as a candidate



p(green) = 6 green balls out of 12 total 6/12 = 0.5
p(red) = 6 red balls out of 12 total = 0.5
<green, red> = <0.5, 0.5>
Entropy_before = -0.5*log2(0.5) - 0.5*log2(0.5) = 1.0

Step 1: Calculating Entropy Prior

- Calculate the entropy of the distribution of the classes before the node you are testing (we will refer this as **entropy_before**)
  - split of the examples when attribute *Patrons* is under consideration
    - Out of 14 examples, if I have 9 positive examples and 5 negative examples my **prior** is:
    - ⟨9/14, 5/14⟩ ≈ ⟨0.64, 0.36⟩

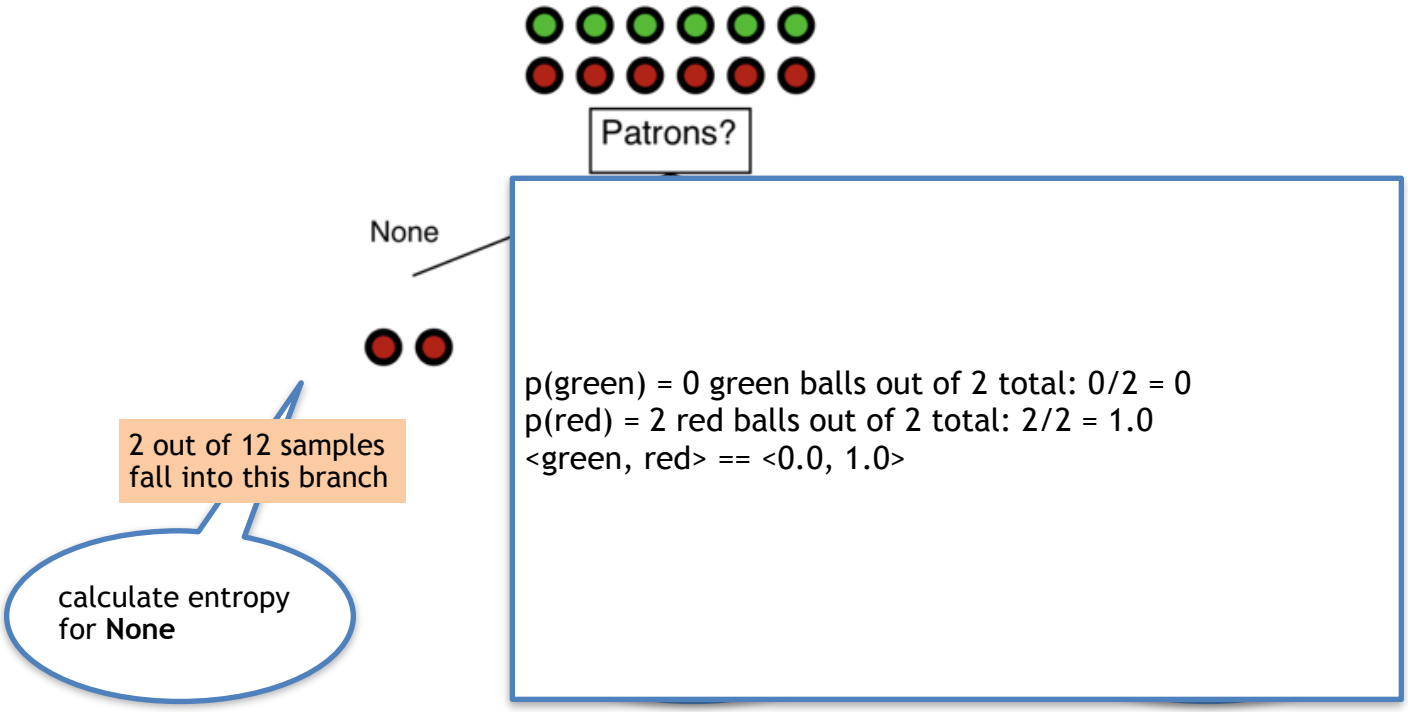


- entropy of prior ⟨0.5, 0.5⟩ -{p_r*log2(p_r) + p_g*log2(p_g)}
  - $-0.5 \log_2 (0.5) - 0.5 \log_2 (0.5) = 1$

So, Entropy_before = 1

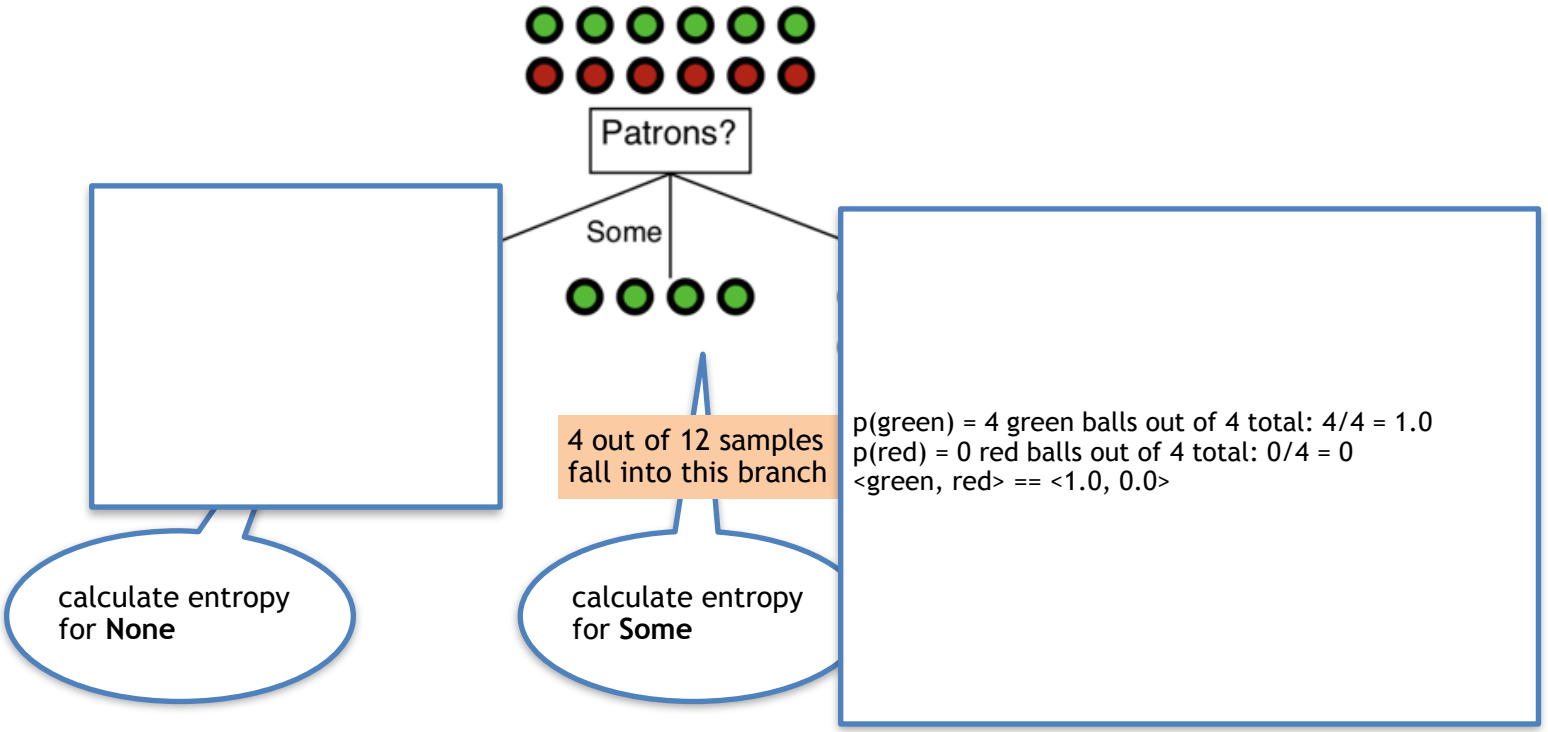
Step 2: Expected Entropy Calculation

The expected entropy for a feature is defined as the weighted sum of entropies multiplied by the fraction of samples that belong to each set.



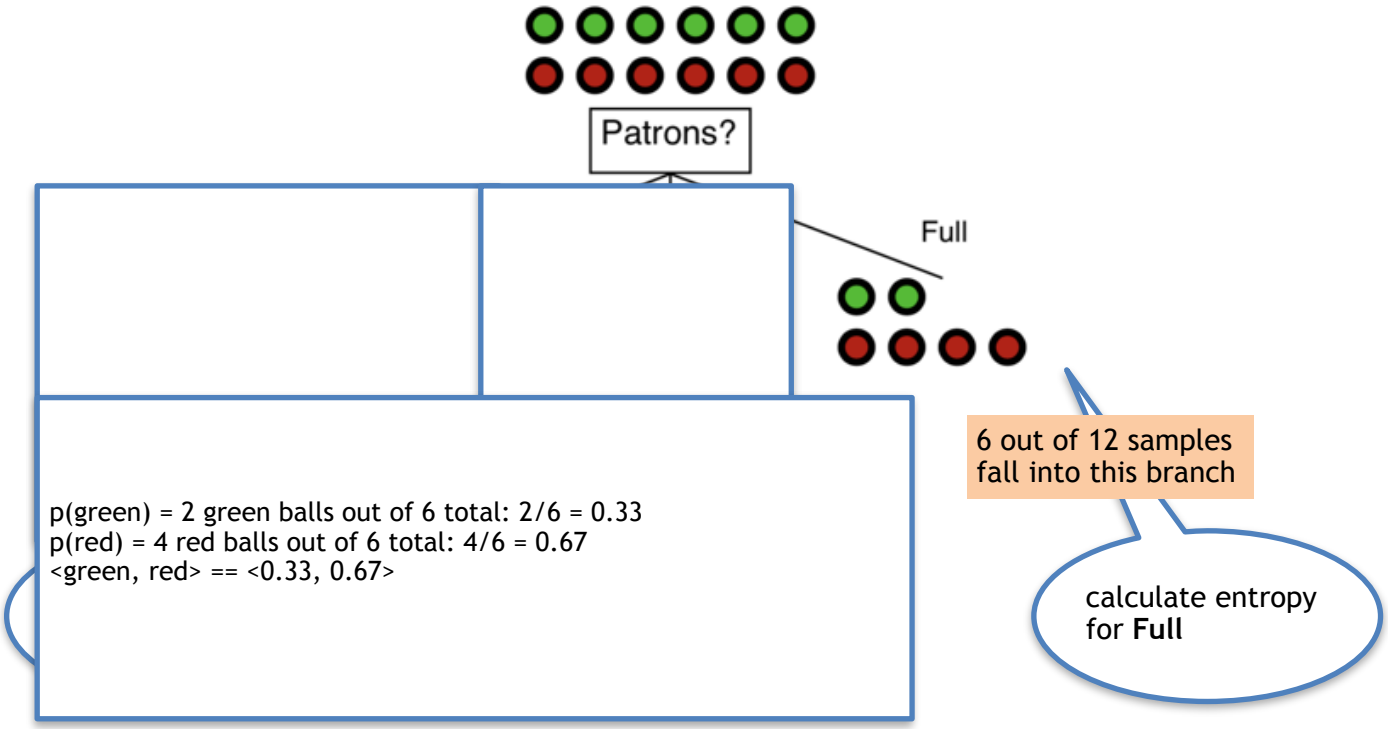
Step 2: Expected Entropy Calculation

The expected entropy for a feature is defined as the weighted sum of entropies multiplied by the fraction of samples that belong to each set.

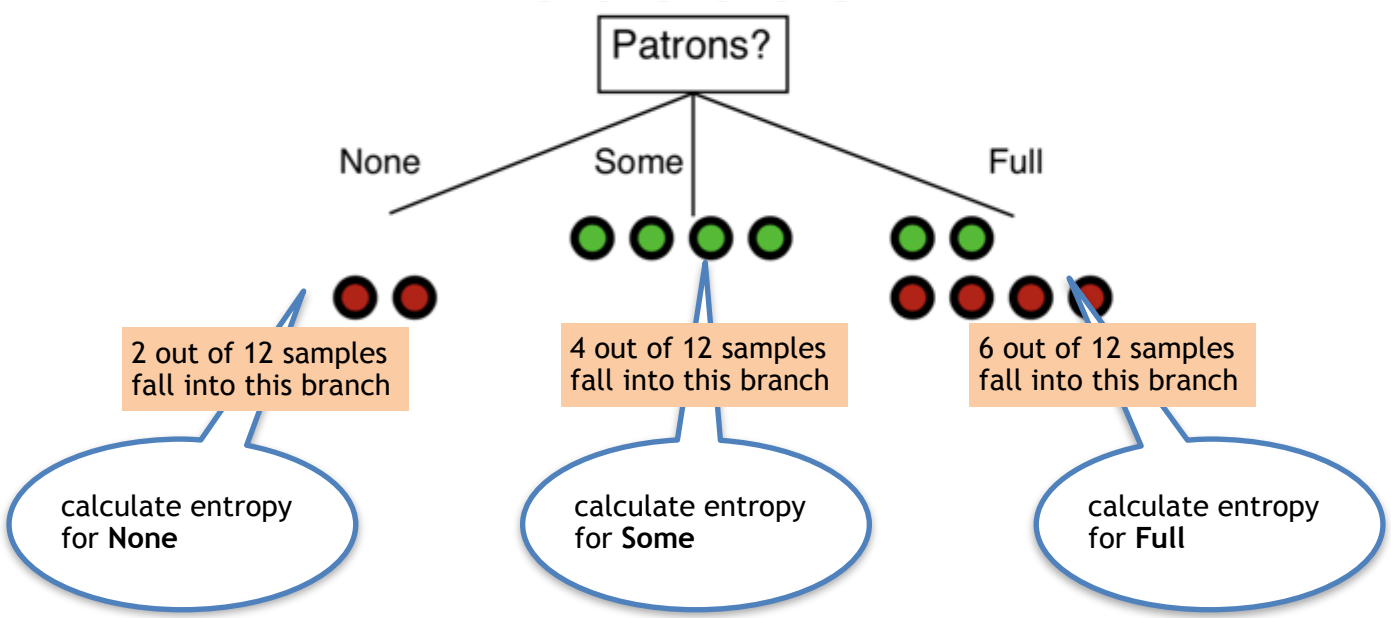


Step 2: Expected Entropy Calculation

The expected entropy for a feature is defined as the weighted sum of entropies multiplied by the fraction of samples that belong to each set.



Step 2: Expected Entropy Calculation



So, the entropy for the three sets after sorting according to *Patrons* is

<green, red> == <0.0, 1.0>

$$-\frac{0}{2}\log_2\frac{0}{2}-\frac{2}{2}\log_2\frac{2}{2}=0,$$

calculate entropy for **None**

<green, red> == <0.0, 1.0>

$$-\frac{4}{4}\log_2\frac{4}{4}-\frac{0}{2}\log_2\frac{0}{2}=0,$$

calculate entropy for **Some**

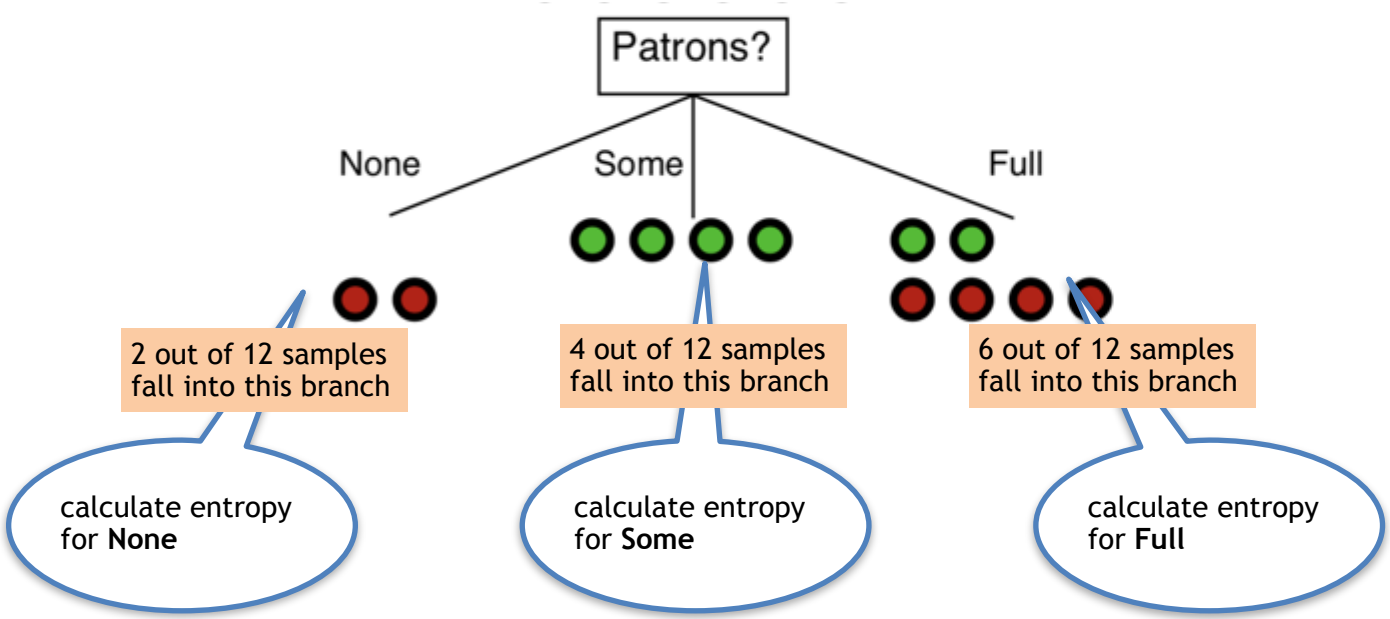
<green, red> == <0.33, 0.67>

$$\text{and } -\frac{2}{6}\log_2\frac{2}{6}-\frac{4}{6}\log_2\frac{4}{6}\approx 0.918$$

calculate entropy for **Full**

Step 2: Expected Entropy Calculation

The expected entropy for a feature is defined as the weighted sum of entropies multiplied by the fraction of samples that belong to each set.



The expected entropy remaining after testing the Patrons attribute is:

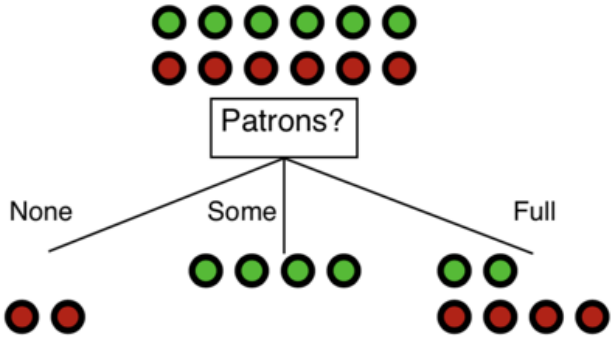
$$\approx \frac{2}{12} \cdot 0 + \frac{4}{12} \cdot 0 + \frac{6}{12} \cdot 0.918 \approx 0.459$$

Step 3: Information Gain Calculation

- The *difference* between the entropy before the test and the expected entropy after the test is the **information gain**

  InformationGain() = Entropy (before) - Expected Entropy (after)

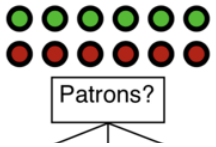
  InformationGain(Patrons) = 1.0 - 0.459 = 0.541



Summarizing all three steps!

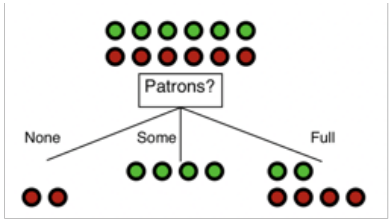
- **Step 1:** Calculate the entropy of the distribution of the classes before the node you are testing. This is the **entropy before**

Entropy_before(Patron) = 1



- **Step 2:** Calculate the **expected entropy**
  - The weighted sum of the entropy of each split of the data

Expected_entropy(Patron) = 0.459



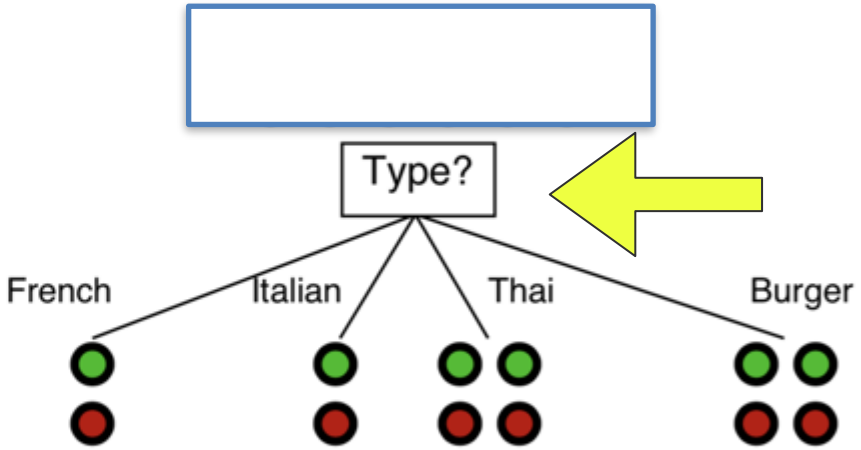
- **Step 3:** Find the difference between the **entropy before** and **expected entropy**
  - **Information Gain(Patron) = Entropy_before(Patron) - Expected_entropy(Patron)**

= 1 - 0.459

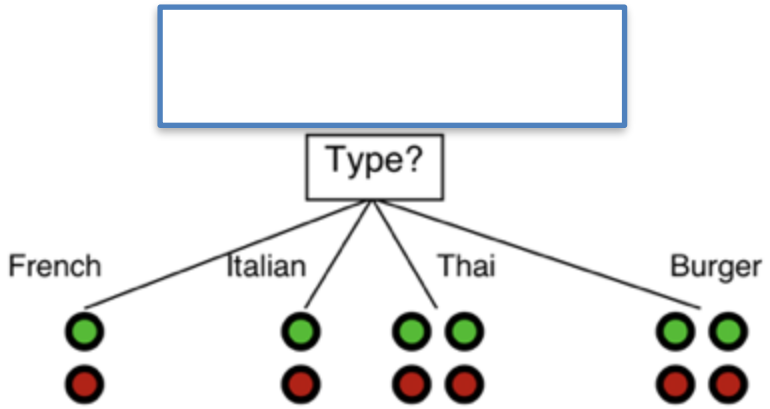
= 0.541

Another Example: Information Gain

- Calculate entropy for feature *Type* as a candidate



Another Example: Information Gain



Note that the expected entropy for the *Type* feature is

$$\frac{2}{12} \cdot Entropy\left(\left\langle \frac{1}{2}, \frac{1}{2} \right\rangle\right) + \frac{2}{12} \cdot Entropy\left(\left\langle \frac{1}{2}, \frac{1}{2} \right\rangle\right)$$

$$+\frac{4}{12} \cdot Entropy\left(\left\langle \frac{2}{4}, \frac{2}{4} \right\rangle\right) + \frac{4}{12} \cdot Entropy\left(\left\langle \frac{2}{4}, \frac{2}{4} \right\rangle\right)$$

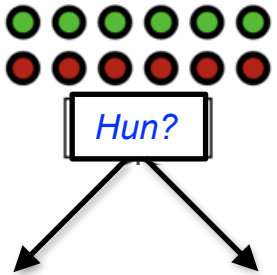
$$= \frac{2}{12} \cdot 1 + \frac{2}{12} \cdot 1 + \frac{4}{12} \cdot 1 + \frac{4}{12} \cdot 1 = 1$$

So,

$$Gain(Type) = 1 - 1 = 0$$

Decision Tree: Exercise Information Gain

- Calculate the Information Gain for *Hun*:

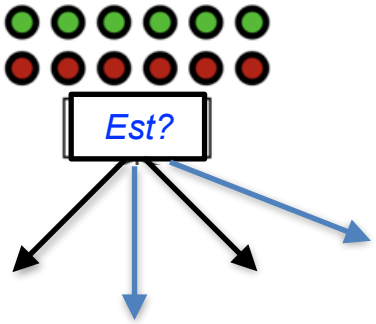


Target feature

Ex	*Alt*	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	Wait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Decision Tree: Exercise Information Gain

- Calculate the Information Gain for *Hun*:



Target feature

Ex	*Alt*	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	Wait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Decision Tree: Exercise Information Gain

- Information Gain results
  - Gain(Alt) = 1 - 1 = 0
  - Gain (Bar) = 1 - 1 = 0
  - Gain(Fri) = 1 - 0.979 = 0.021
  - Gain(Hun) = 1 - 0.804 = 0.196
  - **Gain(Patrons) = 1 - 0.459 = 0.541**
  - Gain(Price) = 1 - 0.804 = 0.196
  - Gain(Rain) = 1 - 1 = 0
  - Gain(Res) = 1 - 0.979 = 0.021
  - Gain(Type) = 1 - 1 = 0
  - Gain(Est) = 1 - 0.792 = 0.208