Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering

Artificial intelligence – VIMIAC16-EN, VIMIAC10

2024 Fall Semester

Dr. Gábor Hullám

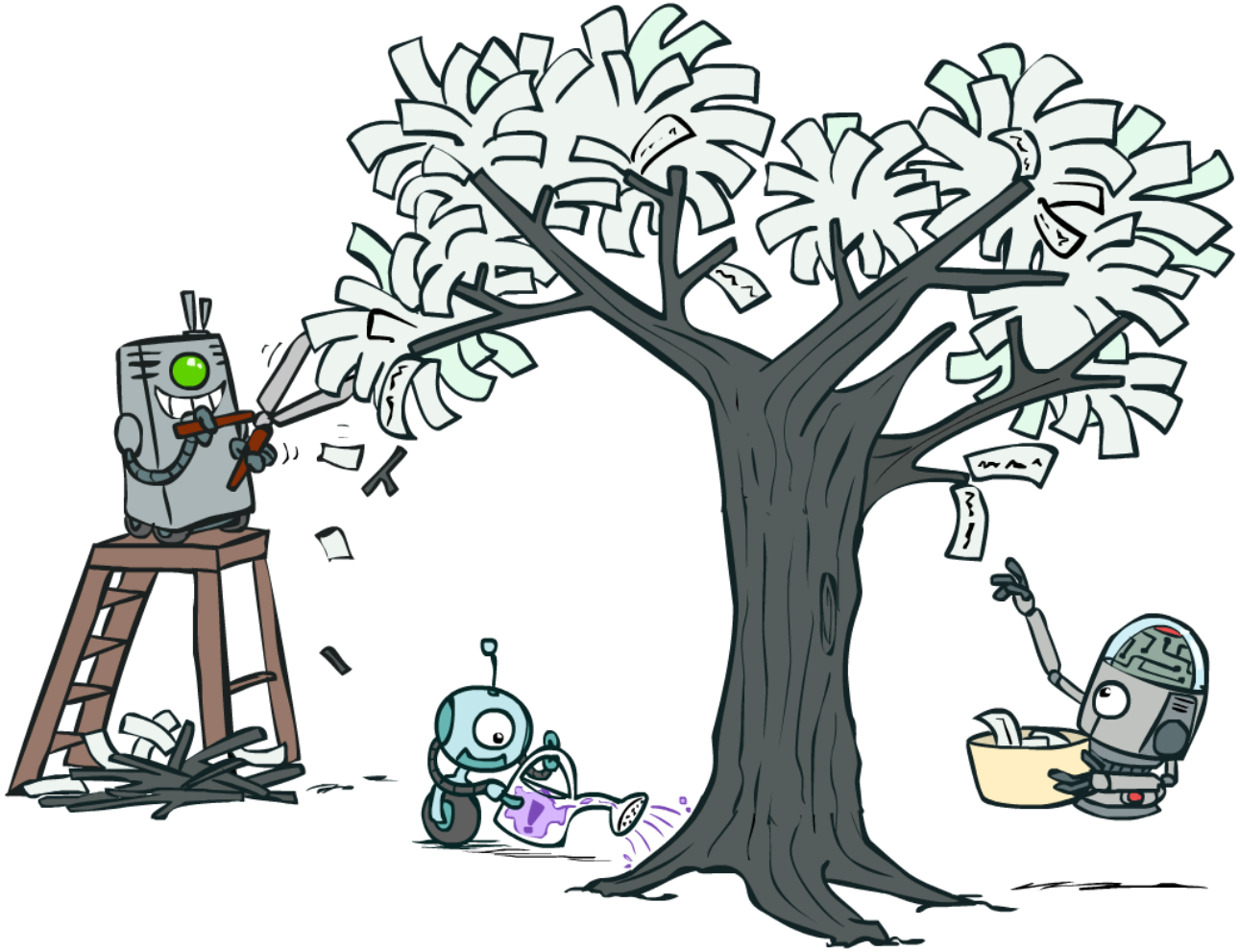
Slides Adapted from Berkeley CS188, from Dan Klein, Pieter Abbeel and Sergey Levine
http://ai.berkeley.edu

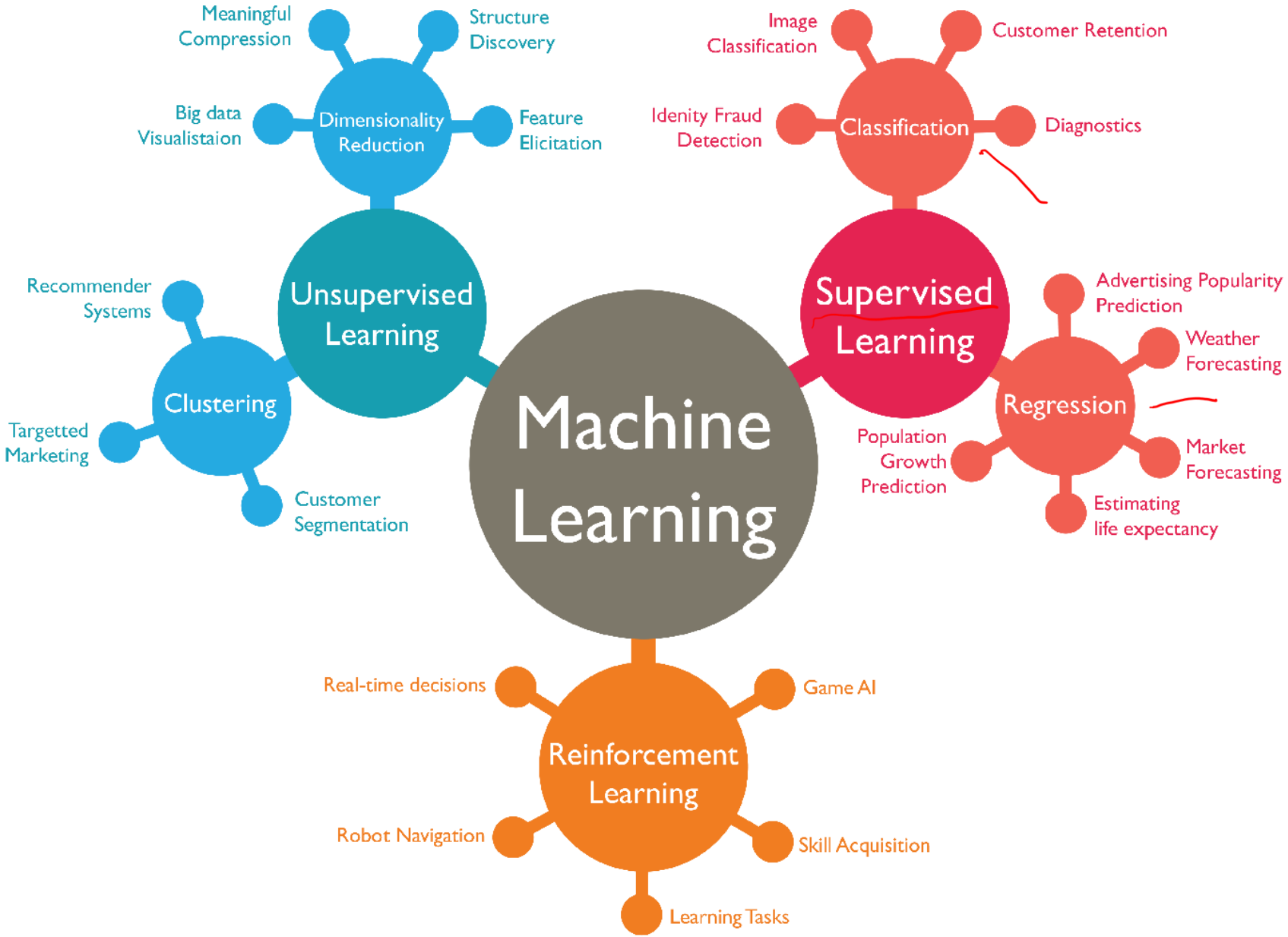
Artificial intelligence lectures

Az előadás diái az AIMA könyvre épülve (http://aima.cs.berkeley.edu) készültek a University of California, Berkeley mesterséges intelligencia kurzusának anyagainak felhasználásával (http://ai.berkeley.edu).

These slides are based on the AIMA book (http://aima.cs.berkeley.edu) and were adapted from the AI course material of University of California, Berkeley (http://ai.berkeley.edu).

Decision Trees





Wahid, Abdul. “Big Data and Machine Learning for Businesses.” LinkedIn SlideShare, April 26, 2017. https://www.slideshare.net/awahid/big-data-and-machine-learning-for-businesses.

Main types of machine learning

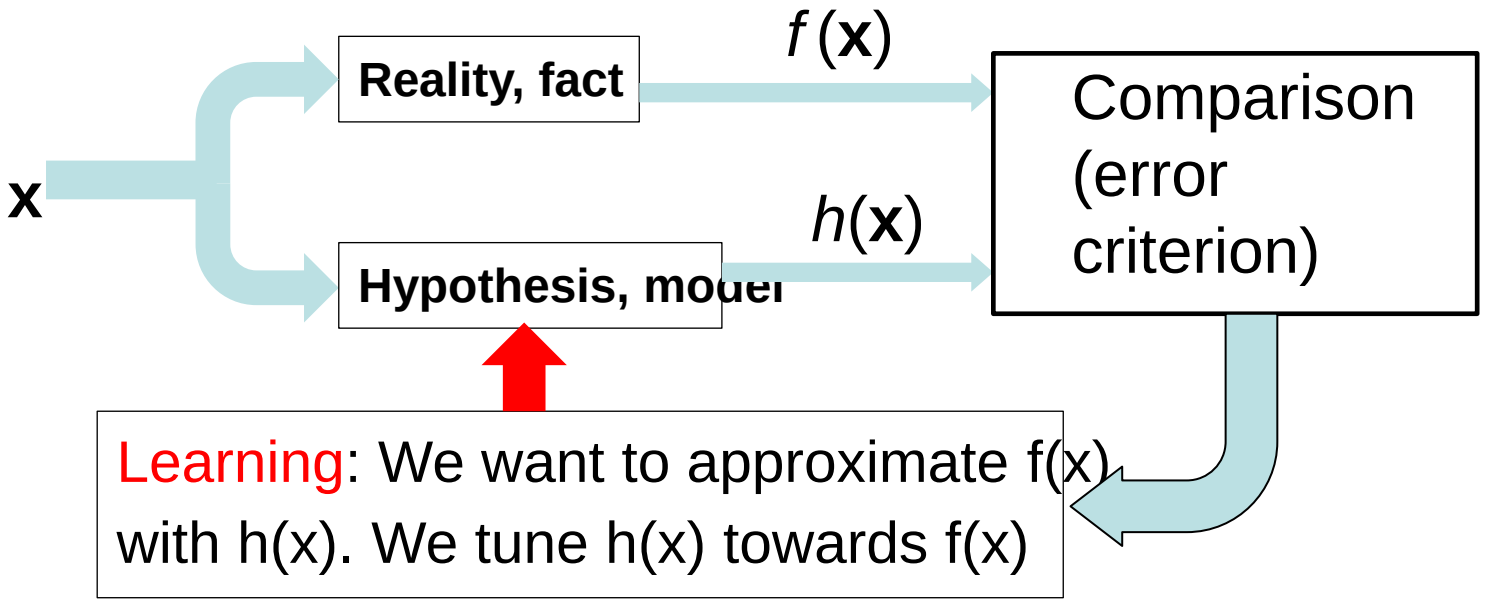
- **supervised learning:** we have both the input and the output (input sample + desired response)
- **reinforcement learning:** The agent receives a certain evaluation of its activity, perhaps not at every step (reward, reinforcement)
- **Unsupervised learning:** no information available at our disposal about the correct output

Learning based on samples (observations)

- In the following, we examine methods for developing a decision-making system based on patterns and examples.
- Often we have a bunch of samples, and that carries the information.
- We don't have pre-established rules for the task, just our samples.
- Of course, in this case we try to use the knowledge carried by the patterns in some structure.

Learning a hypothesis/model

- Learning example: (x, f(x)) Data pair, where f(.) is unknown
- Purpose of learning: meaningful approximation of f(x) by an h(x) hypothesis
- h(x) = f(x), x – on known examples, often complete accuracy
- h(x')~= f(x'), x' – not seen while learning (f(x) generalizing ability)



Features

- Features, aka attributes
  - Sometimes: TYPE=French
  - Sometimes: $f_{\text{TYPE=French}}(x) = 1$

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
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*

Example problem: deciding whether to wait for a table in a particular restaurant?

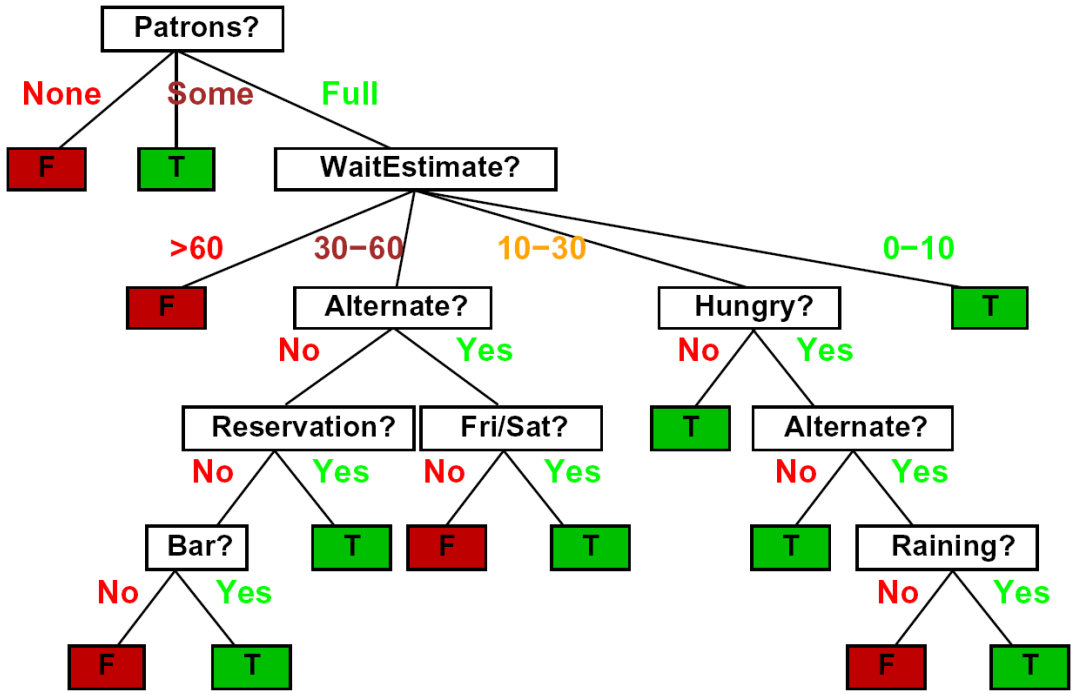
Purpose: to produce a logical function of decision

The problem is characterized by **properties** or attributes.

1. *Alternative*: Is there any other suitable restaurant in the area. (Y/N)
2. *Bar*: is there a comfortable bar in the restaurant where we can wait. (Y/N)
3. *Fri/Sat*: true on Fridays and Saturdays. (Y/N)
4. *Hungry*: whether we are hungry. (Y/N)
5. *Client*: how many are already in (*Nobody, Few and Full*).
6. *Price*: how expensive it is (*cheap, medium, expensive)*
7. *Rain*: whether it rains outside (Y/N)
8. *Reservation*: whether a table must be booked in advance (Y/N)
9. *Type*: Type of restaurant (*French, Italian, Thai v. Burger*)
10. *ApproxTime*: waiting time announced by the waiter
    (0-10 mins, 10-30, 30-60, >60 mins)

Decision Trees

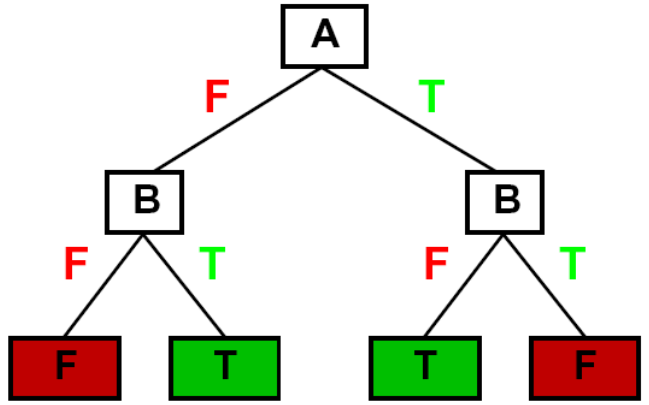
- Compact representation of a function:
  - Truth table
  - Conditional probability table
  - Regression values
- True function
  - Realizable: in H



Expressiveness of DTs

- Can express any* logic function of the features

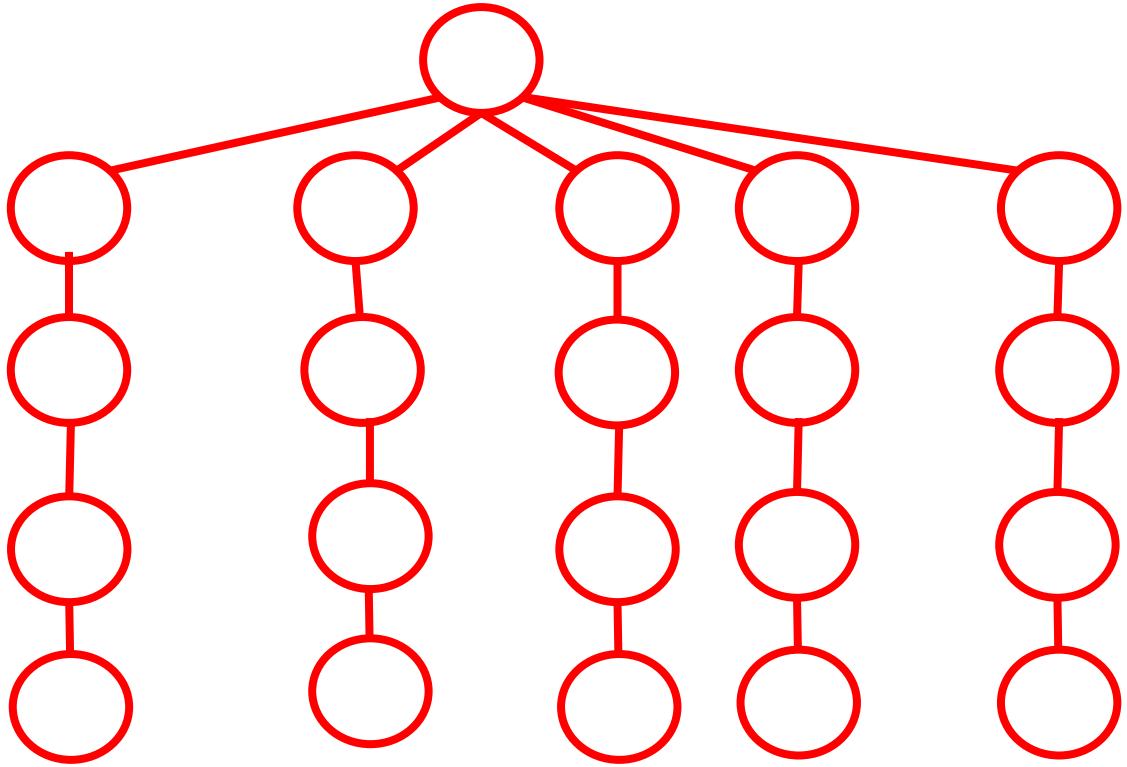
A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



- However, we hope for compact trees

trivial tree, light - for each example there is an independent path
The leaf gives a classification of the example.
The tree simply memorizes the example, does not form characteristic patterns
does not generalize from examples (cannot extrapolate to unknown examples))

"real" tree - extraction of characteristic patterns, the method is able to describe a large number of cases in a concise form, and thus generalize to examples that are not yet unknown



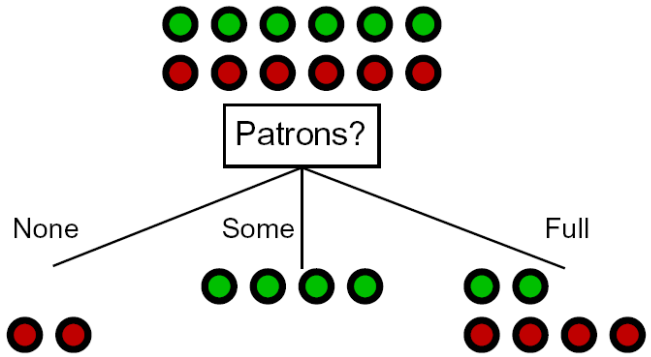
Decision Tree Learning

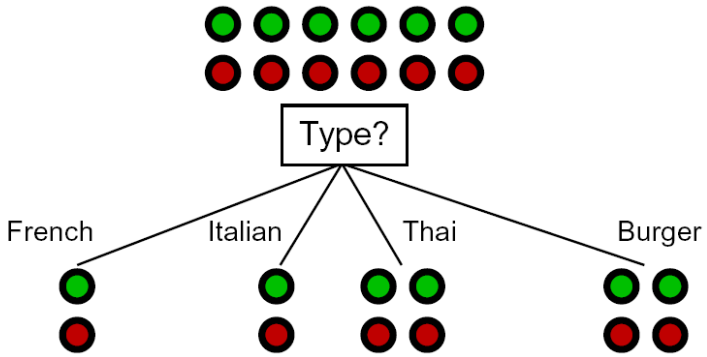
- Aim: find a small tree consistent with the training examples
- Idea: (recursively) choose "most significant" attribute as root of (sub)tree

```
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”

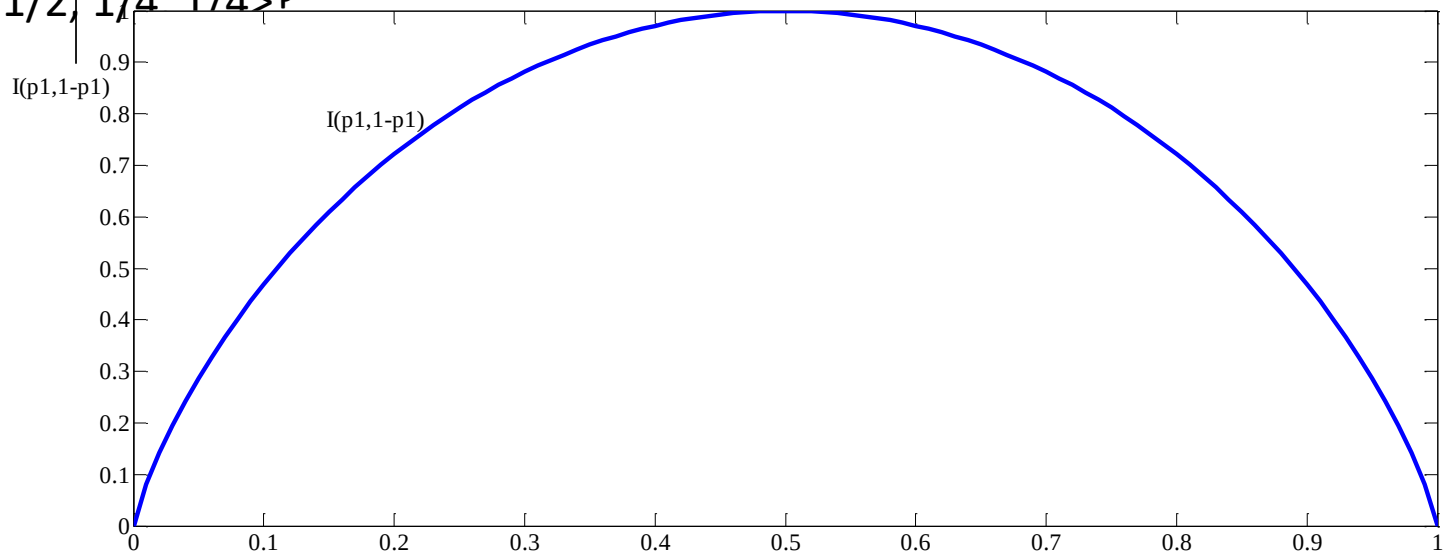




- So: we need a measure of how “good” a split is, even if the results aren’t perfectly separated out

Entropy and Information

- Information answers questions
  - The more uncertain about the answer initially, the more information in the answer
  - Scale: bits
    - Answer to Boolean question with prior <1/2, 1/2>?
    - Answer to 4-way question with prior <1/4, 1/4, 1/4, 1/4>?
    - Answer to 4-way question with prior <0, 0, 0, 1>?
    - Answer to 3-way question with prior <1/2, 1/4, 1/4>?



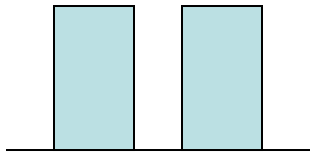
Entropy

- General answer: if prior is $\langle p_1, \ldots, p_n \rangle$:
  - Information is the expected code length

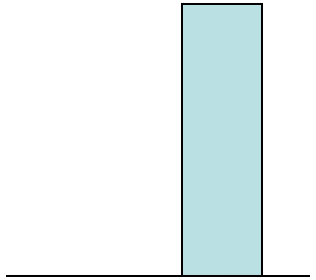
$$H(\langle p_1, \ldots, p_n \rangle) = E_p \log_2 1/p_i$$

$$= \sum_{i=1}^{n} -p_i \log_2 p_i$$

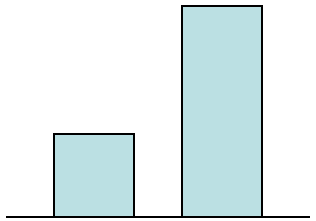
- Also called the entropy of the distribution
  - More uniform = higher entropy
  - More values = higher entropy
  - More peaked = lower entropy



1 bit



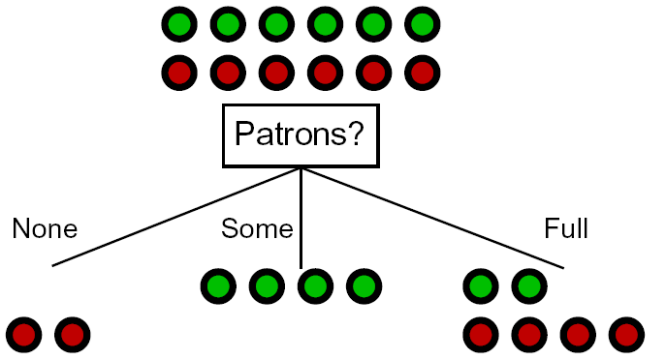
0 bits

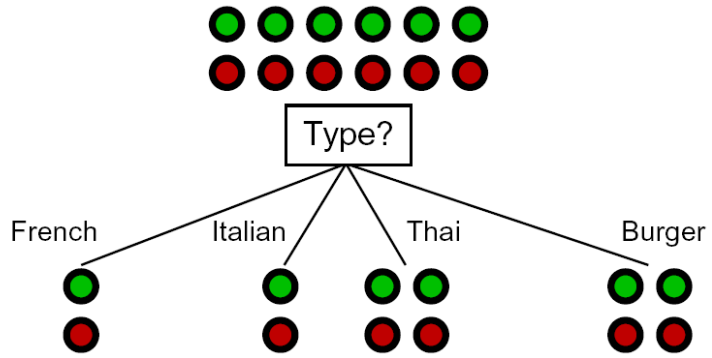


0.5 bit

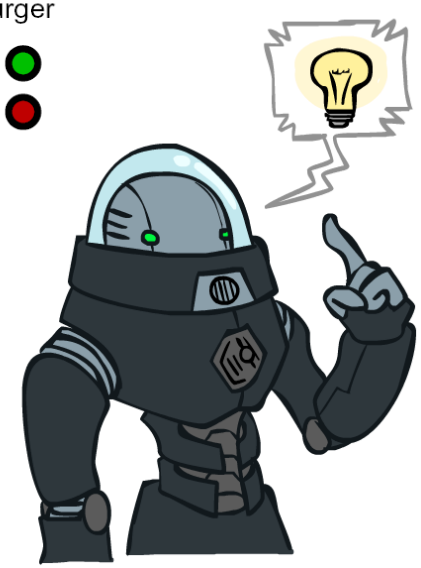
Information Gain

- Back to decision trees!
- For each split, compare entropy before and after
  - Difference is the information gain
  - Problem: there's more than one distribution after split!



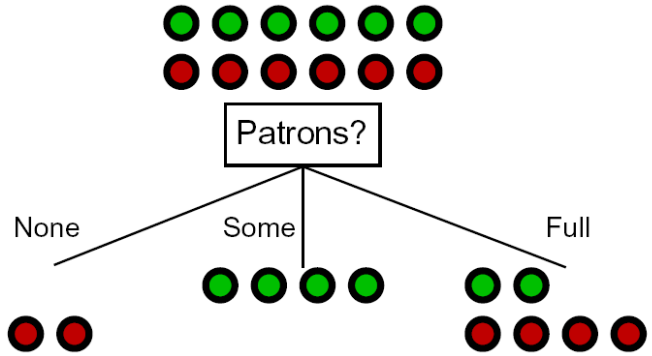


  - Solution: use expected entropy, weighted by the number of examples



Next Step: Recurse

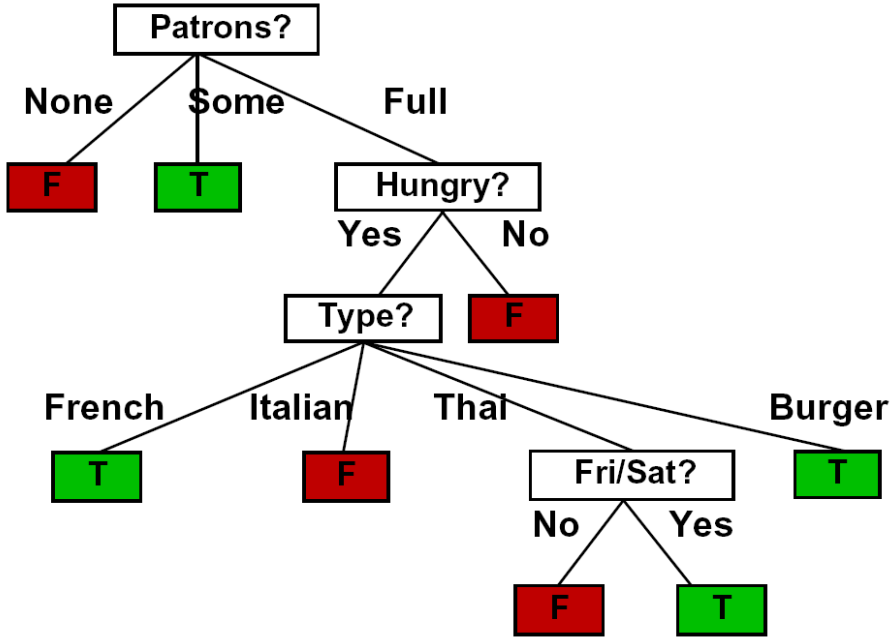
- Now we need to keep growing the tree!
- Two branches are done (why?)
- What to do under “full”?
  - See what examples are there…



Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
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*

Example: Learned Tree

- Decision tree learned from these 12 examples:



- Substantially simpler than “true” tree
  - A more complex hypothesis isn't justified by data
- Also: it’s reasonable, but wrong

Information content of the decision tree

The training set consists of ***p*** positive and ***n*** negative examples.
The two answers:
V1 (positive), V2 (negative), and estimation of their probability:

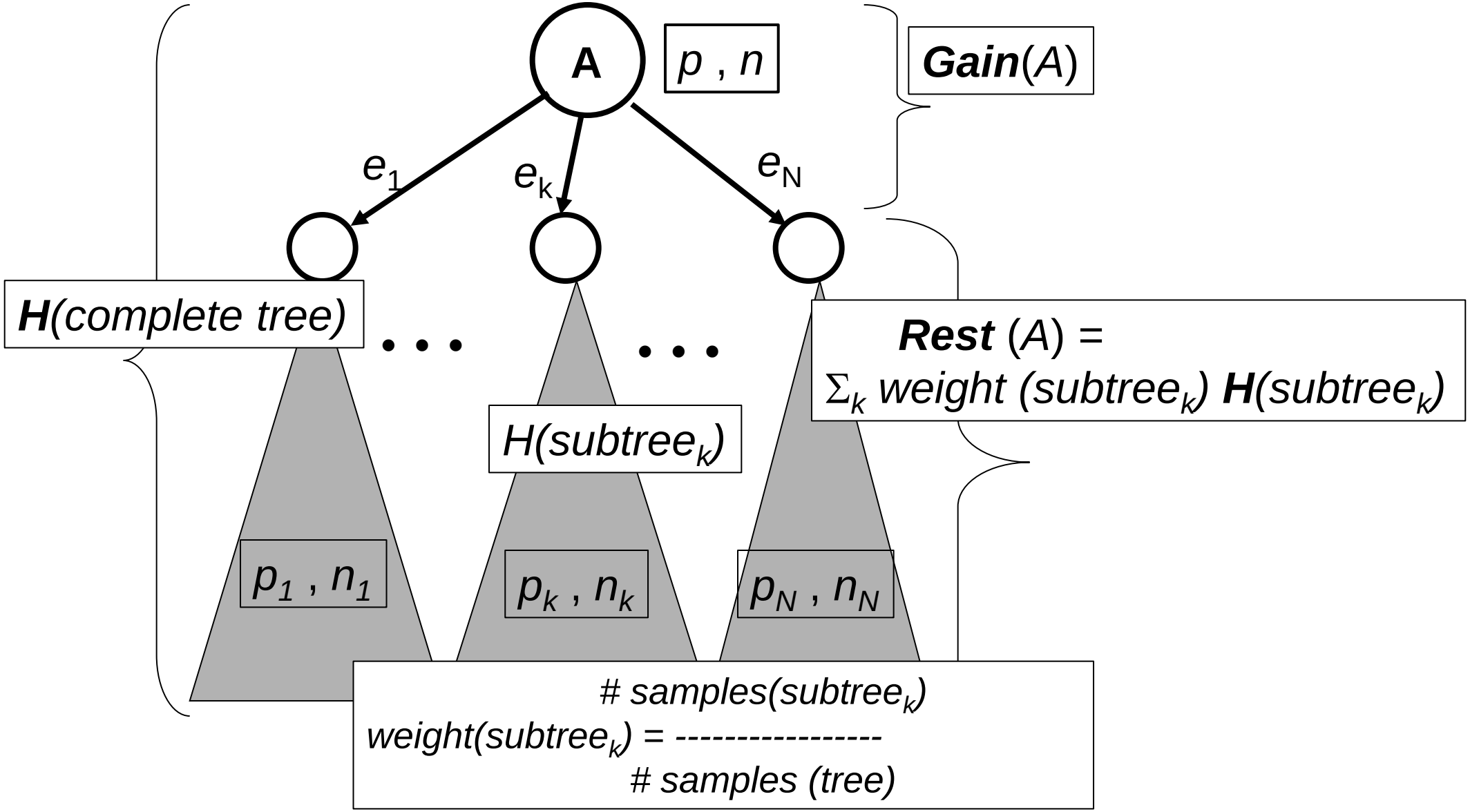
$$P(v_1) \cong \frac{p}{p+n}, \qquad P(v_2) \cong \frac{n}{p+n}$$

- Then the information content of the tree is estimated as :

$$H(P(v_1), P(v_2)) \cong H\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n} \cdot \log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n} \cdot \log_2\left(\frac{n}{p+n}\right)$$

- How to choose which attribute to test?
- How much new information does testing an attribute give us?
- How much information did we need before testing the attribute, and how much more do we need after testing the attribute?

Any attribute A causes the training set E to be subdivided into E1,E2,...,EN. according to the answers to A's test, if A's test gives N different answers.



How to choose an attribute?

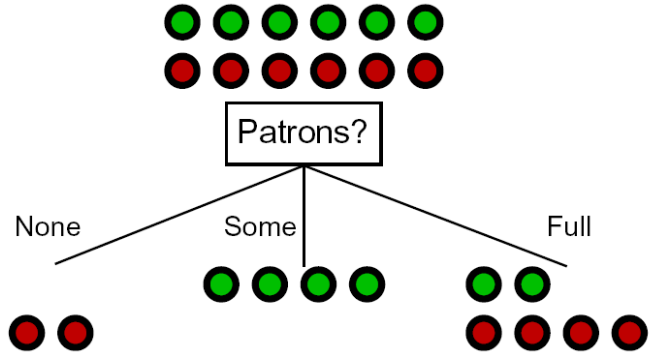
$$Rest(A) = \sum_{k=1}^{N} \frac{p_k + n_k}{p + n} \cdot H\left(\frac{p_k}{p_k + n_k}, \frac{n_k}{p_k + n_k}\right)$$

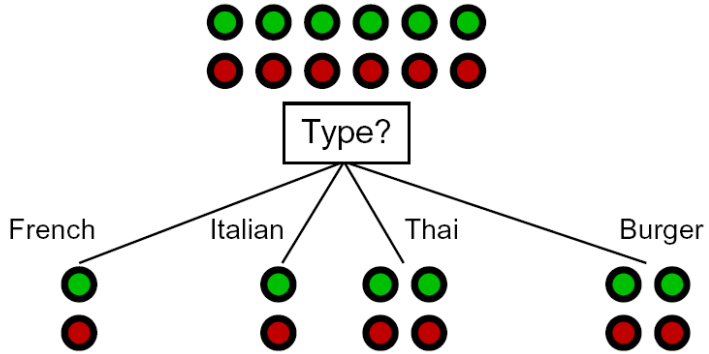
- The information gain of the attribute test is the difference between the information requirement before the test and after the test:

$$GAIN(A) = H\left(\frac{p}{p + n}, \frac{n}{p + n}\right) - Rest(A)$$

- Choose the attribute that gives you the largest gain at the moment!
  (Greedy procedure)

Let's look at the „Patrons” and „Type” attributes





$$GAIN(Patrons) = 1 - \left[\frac{2}{12} \cdot H(0,1) + \frac{4}{12} \cdot H(1,0) + \frac{6}{12} \cdot H(\frac{2}{6}, \frac{4}{6})\right] \approx 0{,}5$$

$$GAIN\ (Types) = 1 - \left[\frac{2}{12} \cdot H(\frac{1}{2}, \frac{1}{2}) + \frac{2}{12} \cdot H(\frac{1}{2}, \frac{1}{2}) + \frac{4}{12} \cdot H(\frac{2}{4}, \frac{2}{4}) + \frac{4}{12} \cdot H(\frac{2}{4}, \frac{2}{4})\right] = 0$$

Usually it takes several steps to calculate the information gain of all the attributes.

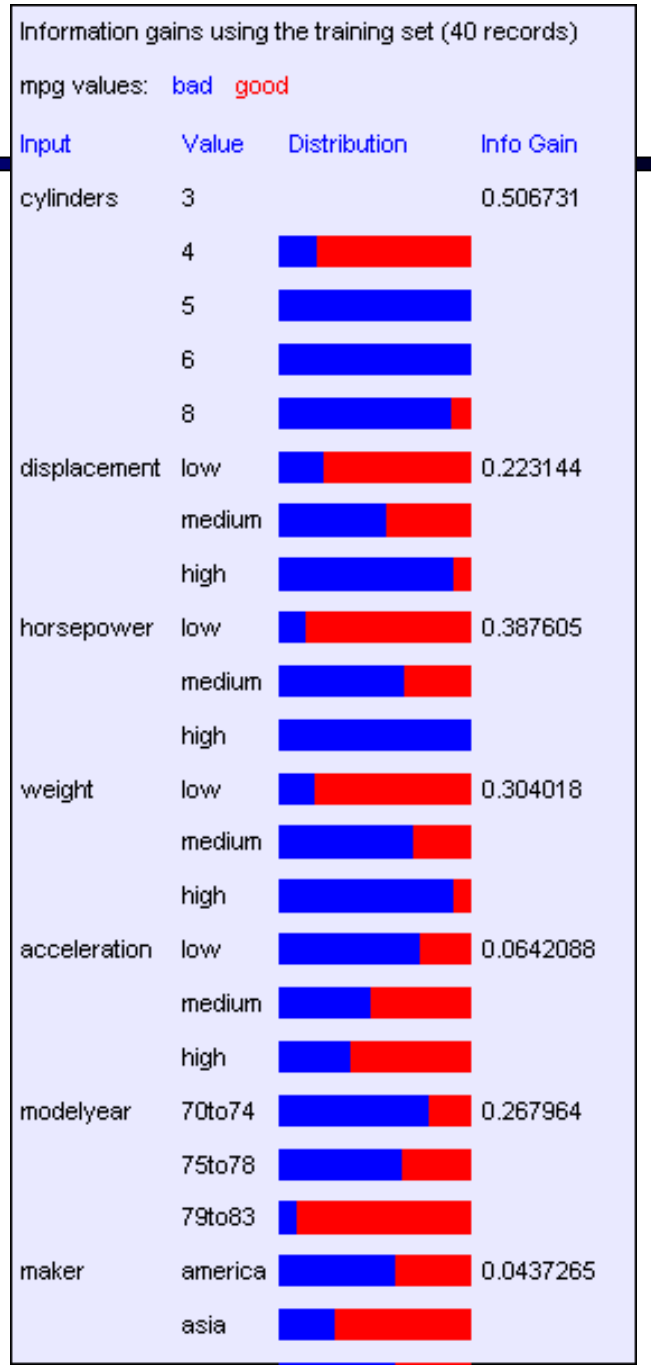
Example: Miles Per Gallon

40 Examples

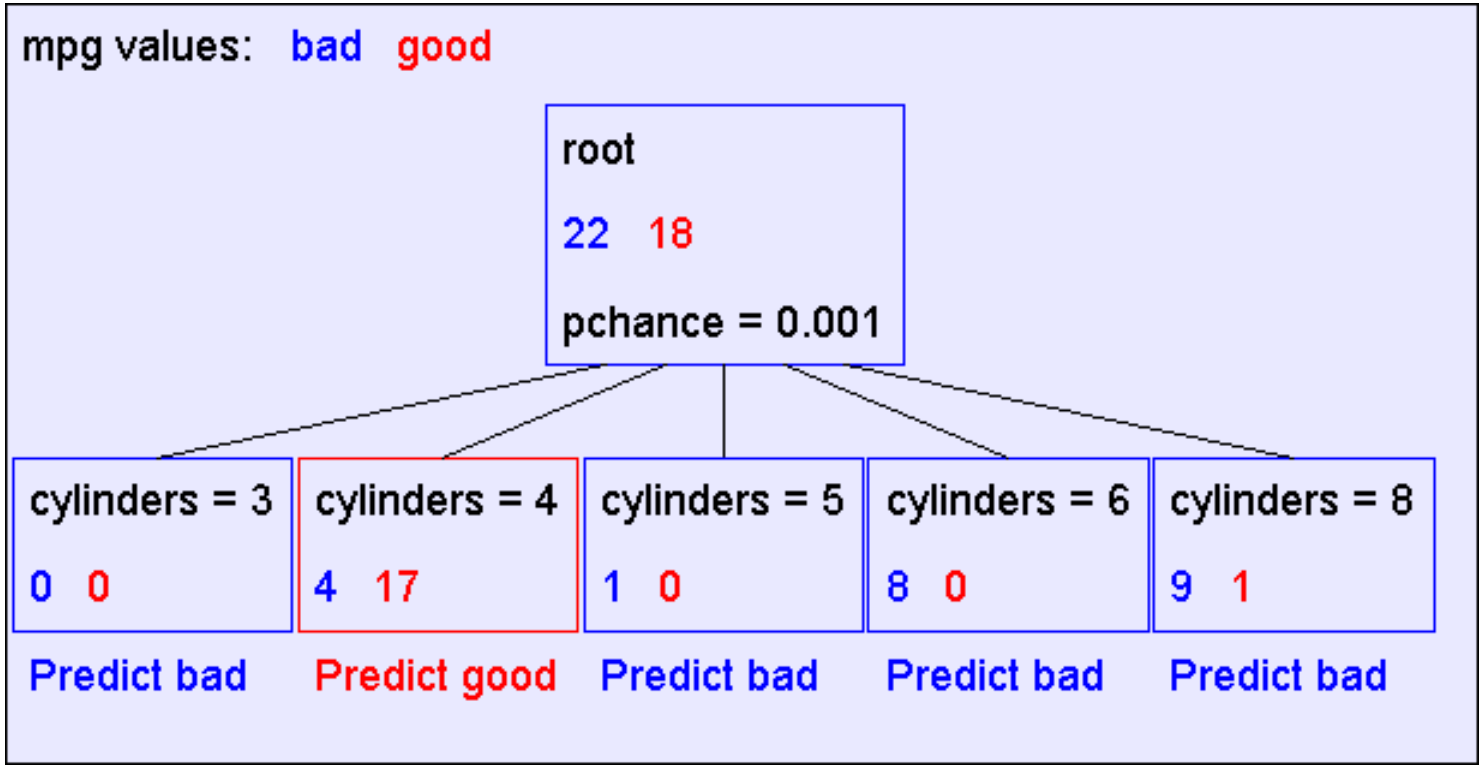
mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low	medium	low	medium	70to74	asia
bad	4	low	medium	low	low	70to74	asia
bad	8	high	high	high	low	75to78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
bad	8	high	high	high	low	70to74	america
good	8	high	medium	high	high	79to83	america
bad	8	high	high	high	low	75to78	america
good	4	low	low	low	low	79to83	america
bad	6	medium	medium	medium	high	75to78	america
good	4	medium	low	low	low	79to83	america
good	4	low	low	medium	high	79to83	america
bad	8	high	high	high	low	70to74	america
good	4	low	medium	low	medium	75to78	europe
bad	5	medium	medium	medium	medium	75to78	europe

Find the First Split

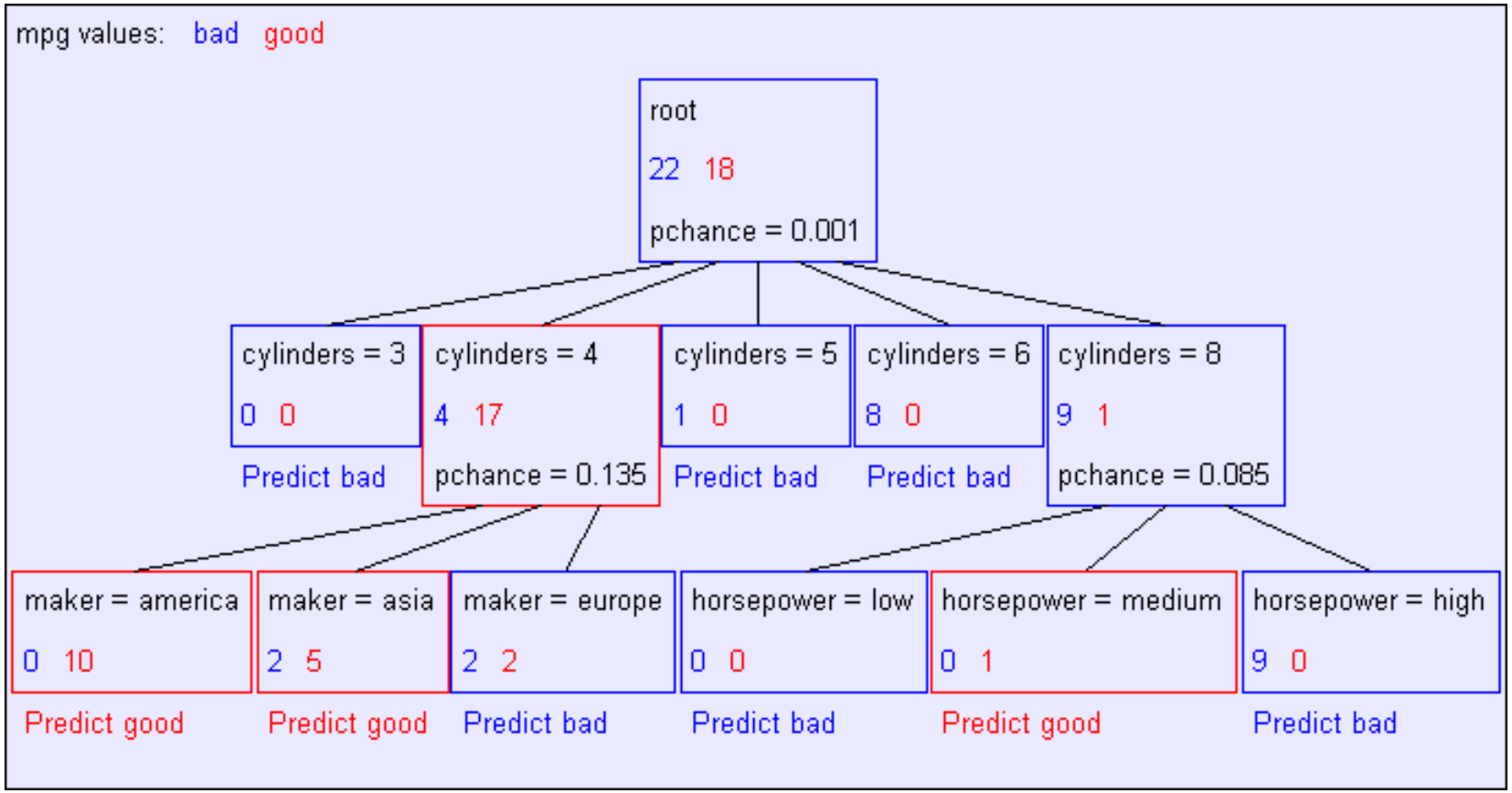
- Look at information gain for each attribute
- Note that each attribute is correlated with the target!
- What do we split on?



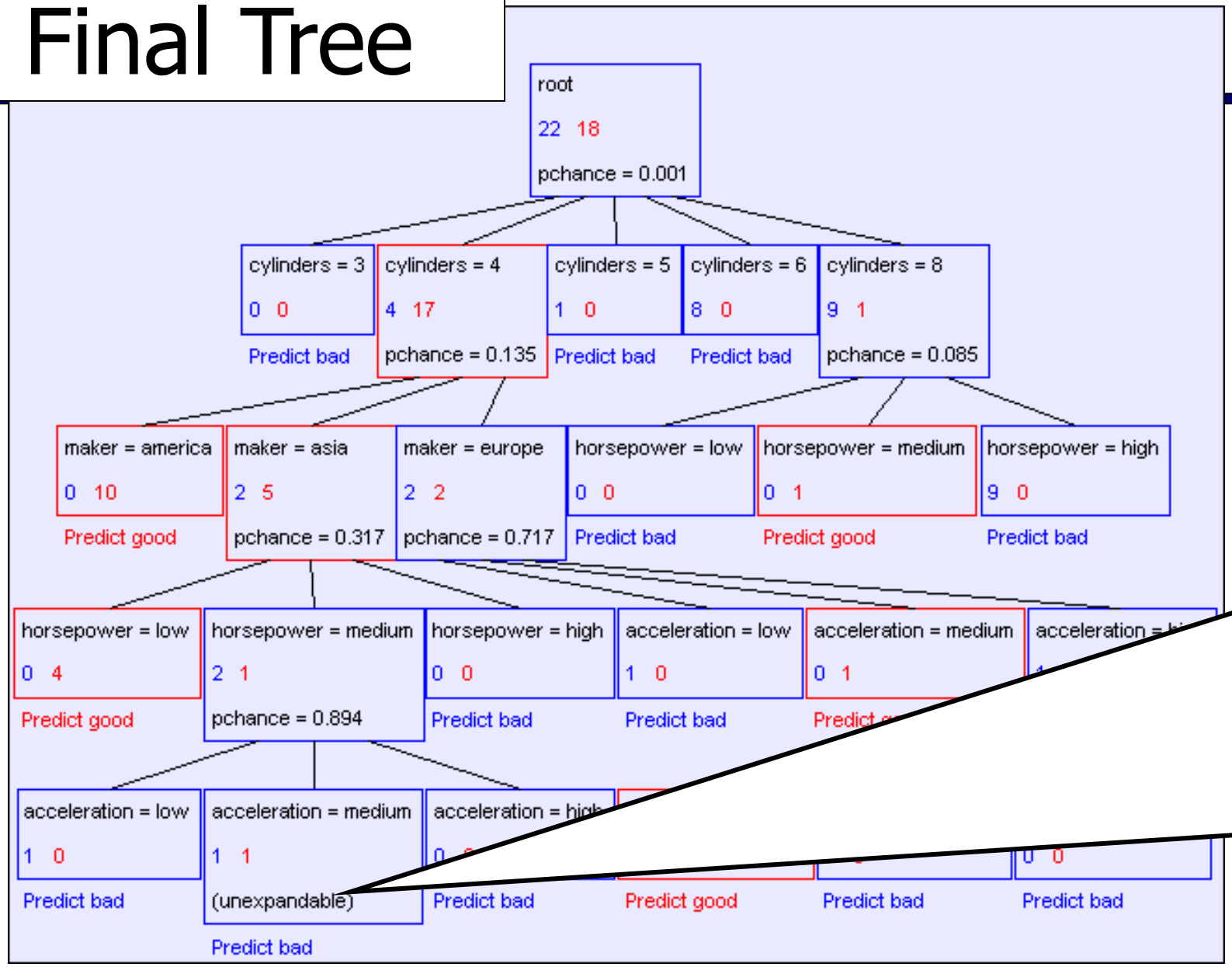
Result: Decision Stump



Second Level



Final Tree

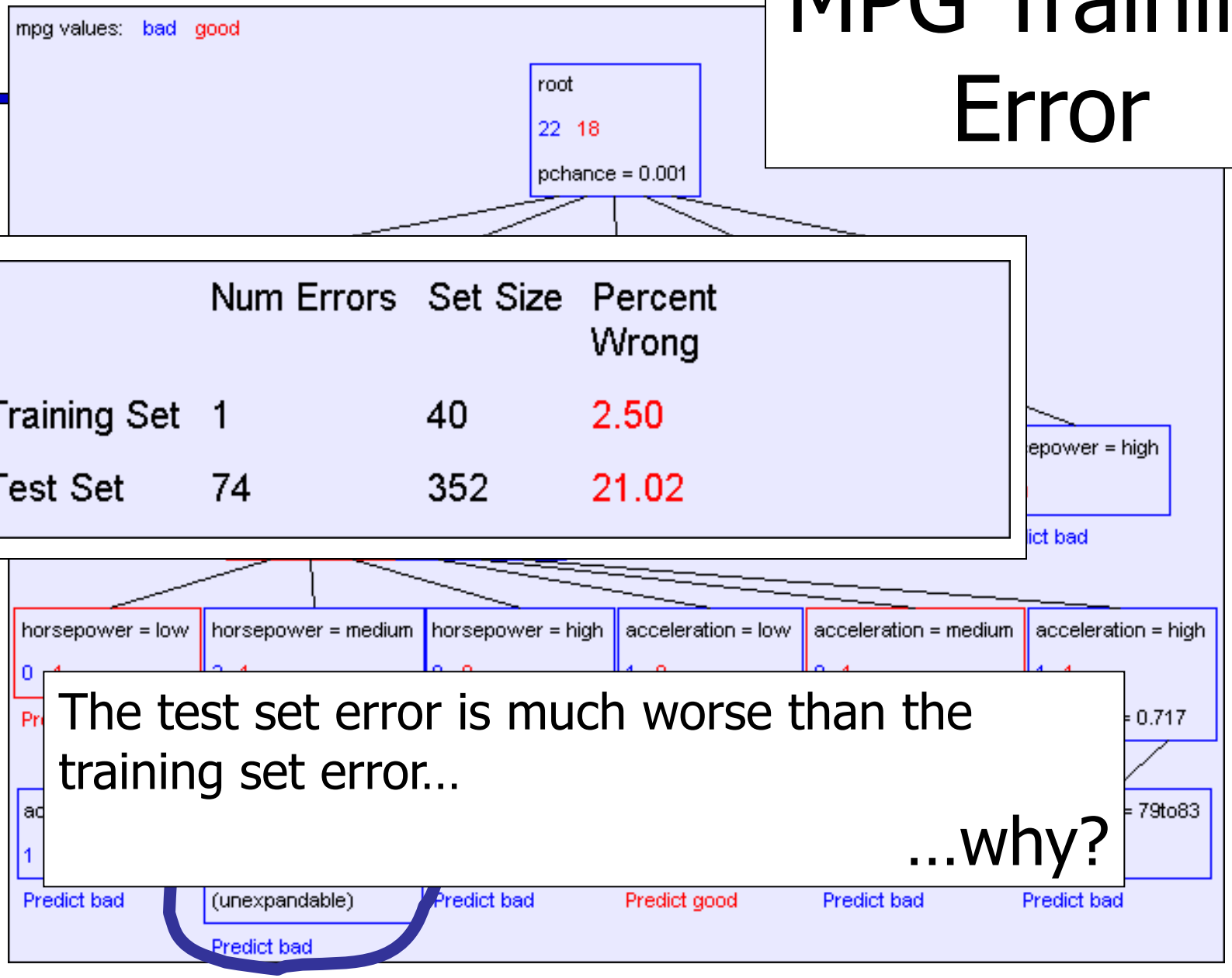


Information gains using the training set (2 records)

mpg values: bad good

Input	Value	Distribution	Info Gain
cylinders	3		0
	4		
	5		
	6		
	8		
displacement	low		0
	medium		
	high		
horsepower	low		0
	medium		
	high		
weight	low		0
	medium		
	high		
acceleration	low		0
	medium		
	high		
modelyear	70to74		0
	75to78		
	79to83		
maker	america		0
	asia		
	europe		

MPG Training Error



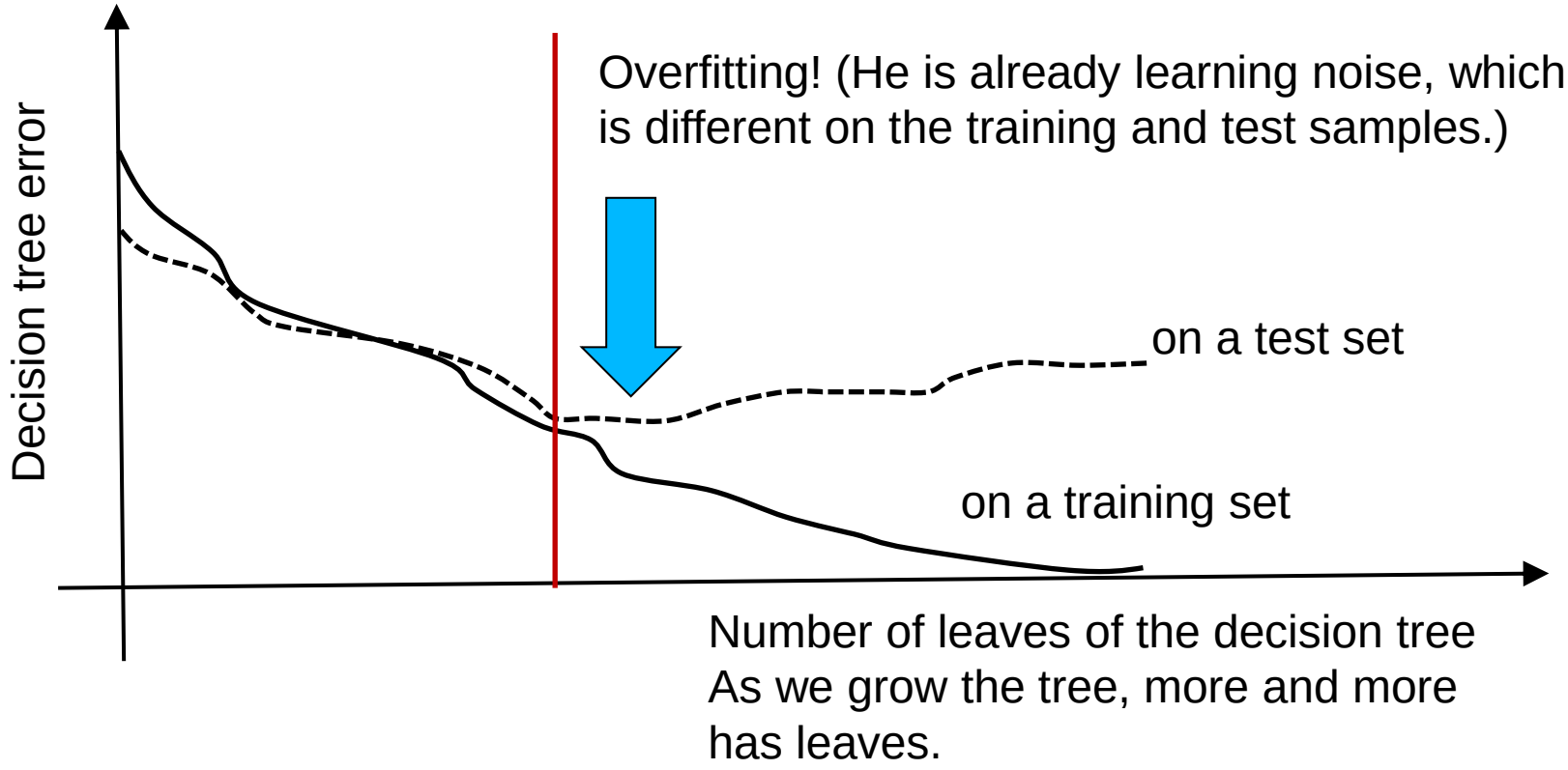
	Num Errors	Set Size	Percent Wrong
Training Set	1	40	2.50
Test Set	74	352	21.02

The test set error is much worse than the training set error…

…why?

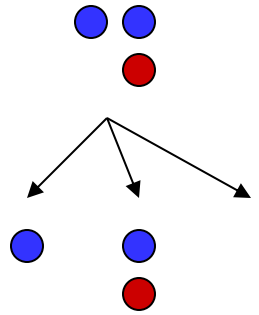
Overfitting

- Overfitting:
  - When you stop modeling the patterns in the training data (which generalize)
  - And start modeling the noise (which doesn't)



Significance of a Split

- Starting with:
  - Three cars with 4 cylinders, from Asia, with medium HP
  - 2 bad MPG
  - 1 good MPG



- What do we expect from a three-way split?
  - Maybe each example in its own subset?
  - Maybe just what we saw in the last slide?
- Probably shouldn't split if the counts are so small they could be due to chance
- A chi-squared test can tell us how likely it is that deviations from a perfect split are due to chance*
- Each split will have a significance value, p_{CHANCE}

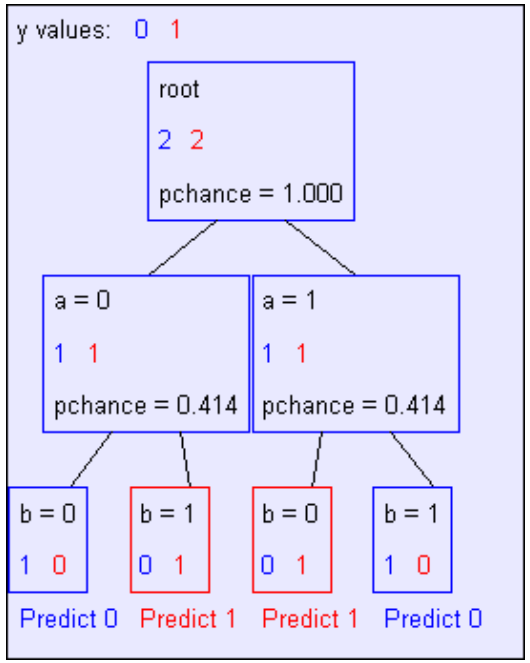
Keeping it General

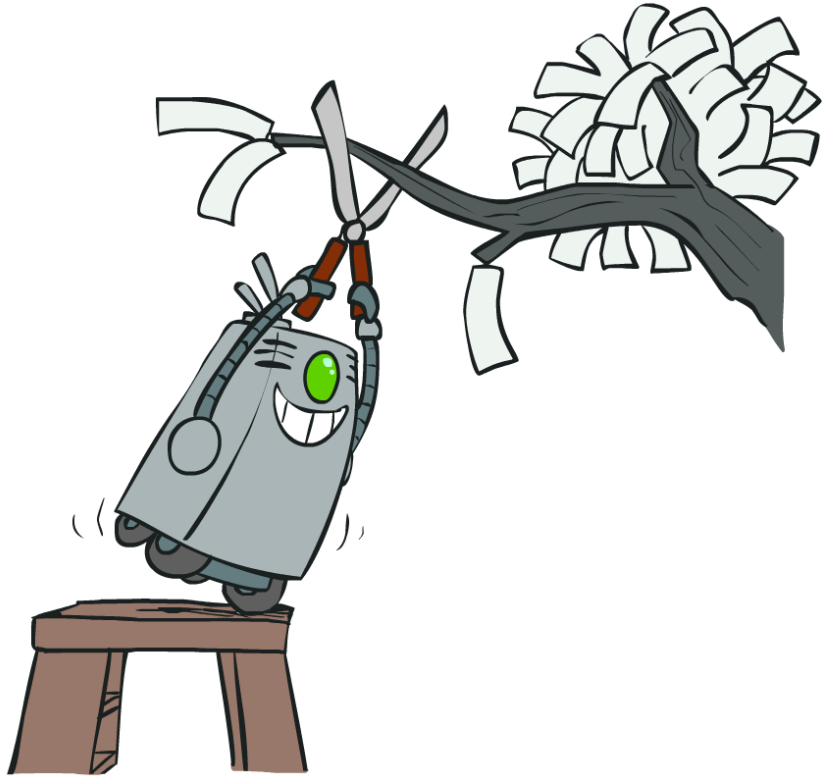
- Pruning:
  - Build the full decision tree
  - Begin at the bottom of the tree
  - Delete splits in which

    $p_{CHANCE} > MaxP_{CHANCE}$
  - Continue working upward until there are no more prunable nodes

y = a XOR b

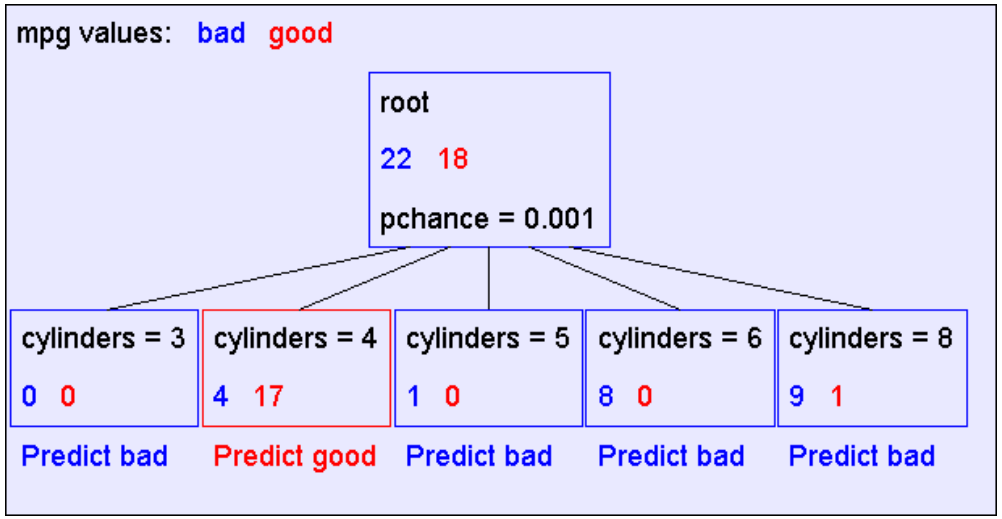
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0





Pruning example

- With $MaxP_{CHANCE} = 0.1$:



Note the improved test set accuracy compared with the unpruned tree

	Num Errors	Set Size	Percent Wrong
Training Set	5	40	12.50
Test Set	56	352	15.91

Decision tree pruning

(1) Set max. depth (as in depth limited search), if we reach it, we will definitely stop at this level, we will not develop the given branch further.

(2) Examine the tests performed, recognize the irrelevant (meaningless) attribute used, and finish the branch there (prune the tree back).

Trimming a decision tree

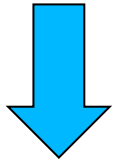
2. Let it overgrow, then cut it back, trim it back

- *First option is to recognize if an irrelevant attribute has been used, and stop elaborating on the branch and even remove the formed subtree.*

- Irrelevant attribute: when splitting an example set, the resulting subsets contain approximately the same proportion of positive and negative examples as the original set.
- The information gain is close to 0.
  - The lack of information gain is a good indicator of irrelevance.
  - How big should the information gain be to still use an attribute to share the set of examples?
  - We perform a statistical test: we assume that there is no characteristic pattern = so-called null hypothesis.

Trimming a decision tree

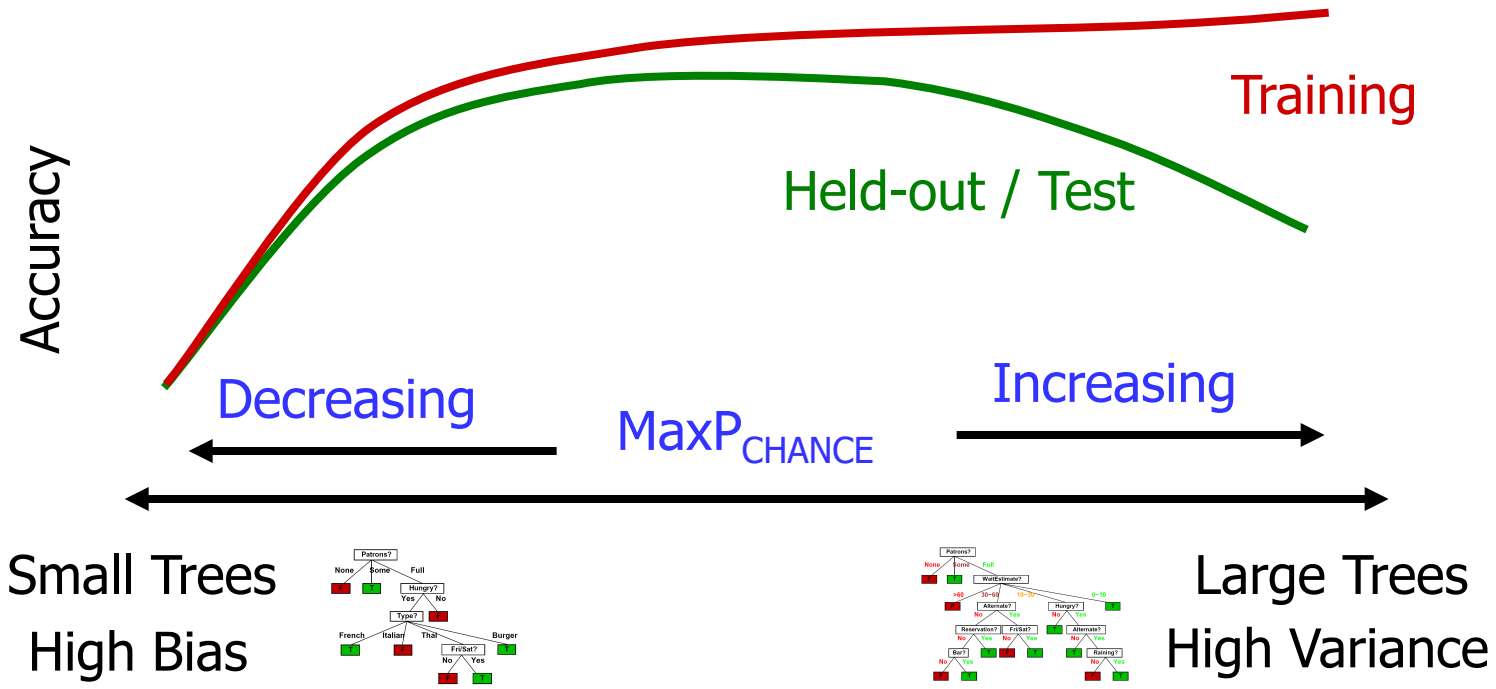
- Statistical test – we think the other way around, we assume that there is no characteristic pattern in the set = a so-called null hypothesis. Then we try to refute this – or at least admit that it is highly unlikely!



- **If the null hypothesis is true, then in principle the information gain would be 0**, Any deviation from this is only accidental.
- If the deviation is large (well greater than 0), it is unlikely that the detected deviation is due to random fluctuations
-
- The null hypothesis is unlikely: there is probably a substantial sample in the data. (We can refute the zero hypothesis with some probability.)

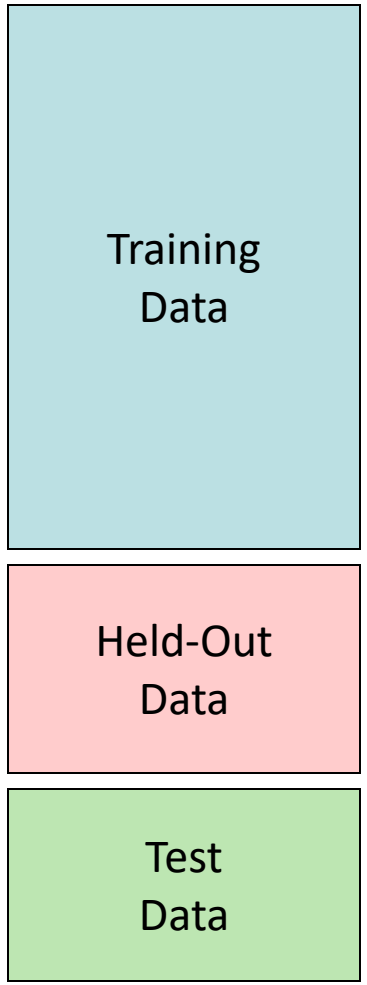
Regularization

- $MaxP_{CHANCE}$ is a regularization parameter
- Generally, set it using held-out data (as usual)



A few important points about learning

- Data: labeled instances, e.g. emails marked spam/ham
  - Training set
  - Held out set
  - Test set
- Features: attribute-value pairs which characterize each x
- Experimentation cycle
  - Learn parameters (e.g. model probabilities) on training set
  - (Tune hyperparameters on held-out set)
  - Compute accuracy of test set
  - Very important: never "peek" at the test set!
- Evaluation
  - Accuracy: fraction of instances predicted correctly
- Overfitting and generalization
  - Want a classifier which does well on *test* data
  - Overfitting: fitting the training data very closely, but not generalizing well
  - Underfitting: fits the training set poorly



A few important points about learning

- What should we learn where?
  - Learn parameters from training data
  - Tune hyperparameters on different data
    - Why?
  - For each value of the hyperparameters, train and test on the held-out data
  - Choose the best value and do a final test on the test data
- What are examples of hyperparameters?