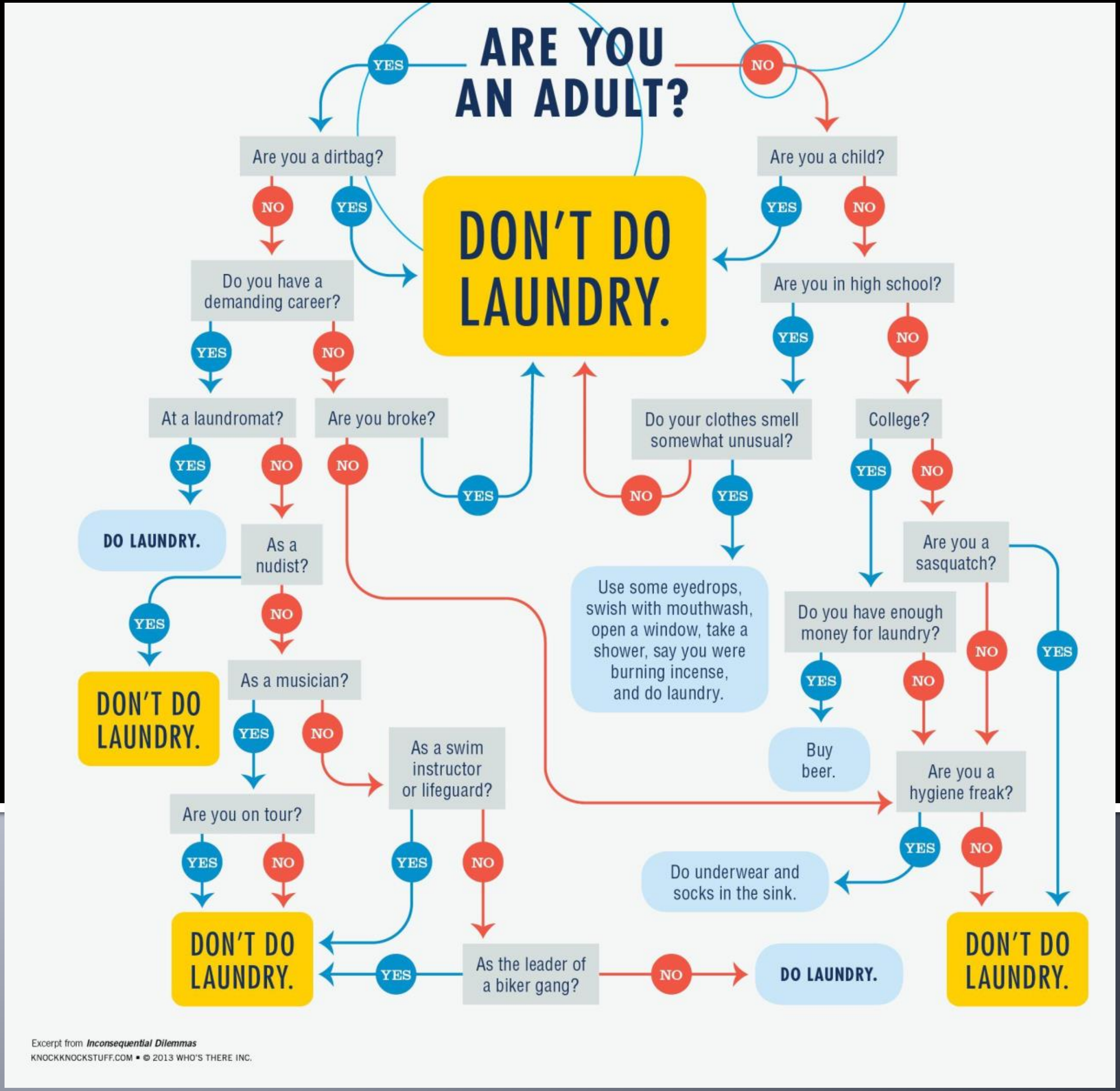
Should I do the laundry? A decision tree:


ARE YOU AN ADULT?
YES
NO
Are you a dirtbag?
Are you a child?
DON'T DO LAUNDRY.
Do you have a demanding career?
Are you in high school?
At a laundromat?
Are you broke?
Do your clothes smell somewhat unusual?
College?
DO LAUNDRY.
As a nudist?
Use some eyedrops, swish with mouthwash, open a window, take a shower, say you were burning incense, and do laundry.
Are you a sasquatch?
Do you have enough money for laundry?
As a musician?
Buy beer.
As a swim instructor or lifeguard?
Are you a hygiene freak?
Are you on tour?
Do underwear and socks in the sink.
As the leader of a biker gang?
Excerpt from Inconsequential Dilemmas
KNOCKKNOCKSTUFF.COM • © 2013 WHO'S THERE INC.

INFO 251: Applied ML

Decision Trees

Course Outline

- Causal Inference and Research Design
  - Experimental methods
  - Non-experiment methods
- **Machine Learning**
  - Design of Machine Learning Experiments
  - Linear Models and Gradient Descent
  - **Non-linear models**
  - Fairness and Bias in ML
  - Neural models
  - Deep Learning
  - Practicalities
  - Unsupervised Learning
- Special topics

Last Lecture: Key Concepts

- Bayes' theorem
- Prior probability
- Conditional probability
- Posterior probability
- Log-Likelihood
- Spam classification
- Laplace smoothing

Sample quiz question: Bayes Rule

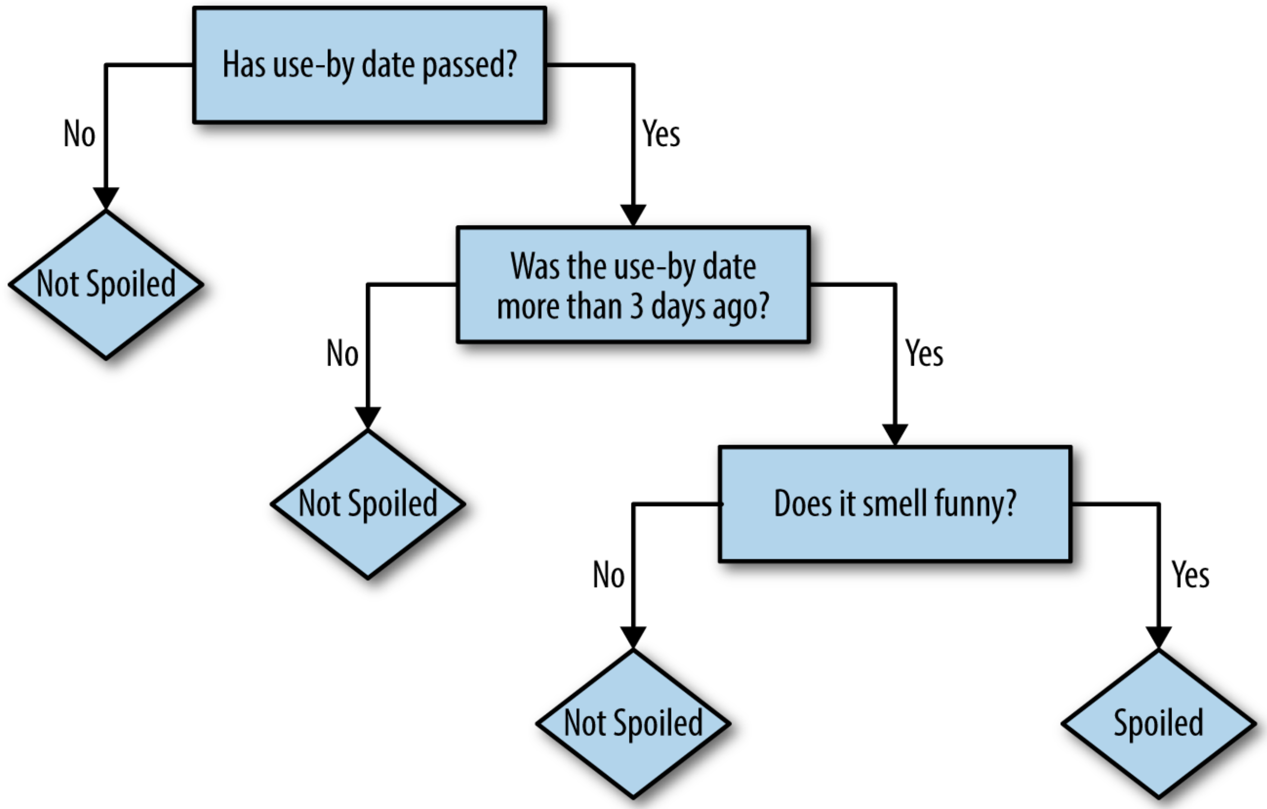
- Calculate the probability that an email is spam given that it contains the word "online"
  - P(spam) = 0.30
  - P("online") = 0.10
  - P("online" | spam) = 0.20
  - P("online" | ham) = 0.06

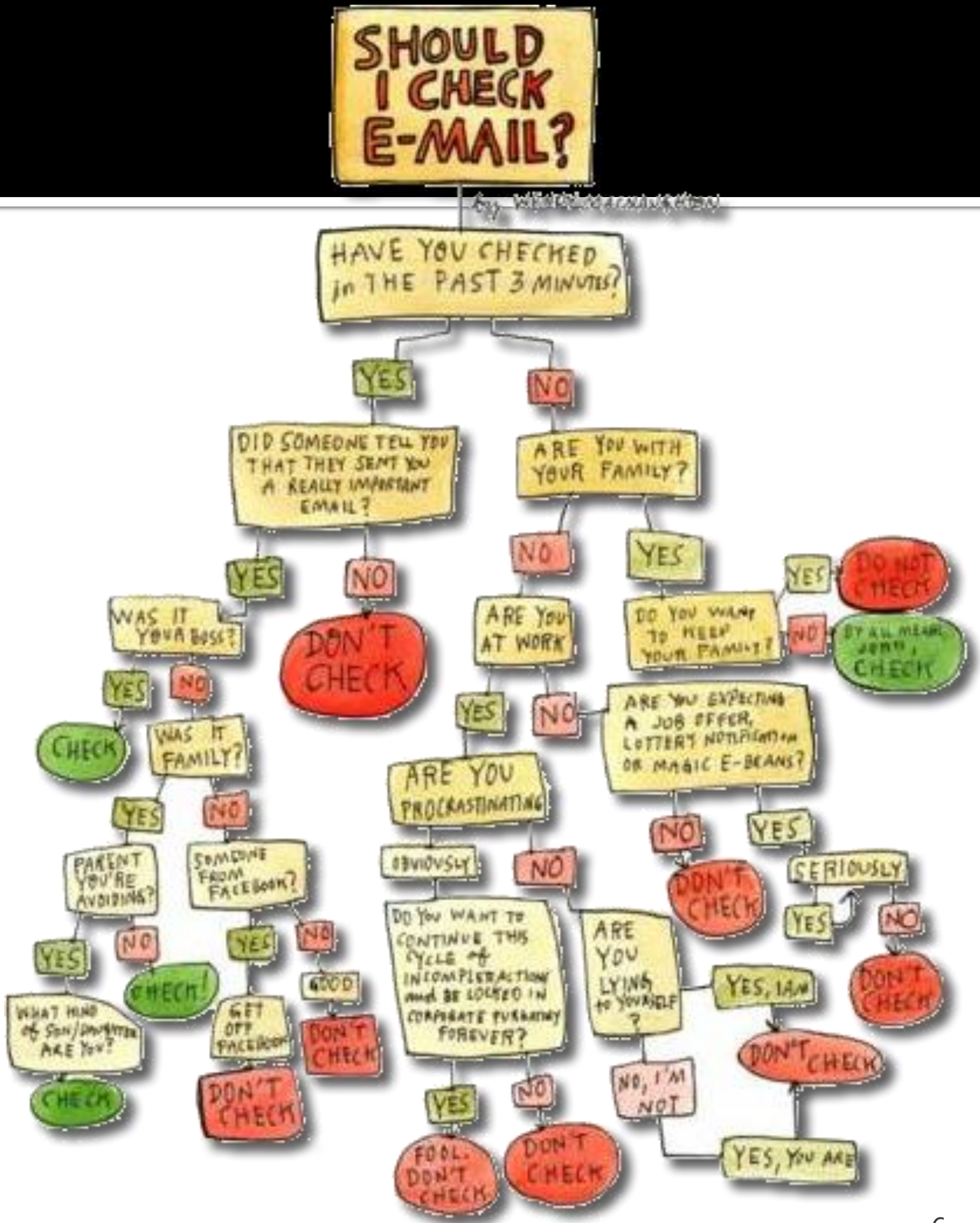
- Answer:
  - $P(spam|online) = \frac{P(online|spam)P(spam)}{P(online)} = \frac{0.2*0.3}{0.1} = 0.6$

Today's Outline

- Decision Trees
  - Introduction
  - Representation
  - Algorithms
  - Splitting
  - Extended example
  - Overfitting and Pruning
  - Extensions

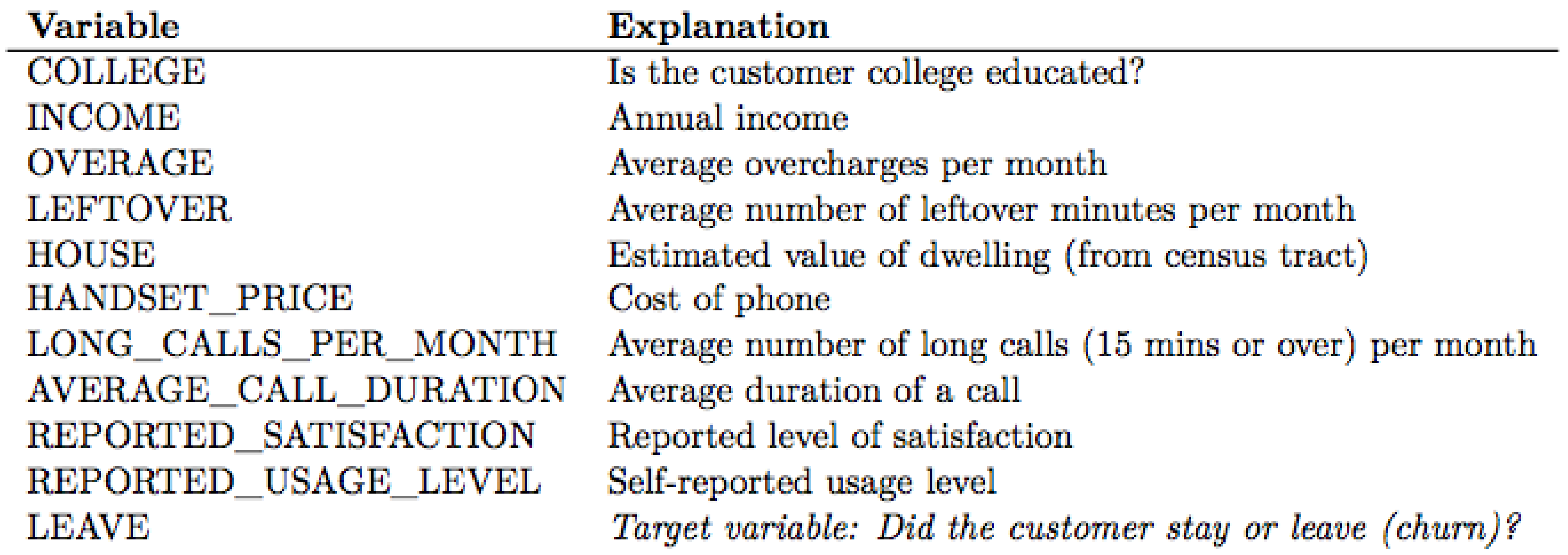
Decision Trees





Example: Customer churn

- Goal: reduce customer churn
  - E.g., target customers to encourage retention
  - Start by predicting who is likely to churn
  - What features to include?



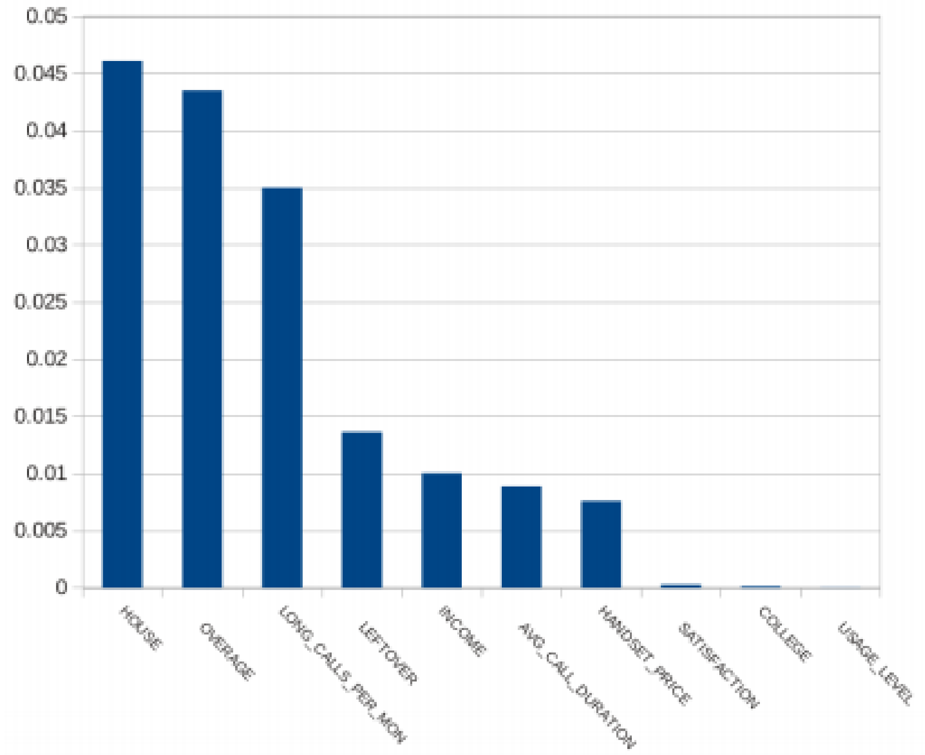
Variable	Explanation
COLLEGE	Is the customer college educated?
INCOME	Annual income
OVERAGE	Average overcharges per month
LEFTOVER	Average number of leftover minutes per month
HOUSE	Estimated value of dwelling (from census tract)
HANDSET_PRICE	Cost of phone
LONG_CALLS_PER_MONTH	Average number of long calls (15 mins or over) per month
AVERAGE_CALL_DURATION	Average duration of a call
REPORTED_SATISFACTION	Reported level of satisfaction
REPORTED_USAGE_LEVEL	Self-reported usage level
LEAVE	*Target variable: Did the customer stay or leave (churn)?*

[Provost and Fawcett]

Example: Customer churn

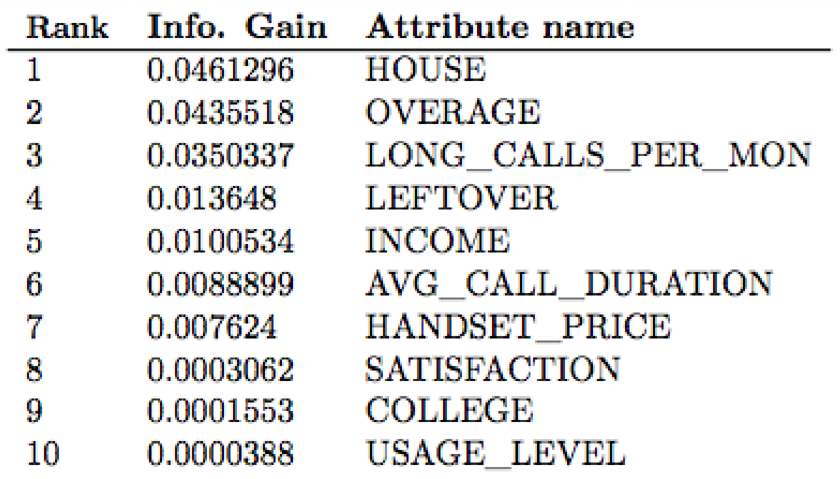
- A simple approach: “Forward Selection”
  - Add features to a model (e.g., kNN, logistic regression, Naïve Bayes) one at a time, based on the relevance of that feature
  - How to define “relevance”?
    - Unconditional correlation
    - Information gain (we’ll define this soon)

Rank	Info. Gain	Attribute name
1	0.0461296	HOUSE
2	0.0435518	OVERAGE
3	0.0350337	LONG_CALLS_PER_MON
4	0.013648	LEFTOVER
5	0.0100534	INCOME
6	0.0088899	AVG_CALL_DURATION
7	0.007624	HANDSET_PRICE
8	0.0003062	SATISFACTION
9	0.0001553	COLLEGE
10	0.0000388	USAGE_LEVEL



Example: Customer churn

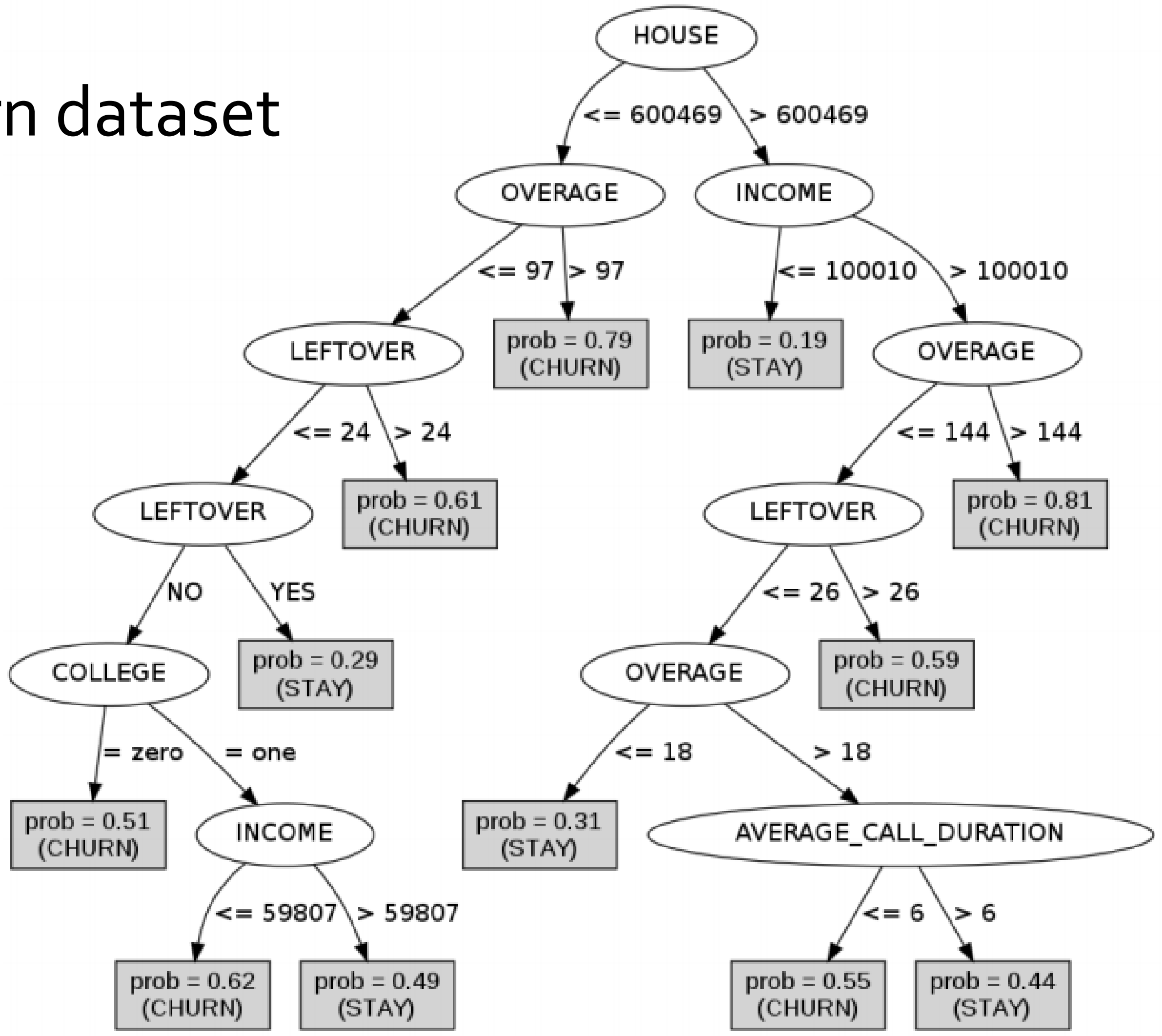
- A more structured approach: Decision Tree
  - Idea: Build tree from training data to explain labeled examples
  - Progressively add features that are most informative
  - Keep tree as small and as simple as possible, to help ensure generalization
- Different from forward selection
  - Decisions are made conditional on existing tree
  - E.g., second variable is not necessarily OVERAGE



Rank	Info. Gain	Attribute name
1	0.0461296	HOUSE
2	0.0435518	OVERAGE
3	0.0350337	LONG_CALLS_PER_MON
4	0.013648	LEFTOVER
5	0.0100534	INCOME
6	0.0088899	AVG_CALL_DURATION
7	0.007624	HANDSET_PRICE
8	0.0003062	SATISFACTION
9	0.0001553	COLLEGE
10	0.0000388	USAGE_LEVEL

Example: Customer churn

- Example decision tree with churn dataset



Decision trees

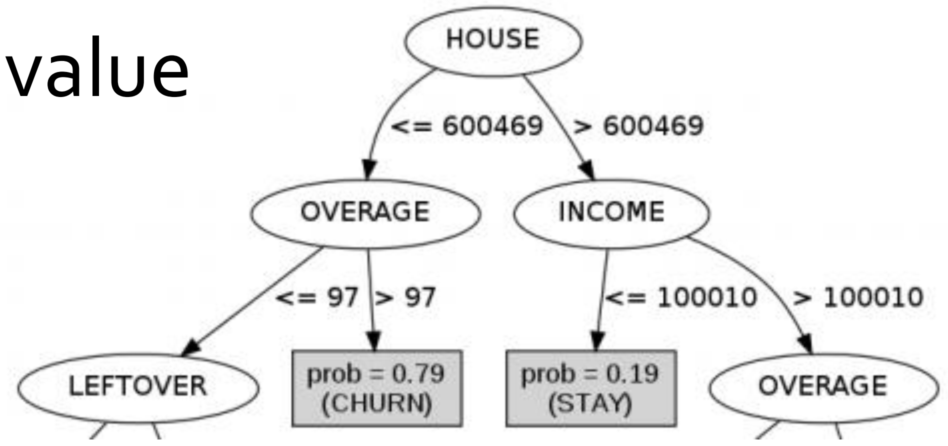
- Desirable properties
  - Very popular, easy to interpret
  - Reflects the logic of decision-making
  - Arbitrarily complex (non-linear functions)
  - Can be used for classification or continuous-valued prediction
  - Simple, fast algorithms for generating compact trees from data

Outline

- Decision Trees
  - Introduction
  - **Representation**
  - Algorithms
  - Splitting
  - Extended example
  - Overfitting and Pruning
  - Extensions

Representation

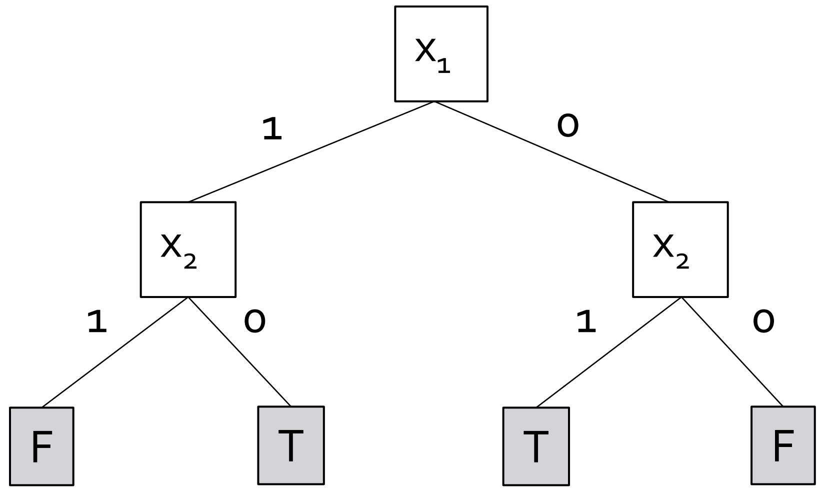
- Internal nodes test the value of a feature
  - Categorical
  - Binary
  - Continuous
- Branches indicate possible values for feature
- Leaf nodes output a predicted class or value



Simple Example: XOR

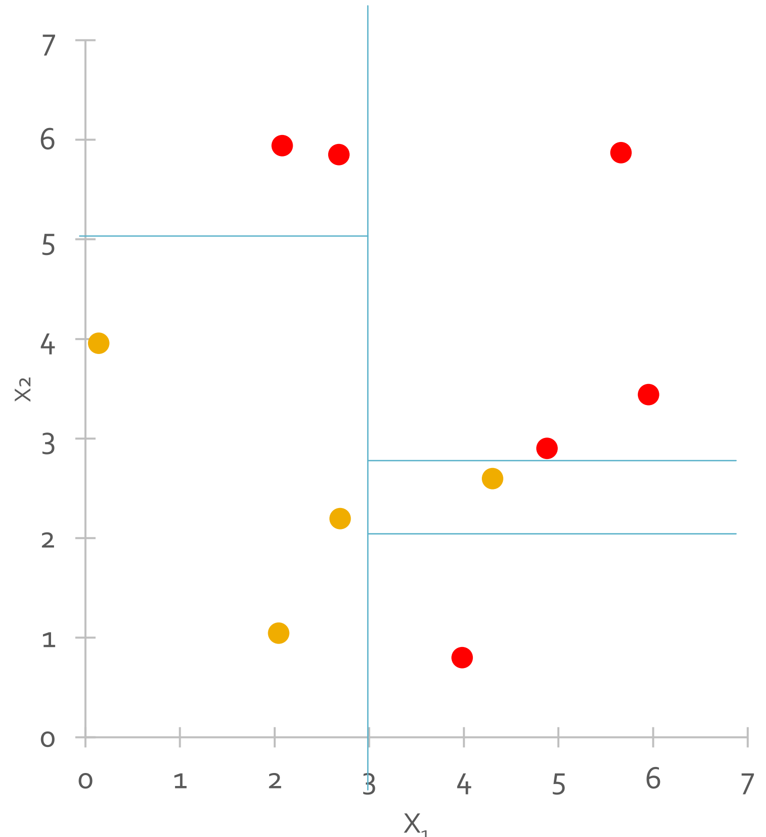
- Expressiveness
  - Any logical function that can be encoded in a truth table can be expressed in a decision tree!
  - Why? For any truth table, use each variables as a split

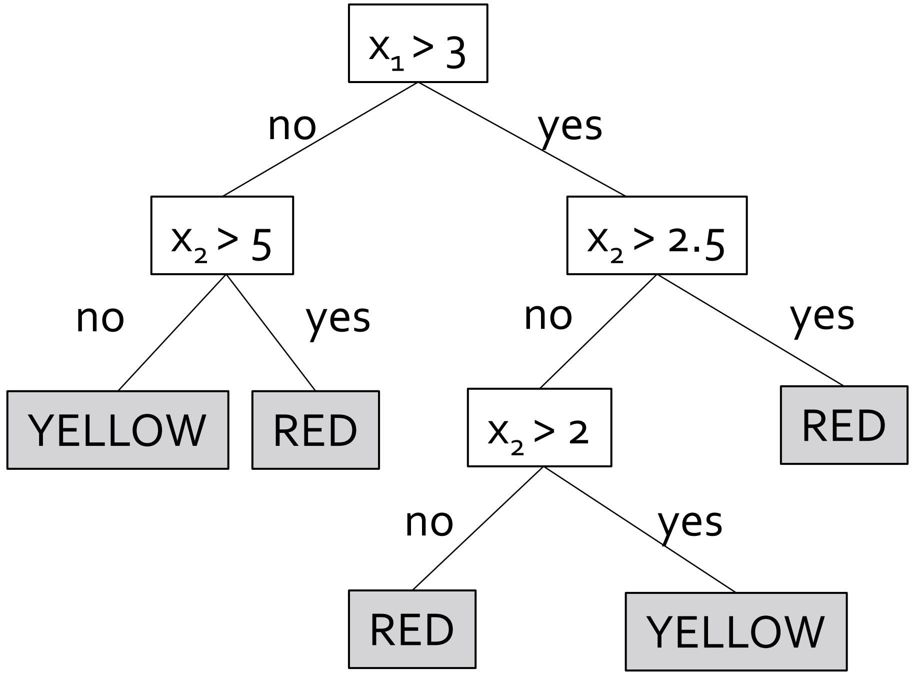
x_1	x_2	y
1	1	FALSE
1	0	TRUE
0	1	TRUE
0	0	FALSE



Decision boundaries

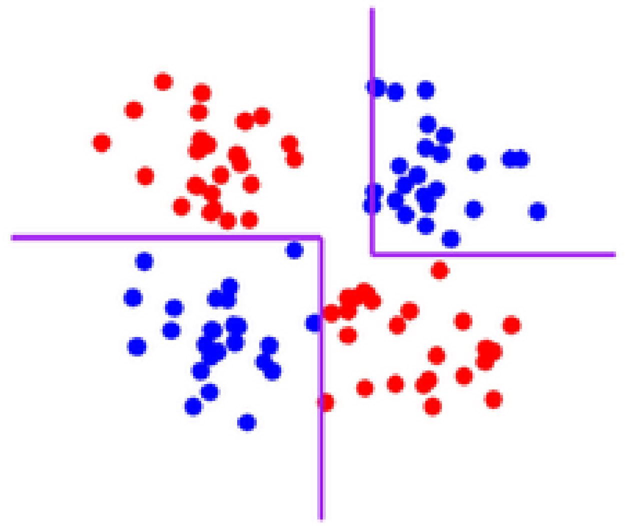
- What does the decision boundary of a decision tree look like?
  - Compare to k-NN? Logistic regression?

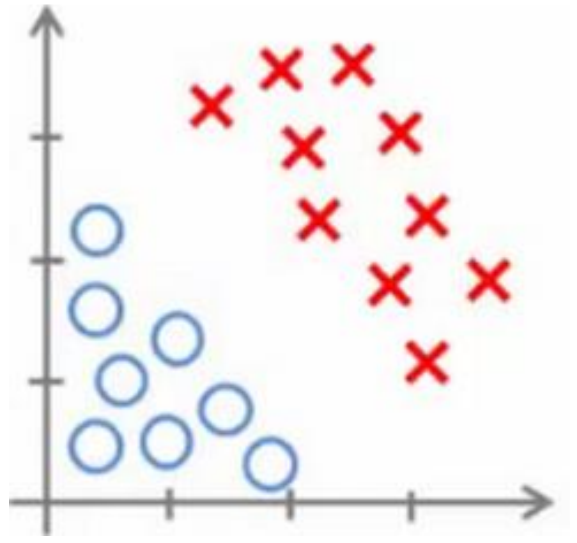




Decision boundaries

- "Hyper-rectangles" partition high-dimensional space
- Diagonal lines approximated as step function





Hypothesis space

- How many possible decision trees over *N* binary variables?
  - = number of *distinct* truth tables with 2^N rows
  - = 2^(2^N)
  - e.g. 6 attributes: 18,446,744,073,709,551,616 possible trees
  - Be careful!

- More expressive hypothesis spaces…
  - Increases chance that target function can be expressed
  - Increases hypotheses consistent with training data
  - But we may fail to generalize (i.e., overfitting)

Good vs. bad trees

- Many trees perfectly classify all training examples
  - A trivial model has a leaf for every training example
  - Doesn’t matter what the root feature is
- But we want our tree to generalize
  - i.e., we want the smallest tree that explains the data
  - Will need methods to reduce overfitting (pruning, forests, …)

Intuition check

- True or False: A decision tree is able to recover non-linear decision boundary

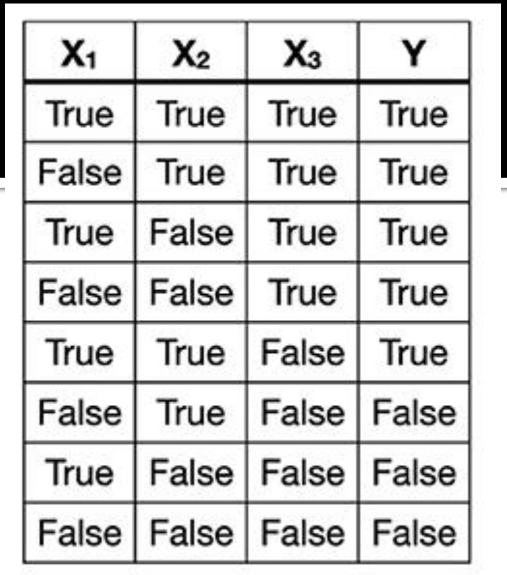
Outline

- Decision Trees
  - Introduction
  - Representation
  - **Algorithms**
  - Splitting
  - Extended example
  - Overfitting and Pruning
  - Extensions

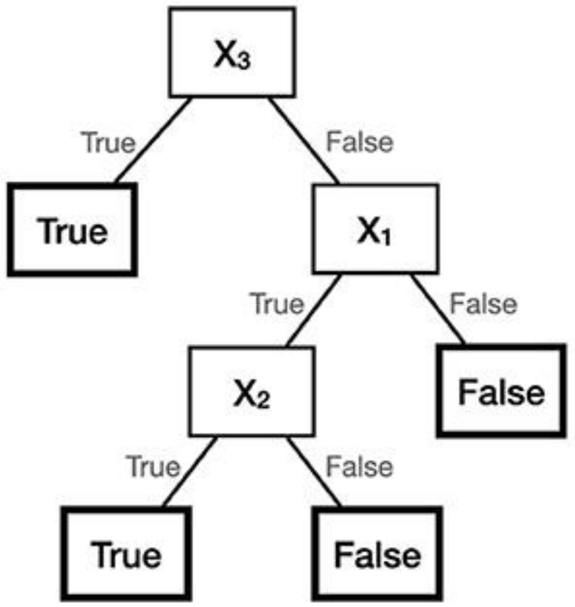
Building a tree (recursively)

- Goal: find a tree consistent with training examples
- Algorithm: (recursively) choose the most significant attribute as the root of the (sub)tree
  - Example with binary attributes

```
GrowTree(S):
        if y==0 for all <x,y> in S:
                return new leaf(0)
        else if y==1 for all <x,y> in S:
                return new leaf(1)
        else:
                choose best attribute x_j
                S0 = all <x,y> in S with x_j==0
                S1 = all <x,y> in S with x_j==1
                return new node(x_j,
                                   GrowTree(S0),
                                   GrowTree(S1))
```



X_1	X_2	X_3	Y
True	True	True	True
False	True	True	True
True	False	True	True
False	False	True	True
True	True	False	True
False	True	False	False
True	False	False	False
False	False	False	False

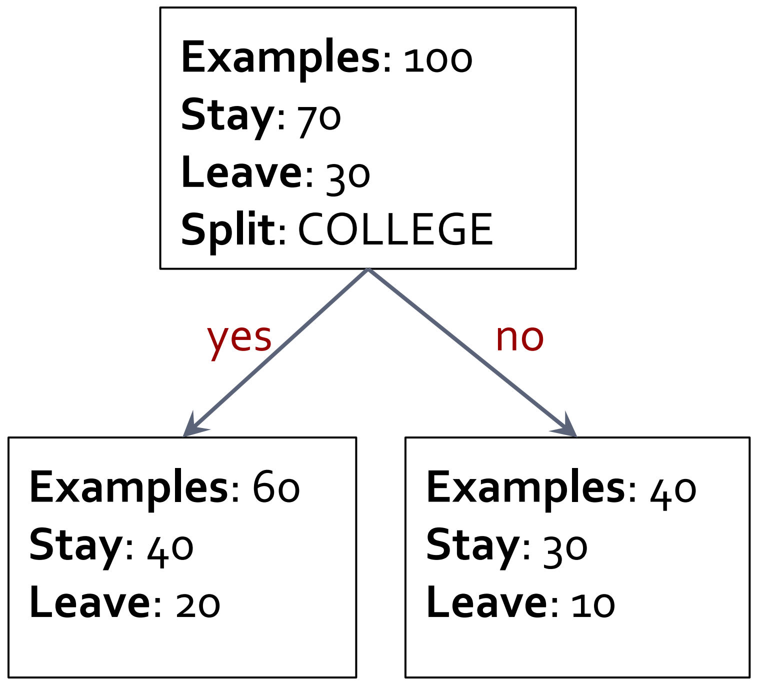


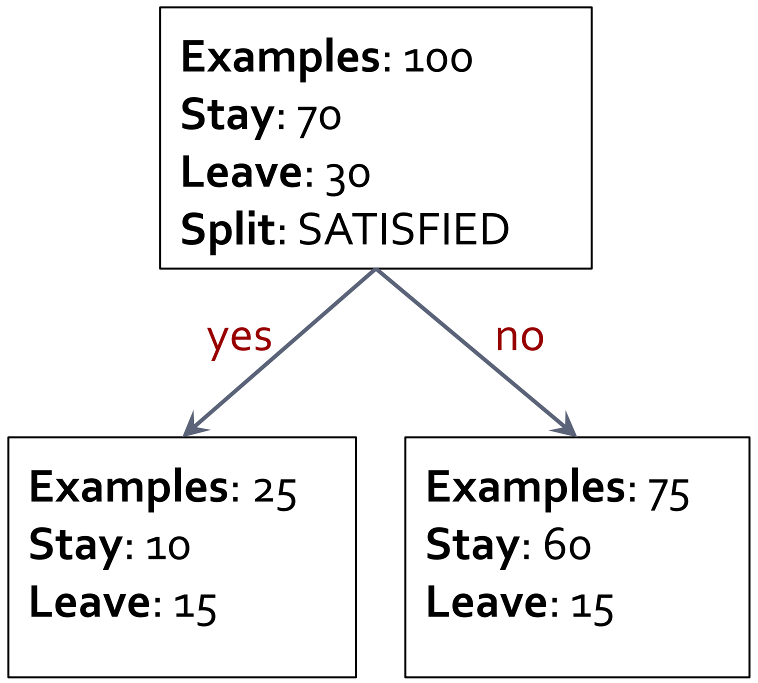
Outline

- Decision Trees
  - Introduction
  - Representation
  - Algorithms
  - **Splitting**
  - Extended example
  - Overfitting and Pruning
  - Extensions

What is the "best" attribute?

- Good attributes split examples into pure subsets
- Should we split on COLLEGE or SATISFIED?





Information Theory

- Shannon's Game: predict the character

```
ARE WE THERE YE_
```

  - Explores mathematics of encoded messages

- How much information is conveyed by a single letter?
  - If the alphabet contains only one letter: 0 bits
  - With 27 equiprobable letters: 4.8 bits
  - Shannon's estimate of English characters: ~1 bit

- Shannon's "information" is expected code length to convey a message

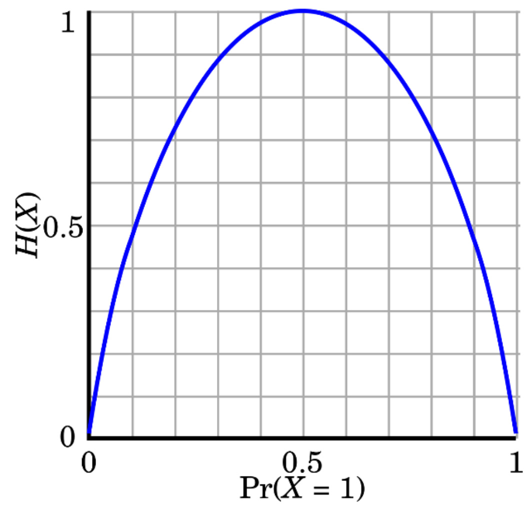
Claude Shannon
1916-2001

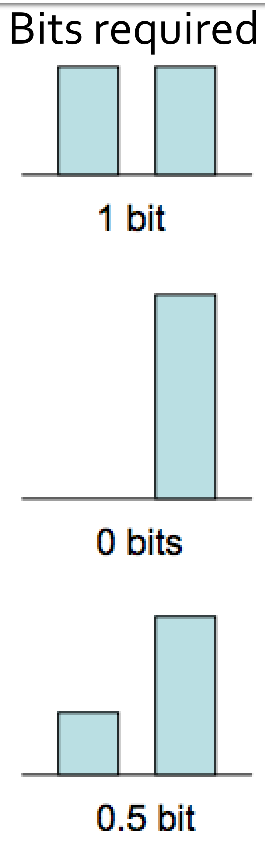
One approach: Reduce entropy

- Entropy is a measure of uncertainty
  - A distribution where one value has P(1) has no entropy
  - More uncertainty requires longer codes
  - Uniform distribution maximizes entropy

- Formula for entropy:

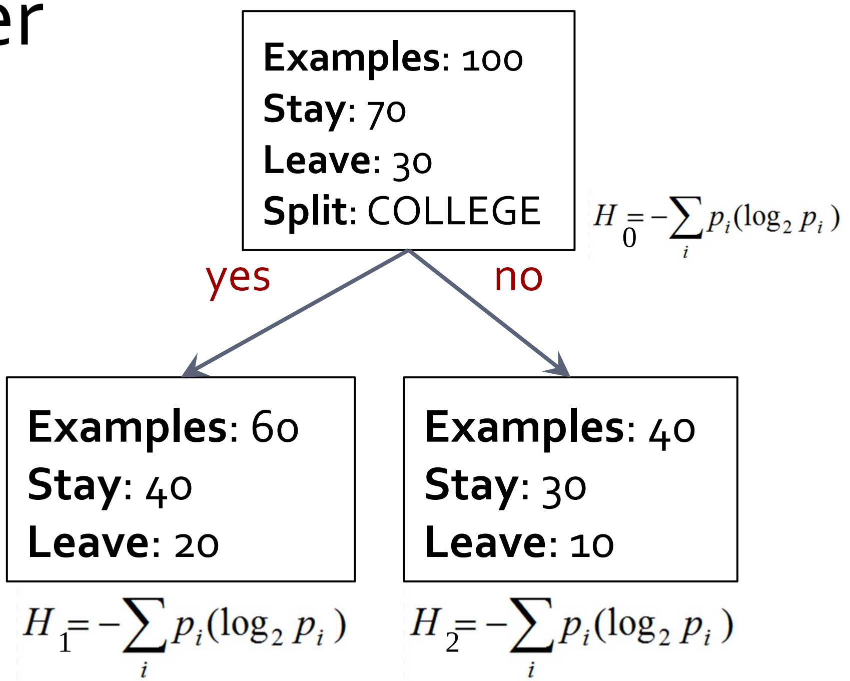
$$H = -\sum_i p_i (\log_2 p_i)$$

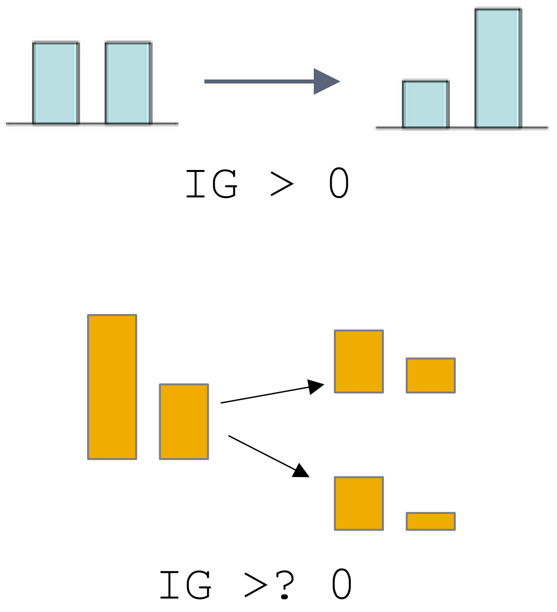




Information gain

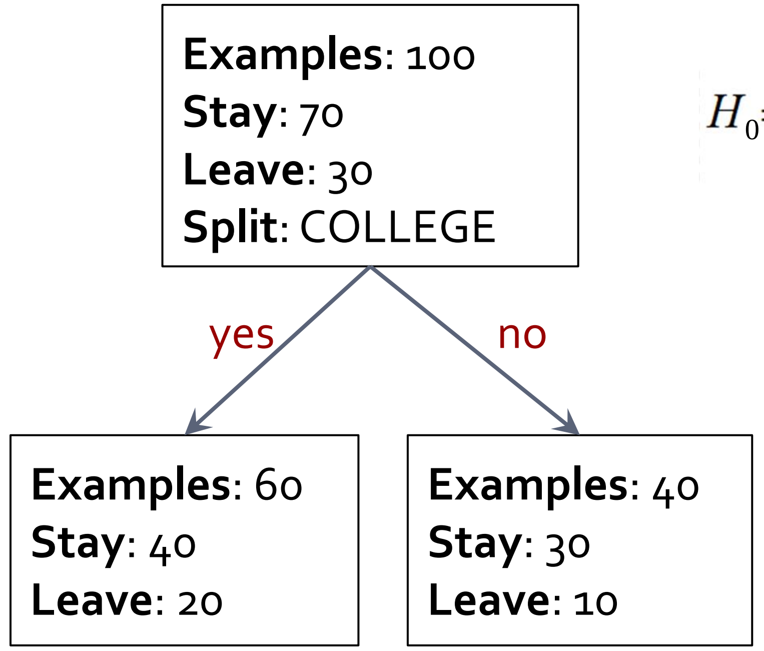
- Information Gain: describes a *change* in entropy
  - Entropy before - Entropy after
- Example with positive information gain:
- Our example is trickier
  - Requires a calculator





Information gain

- Before we calculate information gain, notice that the split produces branches of different sizes (60 vs. 40)
  - Solution: weight by the number of examples



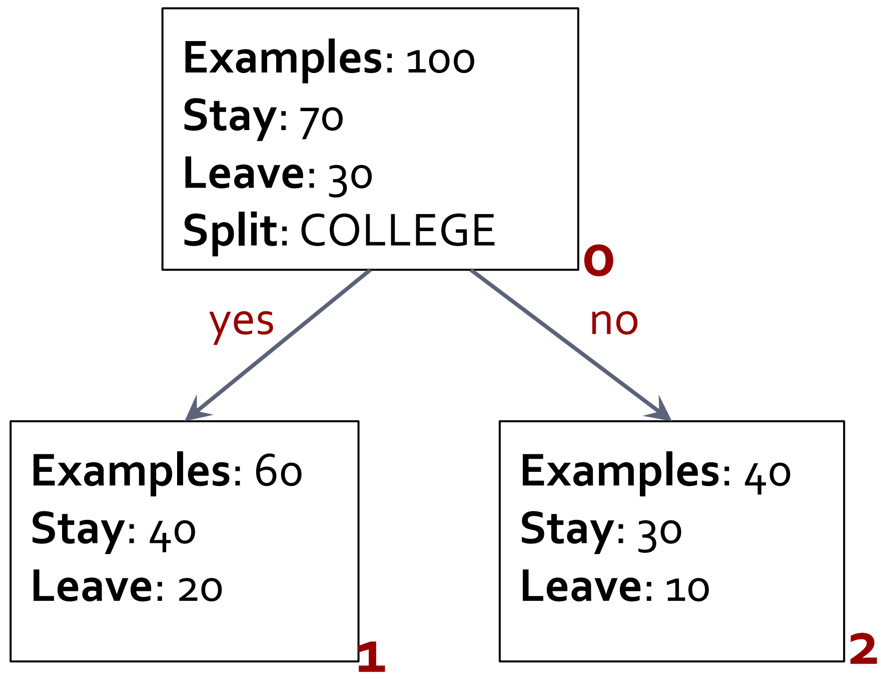
$$H_0 = -\sum_i p_i (\log_2 p_i)$$

$$H_1 = -\sum_i p_i (\log_2 p_i)$$

$$H_2 = -\sum_i p_i (\log_2 p_i)$$

Information gain: example

$$H = -\sum_i p_i(\log_2 p_i)$$



H_0 = -.7 log(.7)-.3 log(.3) = **.881**

H_1 = -.66 log(.66)-.33 log(.33)= **.92**

H_2 = -.75 log(.75) -.25 log(.25)= **.81**

IG(COLLEGE) = .881-[.6(.92)+.4(.81)]
= .881-.875
= **.006**
(= **small gain)**

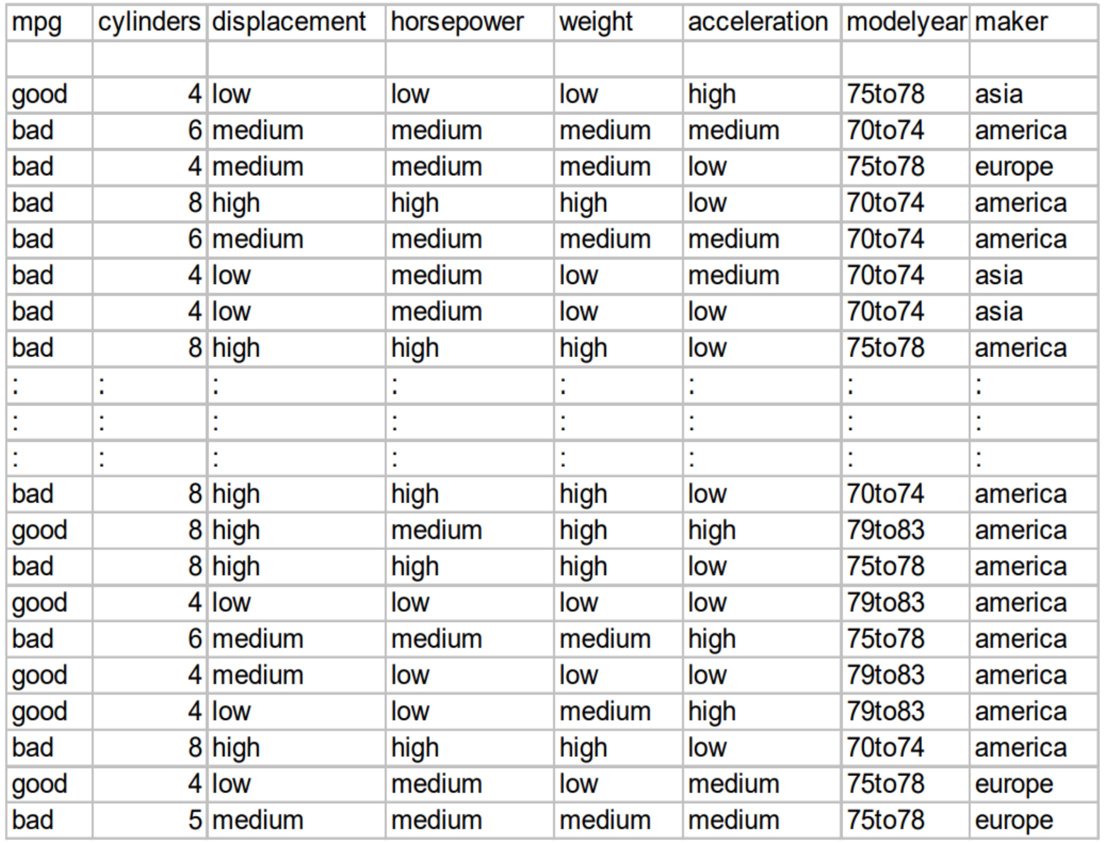
IG(SATISFIED)= .88-[.25(.97)+.75(.72)]
= **.09**
(= **larger gain)**

Outline

- **Decision Trees**
  - Introduction
  - Representation
  - Algorithms
  - Splitting
  - **Extended example**
  - Overfitting and Pruning
  - Extensions

Extended Example: Predicting MPG

- Goal: predict whether a car gets “good” or “bad” gas mileage

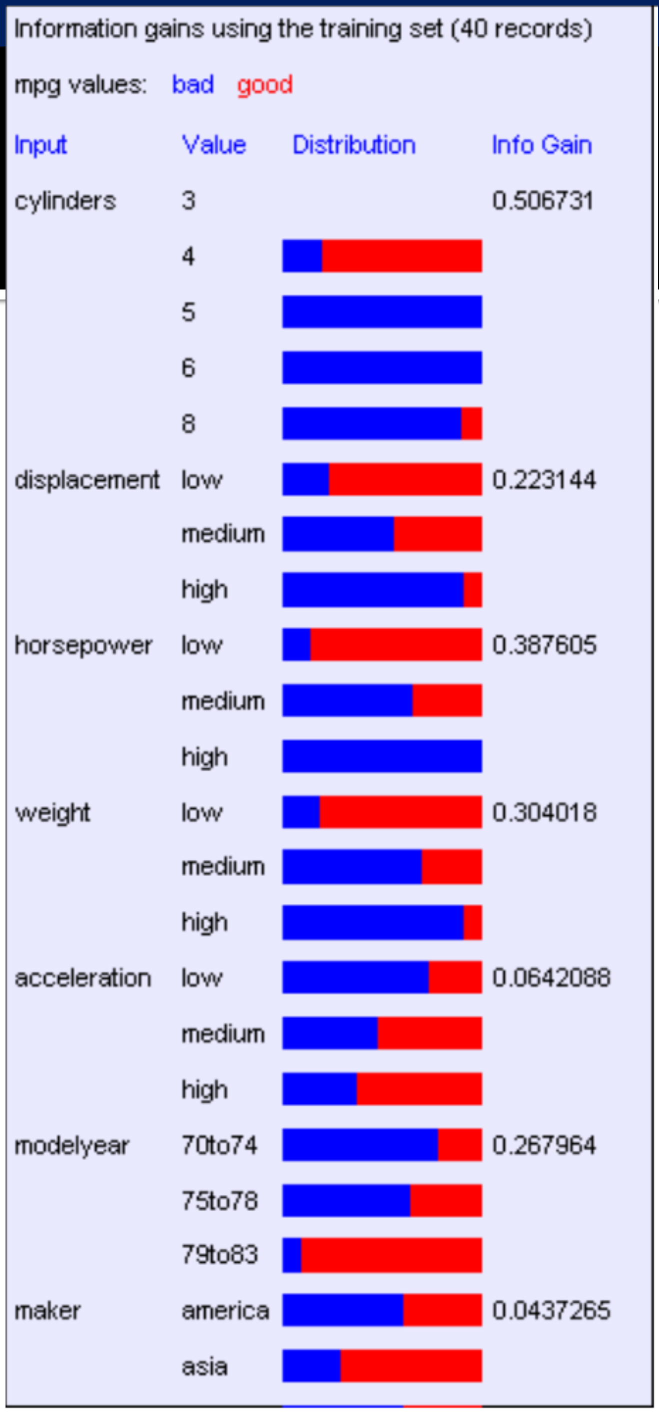


mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low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

[Andrew Moore]

The first split

- Each attribute is correlated with the target, to varying degrees.
- First step: calculate information gain for each possible split
  - (In this example, most attributes are categorical)
- Then: choose split that maximizes info gain

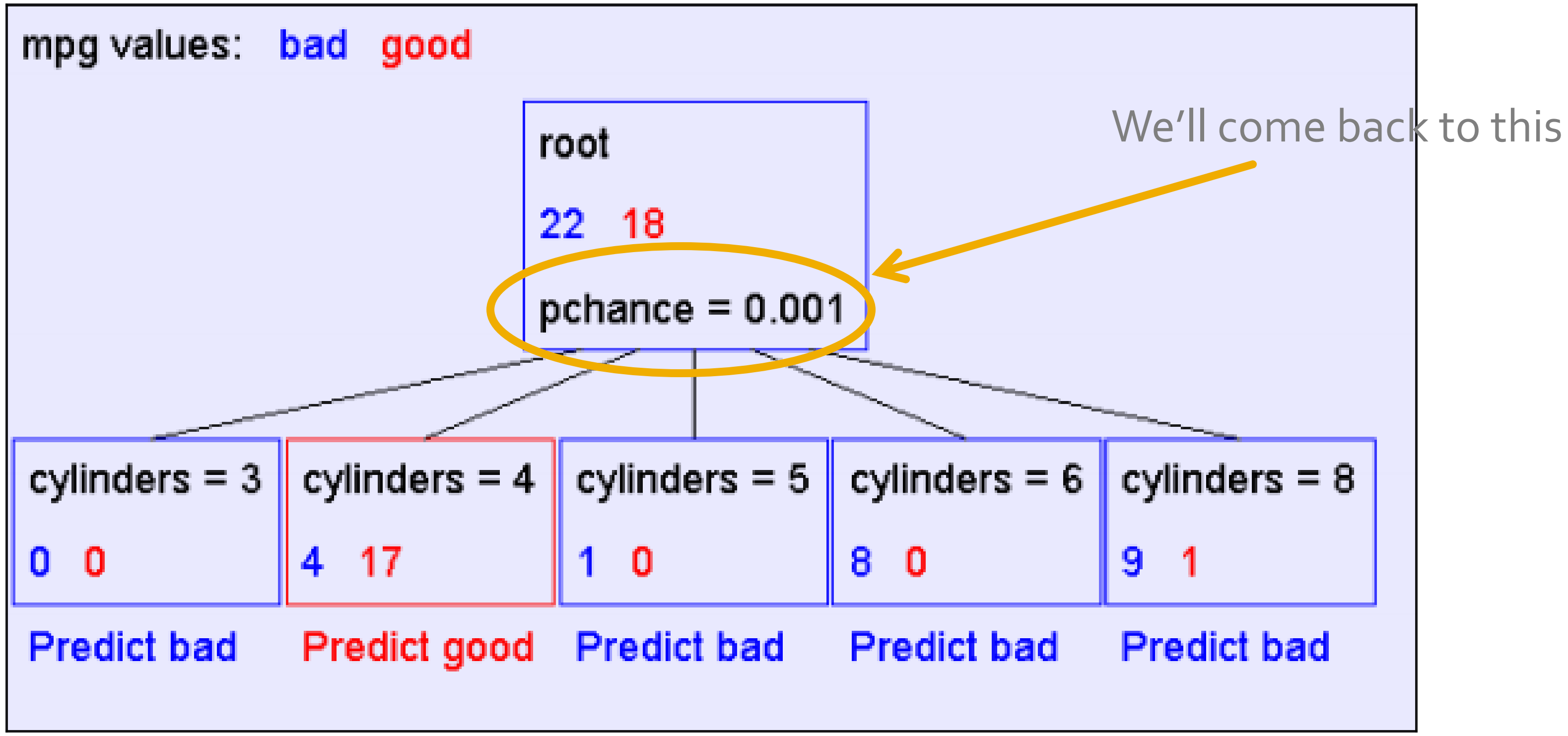


Information gains using the training set (40 records)

mpg values: bad good

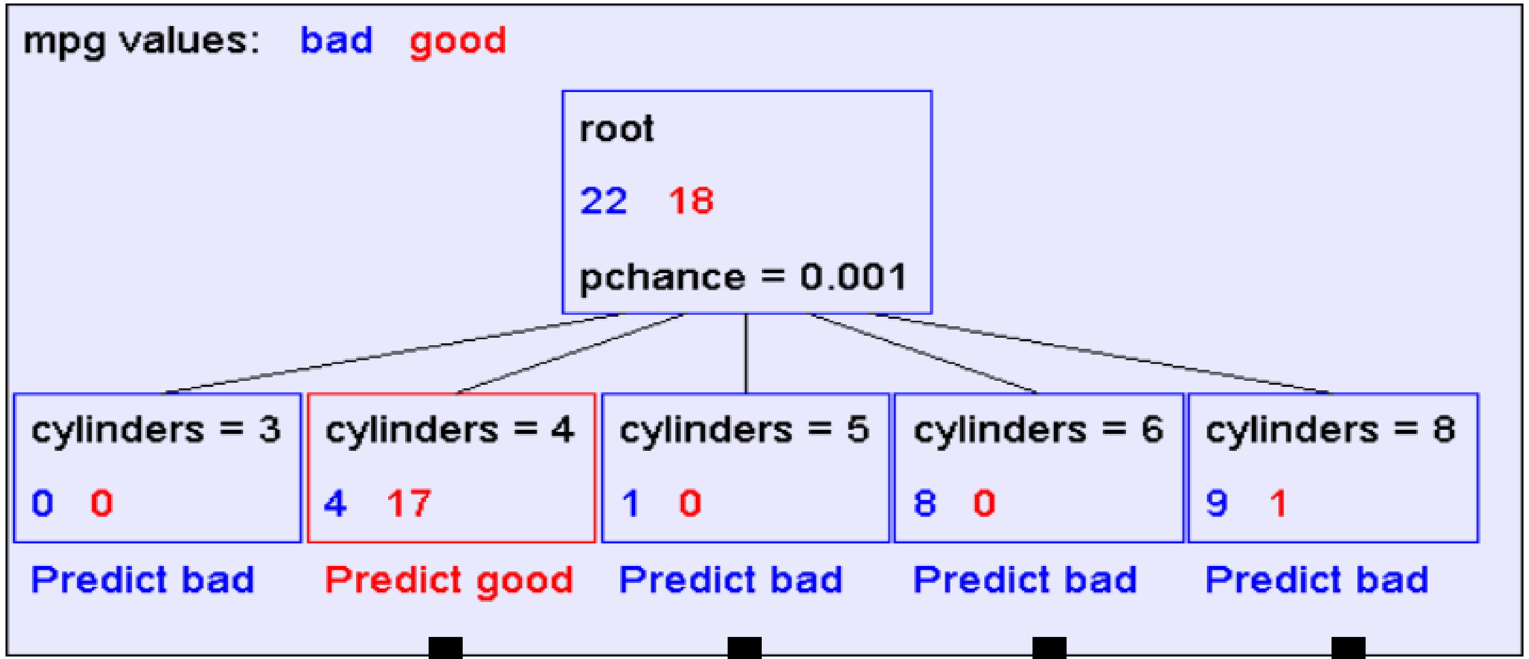
Input	Value	Distribution	Info Gain
cylinders	3		0.506731
	4	🟦🟥	
	5	🟦	
	6	🟦	
	8	🟦🟥	
displacement	low	🟦🟥	0.223144
	medium	🟦🟥	
	high	🟦🟥	
horsepower	low	🟦🟥	0.387605
	medium	🟦🟥	
	high	🟦	
weight	low	🟦🟥	0.304018
	medium	🟦🟥	
	high	🟦🟥	
acceleration	low	🟦🟥	0.0642088
	medium	🟦🟥	
	high	🟦🟥	
modelyear	70to74	🟦🟥	0.267964
	75to78	🟦🟥	
	79to83	🟦🟥	
maker	america	🟦🟥	0.0437265
	asia	🟦🟥	

Depth-1 tree: "decision stump"

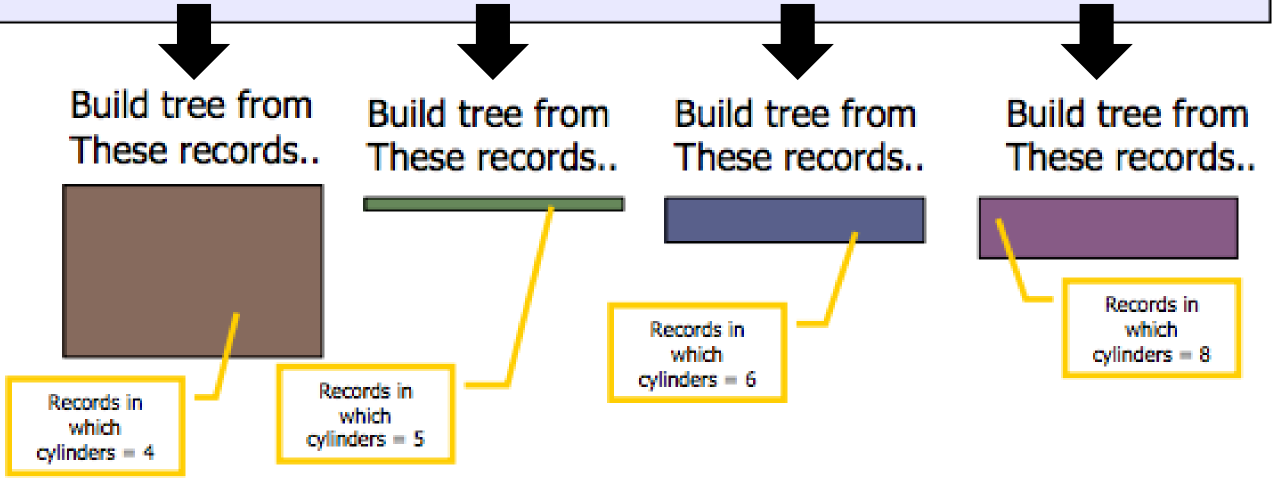


Recursive tree building

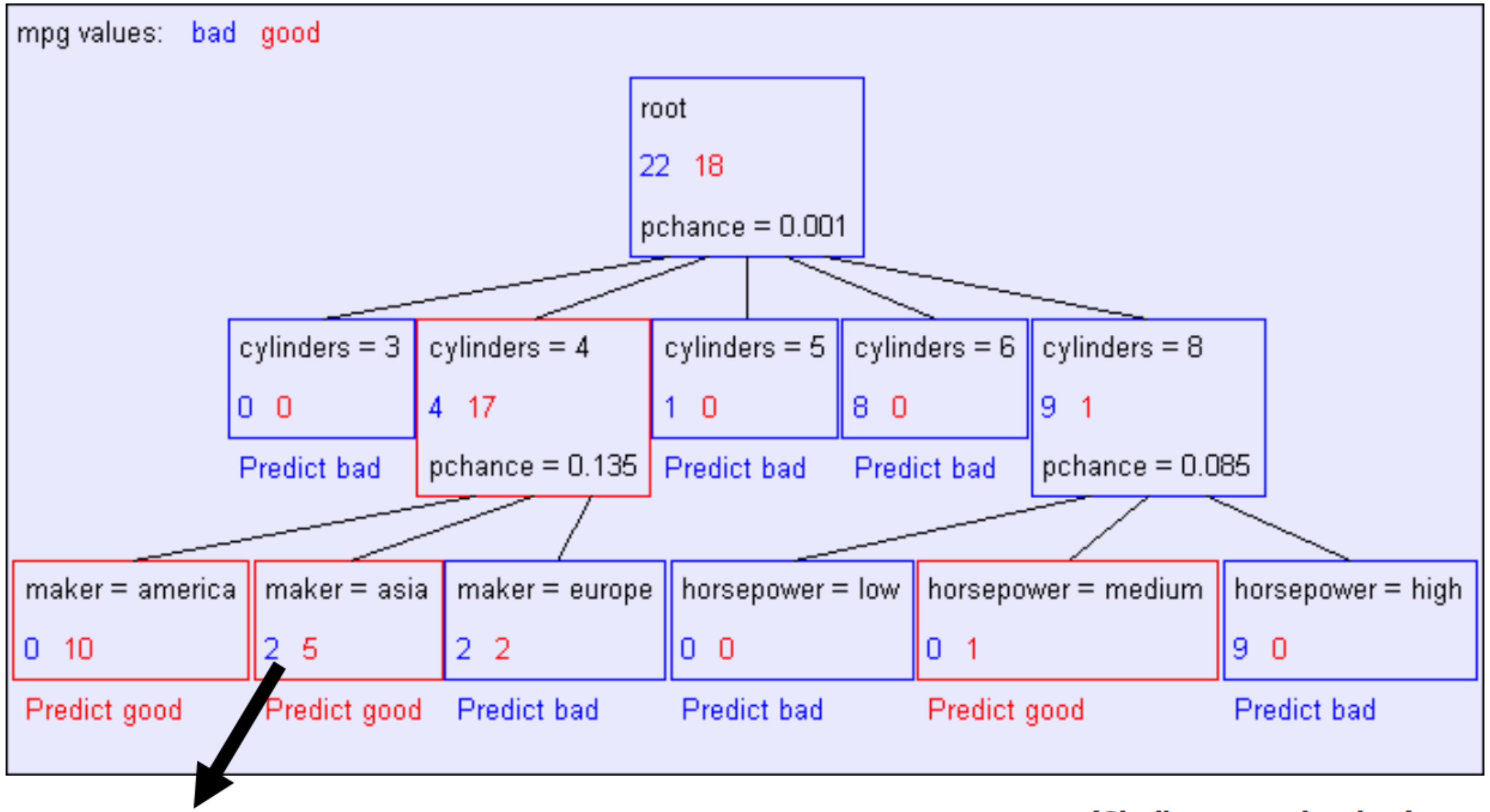
First split:



Recursive step:


Build tree from These records..
Records in which cylinders = 4
Build tree from These records..
Records in which cylinders = 5
Build tree from These records..
Records in which cylinders = 6
Build tree from These records..
Records in which cylinders = 8

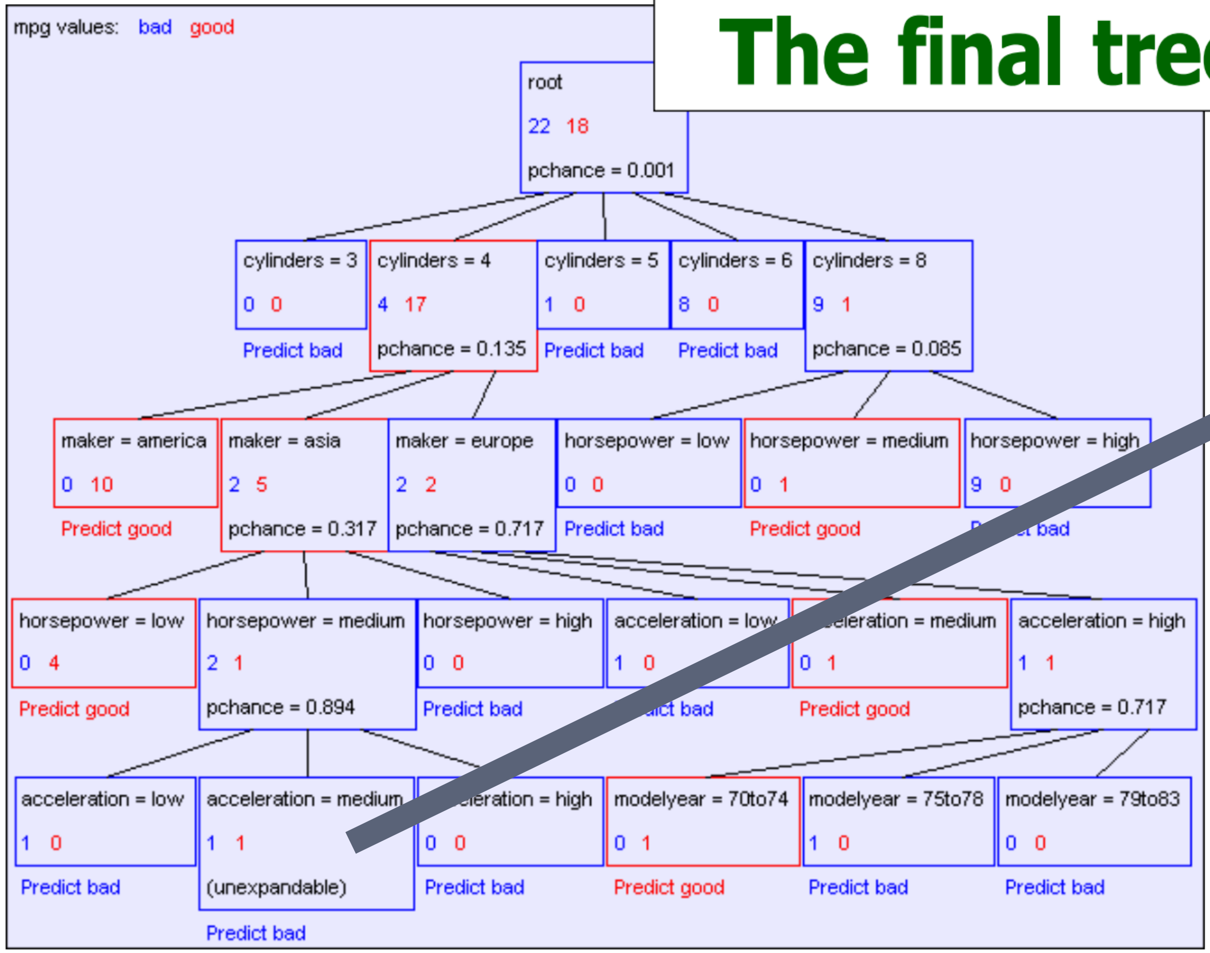
Second level (depth-2 tree)



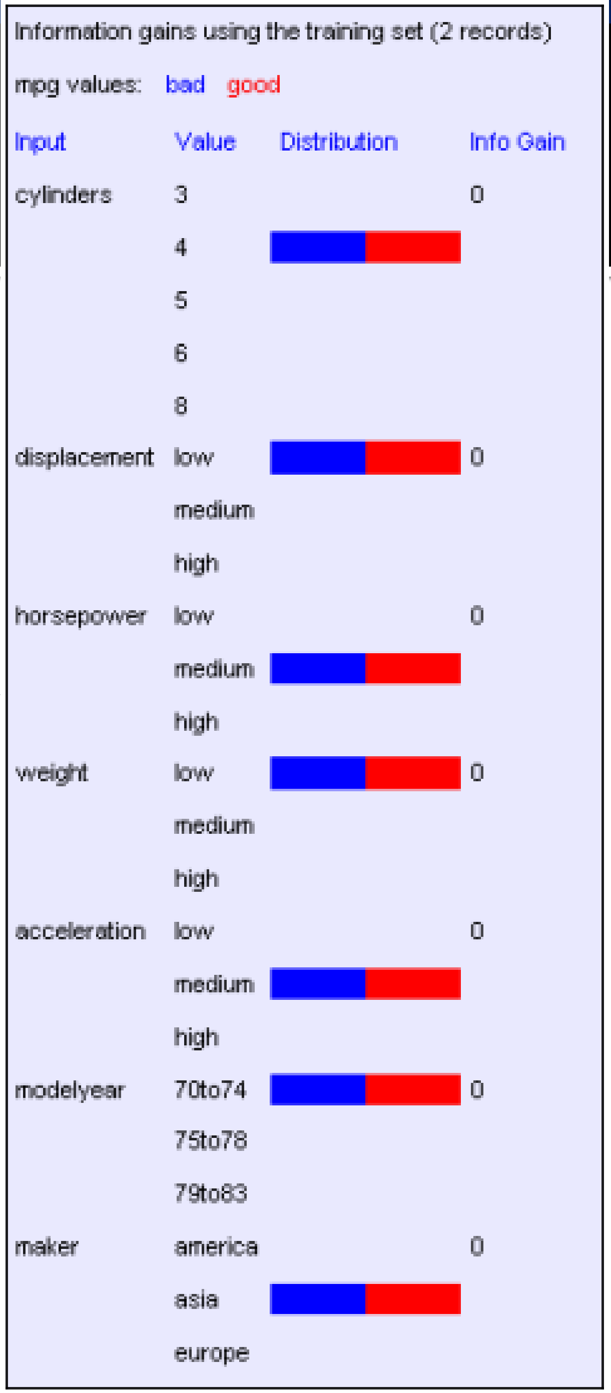
Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia

(Similar recursion in the other cases)

Final tree



The 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		

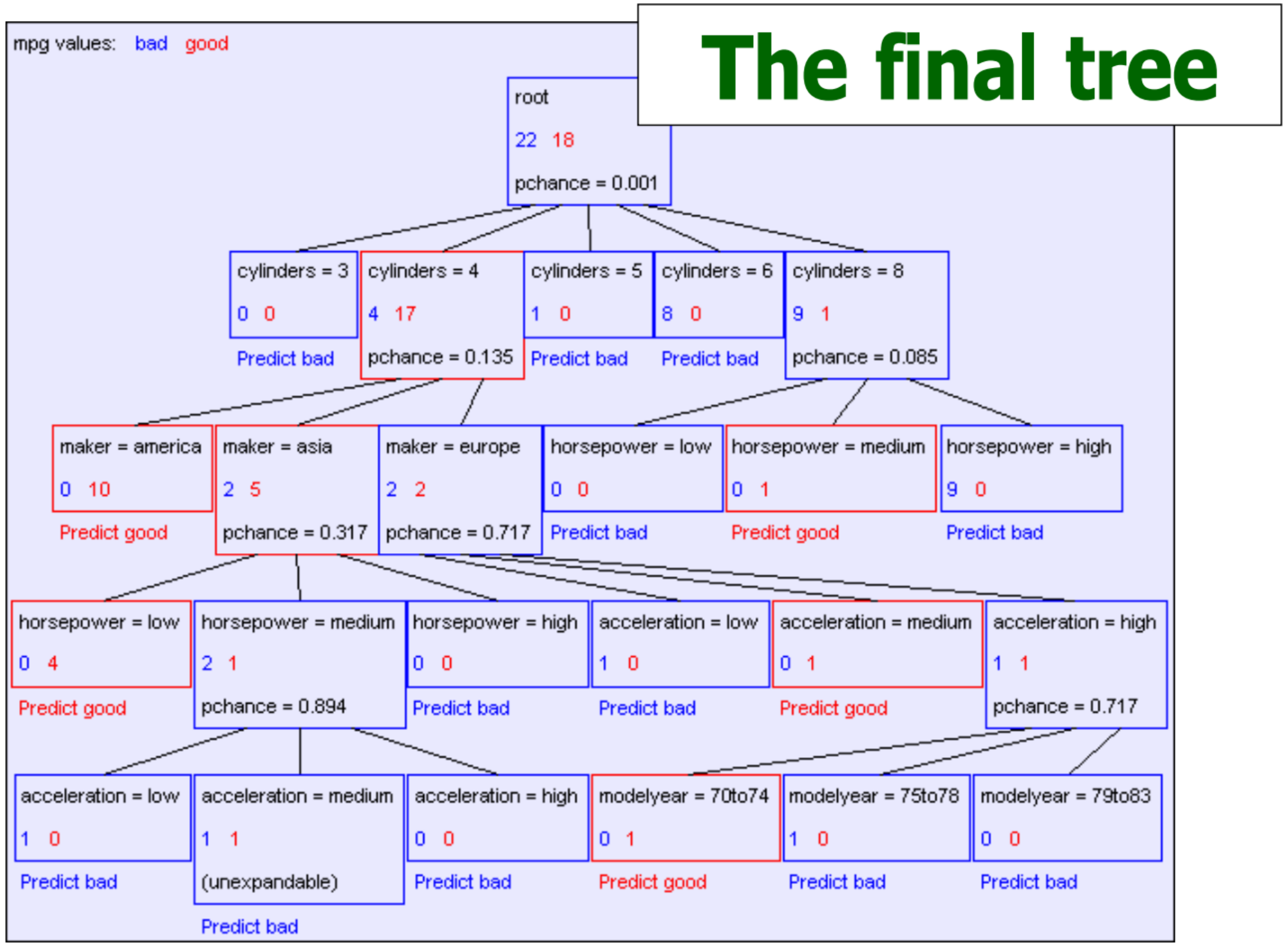
Recap: Decision Tree algorithm

```
GrowTree(S):
    if y==0 for all <x,y> in S:
        return new leaf(0)
    else if y==1 for all <x,y> in S:
        return new leaf(1)
    else:
        $x_j$ = max_info_gain(S)
        S0 = all <x,y> in S with $x_j$==0
        S1 = all <x,y> in S with $x_j$==1
        return new node($x_j$, GrowTree(S0), GrowTree(S1))
```

Outline

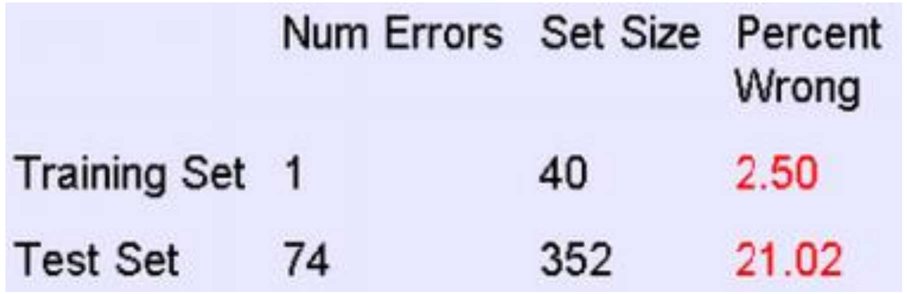
- Decision Trees
  - Introduction
  - Representation
  - Algorithms
  - Splitting
  - Extended example
  - **Overfitting and Pruning**
  - Extensions

Overfitting



Overfitting

- Overfitting strikes again:



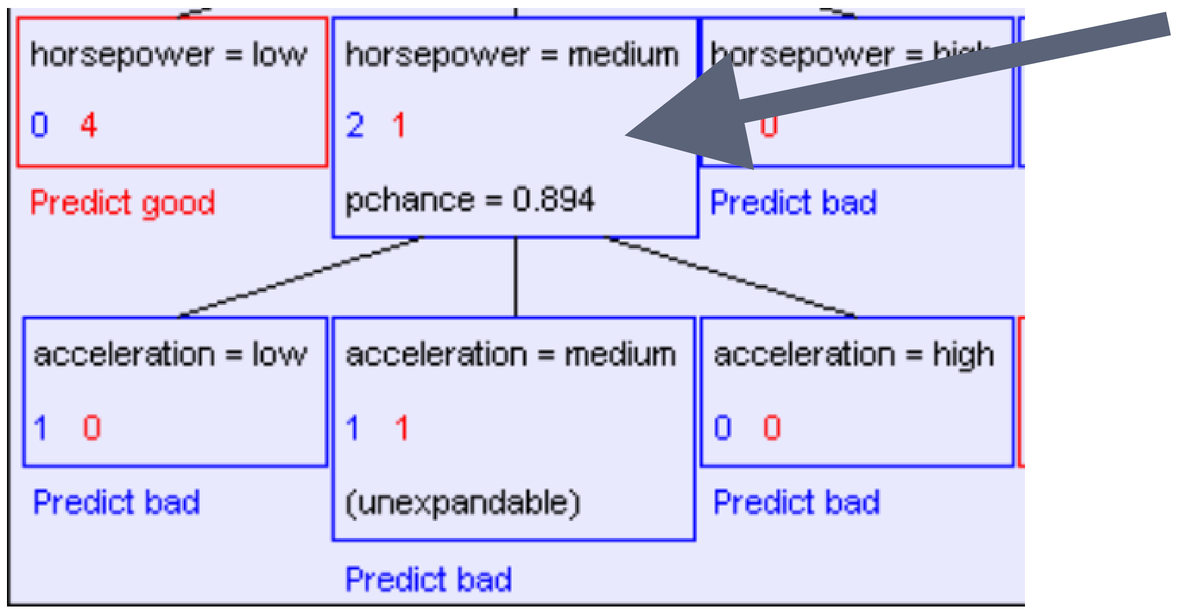
	Num Errors	Set Size	Percent Wrong
Training Set	1	40	2.50
Test Set	74	352	21.02

- How to deal with overfitting in Regression?
  - Regularization
- K-Nearest Neighbors?
  - Increasing K
- Naïve Bayes?
  - Smoothing

Overfitting in Decision Trees

- Three common solutions:
  1. Stop growing tree when split is not “useful”
  2. Grow tree, then prune afterwards
  3. Set maximum depth

Thresholding to stop growing



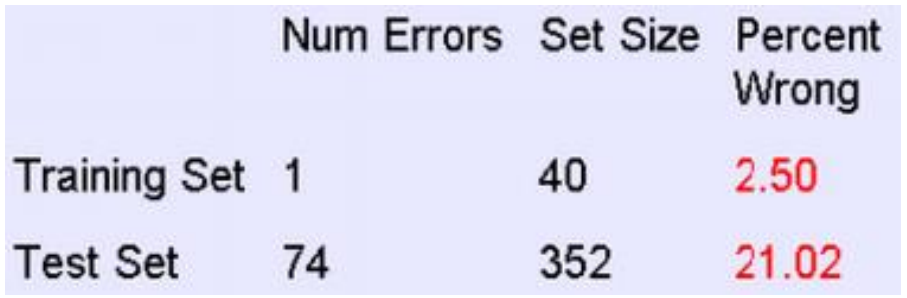
- Should we really split here?
- Only 3 relevant training examples
- The resulting distributions may be due to chance

- **One solution**: compute the value/ significance of each split
  - For instance, using information gain, or a chi-squared test
  - Only split if value exceeds some threshold

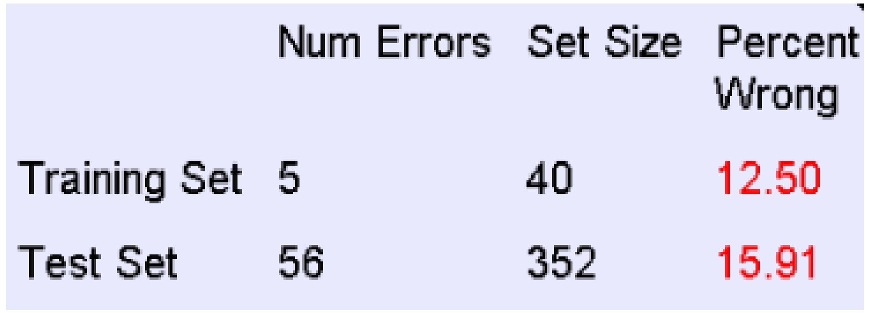
$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

Pruning

- Build the full decision tree
- Starting with the deepest nodes, delete splits where value of split does not exceed some threshold T
- Continue upward until no more nodes need to be pruned



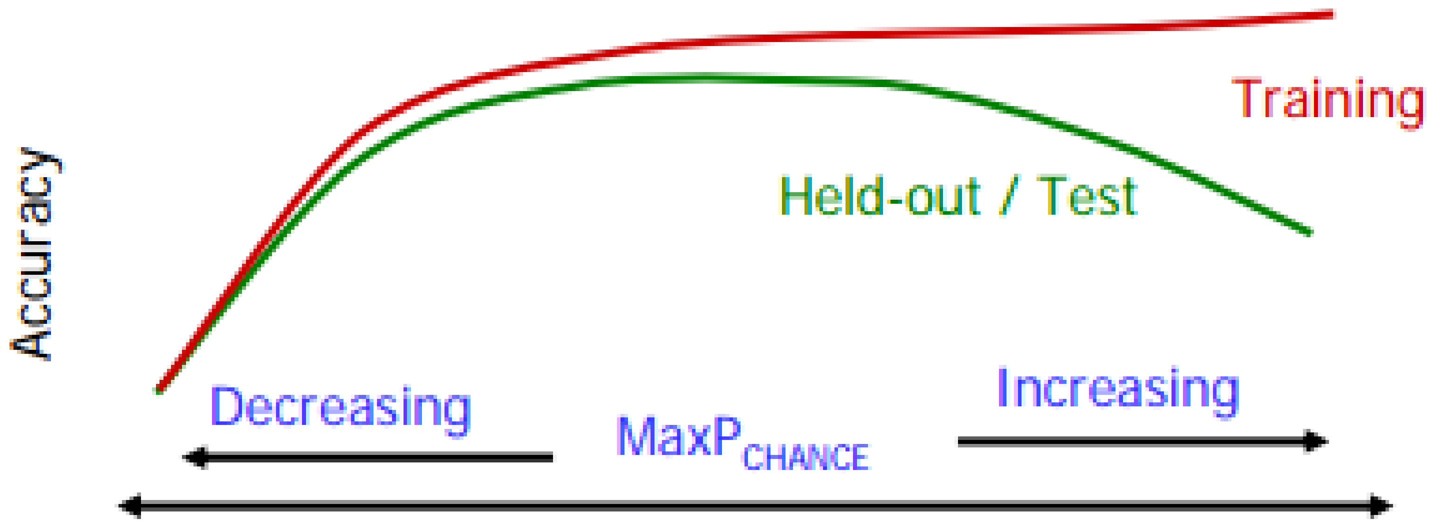
	Num Errors	Set Size	Percent Wrong
Training Set	1	40	2.50
Test Set	74	352	21.02



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

Regularization

- T is a regularization (hyper-)parameter
  - How to determine value?
  - Cross-Validation!

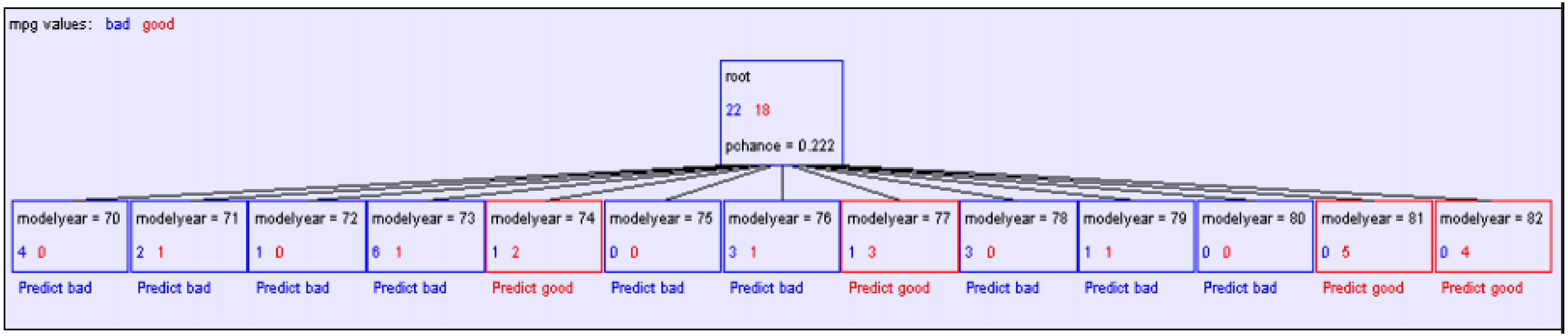


Outline

- Decision Trees
  - Introduction
  - Representation
  - Algorithms
  - Splitting
  - Overfitting and Pruning
  - **Extensions**

Multi-valued features

- Features with many discrete values:
  - Splits with many children (comparing Info Gain?)
  - Can produce degenerate cases
  - Common solution: One vs. all other values

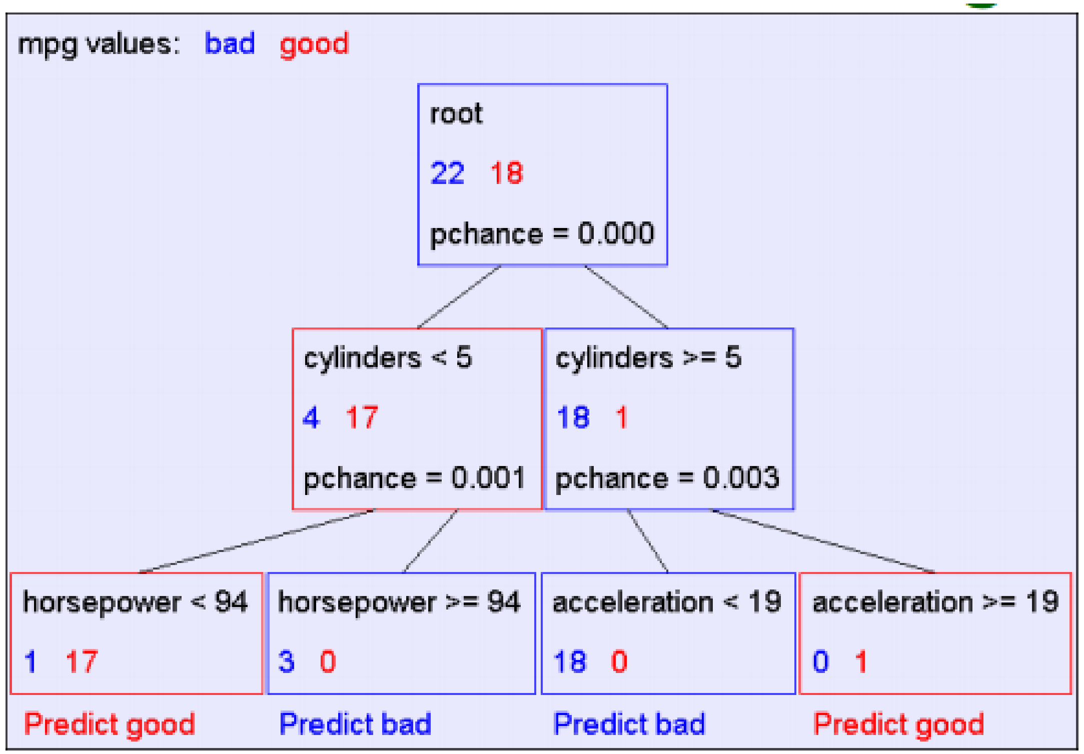


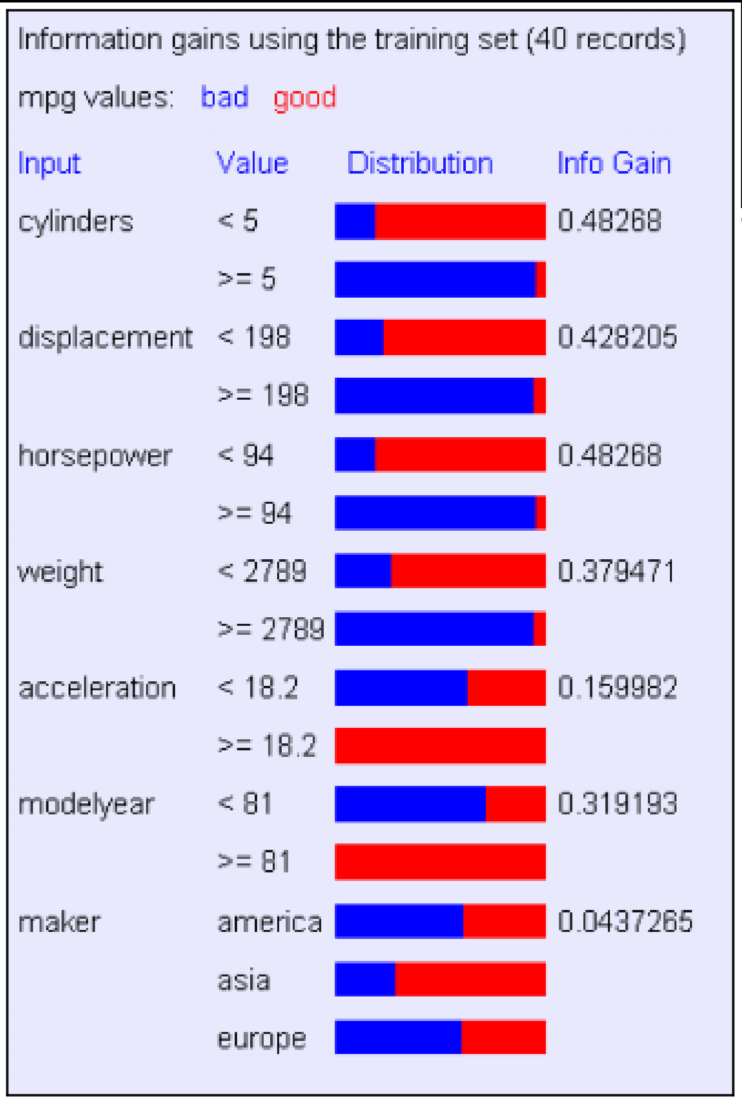
Continuous features

- Continuous features
  - Common solution: Bucket or threshold values
  - E.g., model years <1970, 1970-1980, >1980
- How to choose the buckets/thresholds?
  - Sort instances based on value of an attribute (e.g. year)
  - Identify adjacent examples that differ in their label
    - This generates a set of candidate thresholds splitting thresholds for that feature
  - Use information gain to decide best threshold

Thresholded splits

- Bucketing example:
  - Creates deeper, denser tree





Information gains using the training set (40 records)

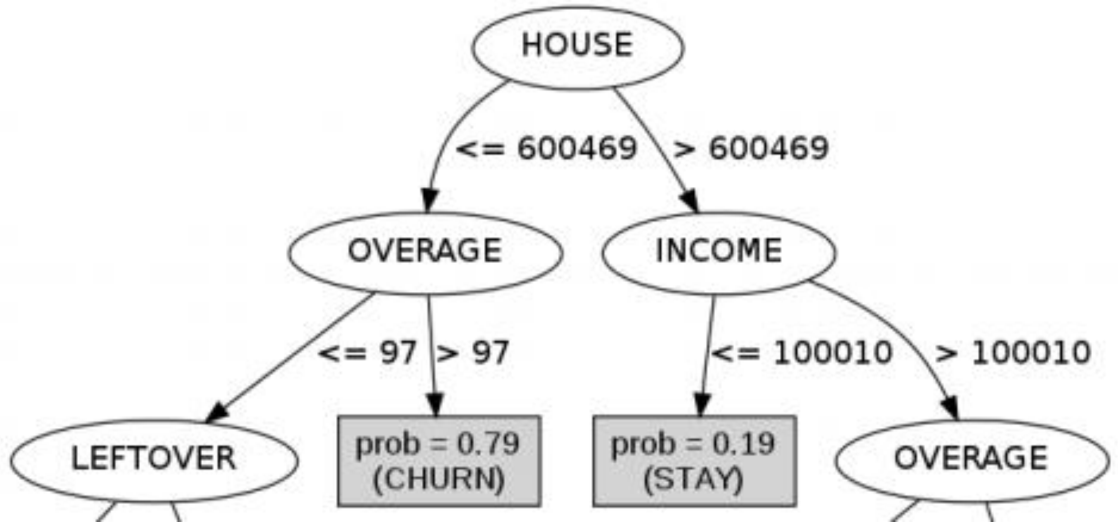
mpg values: bad good

Input	Value	Distribution	Info Gain
cylinders	< 5	🟦🟥	0.48268
	>= 5	🟦🟥	
displacement	< 198	🟦🟥	0.428205
	>= 198	🟦🟥	
horsepower	< 94	🟦🟥	0.48268
	>= 94	🟦🟥	
weight	< 2789	🟦🟥	0.379471
	>= 2789	🟦🟥	
acceleration	< 18.2	🟦🟥	0.159982
	>= 18.2	🟥	
modelyear	< 81	🟦🟥	0.319193
	>= 81	🟥	
maker	america	🟦🟥	0.0437265
	asia	🟦🟥	
	europe	🟦🟥	

	Num Errors	Set Size	Percent Wrong
Training Set	1	40	2.50
Test Set	53	352	15.06

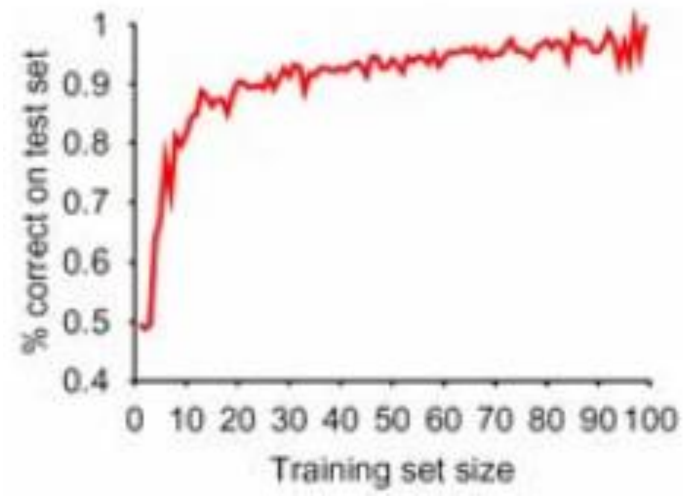
Output probabilities

- How to do better than predicting majority class?
- Estimate probabilities from the relevant examples at each node
- Can use smoothing to improve estimates (e.g., Laplace smoothing)



Scaling up

- More data is almost always better



- Scaling up with standard recursive algorithms can be hard
- New algorithms make single pass through data
  - E.g. Very Fast Decision Trees (VFTD)