Inductive Learning and Decision Trees

Doug Downey
EECS 349

with slides from Pedro Domingos, Bryan Pardo

Outline

- Announcements
  - Homework #1 was assigned yesterday
- Inductive learning
- Decision Trees

Outline

- Announcements
  - Homework #1 was assigned yesterday
- **Inductive learning**
- Decision Trees

Machine Learning tasks

- **Tasks** clearly state inputs and outputs:
  - Predicting the stock market based on past price data
    - Input: A ticker symbol and a date
    - Output: Will the close be higher or lower on the date? (classification)
      Or: what will the price change be? (regression)
  - Predicting outcomes of sporting events
    - Input: A game (two opponents, a date)
    - Output: which team will win (classification)
- On the other hand, these are *not* tasks:
  - "Studying the relationship between weather and sports game outcomes."
  - "Applying neural networks to natural language processing."

Instances

- E.g. Four Days, in terms of weather:

Sky	Temp	Humid	Wind	Forecast
sunny	warm	normal	strong	same
sunny	warm	high	strong	same
rainy	cold	high	strong	change
sunny	warm	high	strong	change

Functions

- "Days on which Anne agrees to get lunch with me"

INPUT

OUTPUT

Sky	Temp	Humid	Wind	Forecast	f(x)
sunny	warm	normal	strong	same	1
sunny	warm	high	strong	same	1
rainy	cold	high	strong	change	0
sunny	warm	high	strong	change	1

Inductive Learning!

- **Predict** the output for a new instance (generalize!)

INPUT

OUTPUT

Sky	Temp	Humid	Wind	Forecast	f(x)
sunny	warm	normal	strong	same	1
sunny	warm	high	strong	same	1
rainy	cold	high	strong	change	0
sunny	warm	high	strong	change	1
rainy	**warm**	**high**	**strong**	**change**	**?**

General Inductive Learning Task

DEFINE:

- Set X of Instances (of n-tuples $\mathbf{x} = \langle x_1, ..., x_n \rangle$)
  - E.g., days decribed by *attributes* (or *features*):

    Sky, Temp, Humidity, Wind, Forecast
- Target function $f : X \rightarrow Y$, e.g.:
  - GoesToLunch $X \rightarrow Y = \{0, 1\}$
  - ResponseToLunch $X \rightarrow Y$ = {"No," "Yes," "How about tomorrow?"}
  - ProbabilityOfLunch $X \rightarrow Y = [0, 1]$

GIVEN:

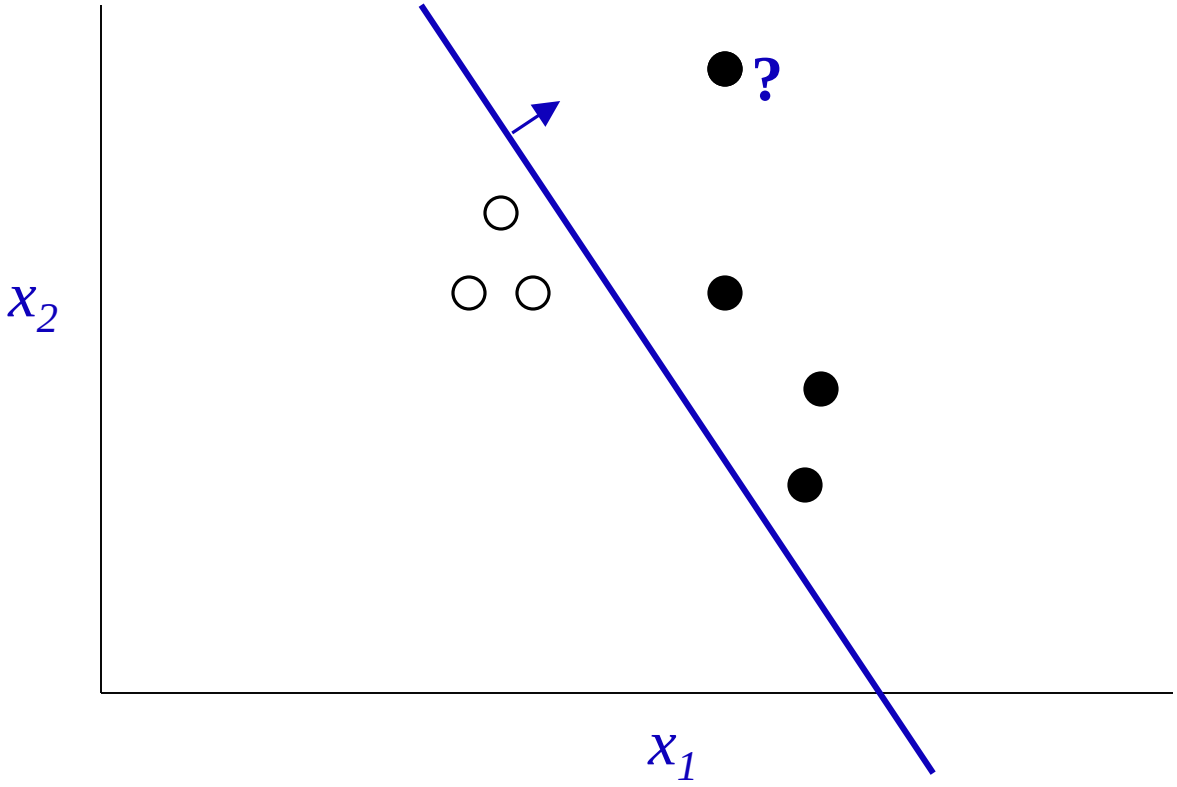
- *Training examples* D
  - examples of the target function: $\langle \mathbf{x}, f(\mathbf{x}) \rangle$

FIND:

- A hypothesis h such that $h(\mathbf{x})$ approximates $f(\mathbf{x})$.

Example w/ continuous attributes

Learn function from $\mathbf{x} = (x_1, \ldots, x_d)$ to $f(\mathbf{x}) \in \{0, 1\}$
given labeled examples $(\mathbf{x}, f(\mathbf{x}))$



Hypothesis Spaces

- Hypothesis space *H* is a **subset** of all $f: X \rightarrow Y$ e.g.:
  - Linear separators
  - Conjunctions of constraints on attributes (humidity must be low, and outlook != rain)
  - Etc.

- In machine learning, we restrict ourselves to *H*

Examples

- Credit Risk Analysis
  - ***X*:** Properties of customer and proposed purchase
  - $f(\mathbf{x})$**:** Approve (1) or Disapprove (0)
- Disease Diagnosis
  - ***X*:** Properties of patient (symptoms, lab tests)
  - $f(\mathbf{x})$**:** Disease (if any)
- Face Recognition
  - ***X*:** Bitmap image
  - $f(\mathbf{x})$**:**Name of person
- Automatic Steering
  - ***X*:** Bitmap picture of road surface in front of car
  - $f(\mathbf{x})$**:** Degrees to turn the steering wheel

When to use?

- Inductive Learning is appropriate for building a face recognizer
- It is not appropriate for building a calculator
  - You'd just write a calculator program

- Question:
  What general characteristics make a problem suitable for inductive learning?

Think/Pair/Share

What general characteristics make a problem suitable for inductive learning?

Think

Start

End

Think/Pair/Share

What general characteristics make a problem suitable for inductive learning?

Pair

Start

End

Think/Pair/Share

What general characteristics make a problem suitable for inductive learning?

Share

Appropriate applications

- Situations in which:
  - There is no human expert
  - Humans can perform the task but can't describe how
  - The desired function changes frequently
  - Each user needs a customized *f*

Outline

- Announcements
  - Homework #1
- Inductive learning
- **Decision Trees**

Why Decision Trees?

- Simple inductive learning approach
  - Training procedure is easy to understand
  - Models are easy to understand

- Popular
  - The most popular learning method, according to surveys [Domingos, 2016]

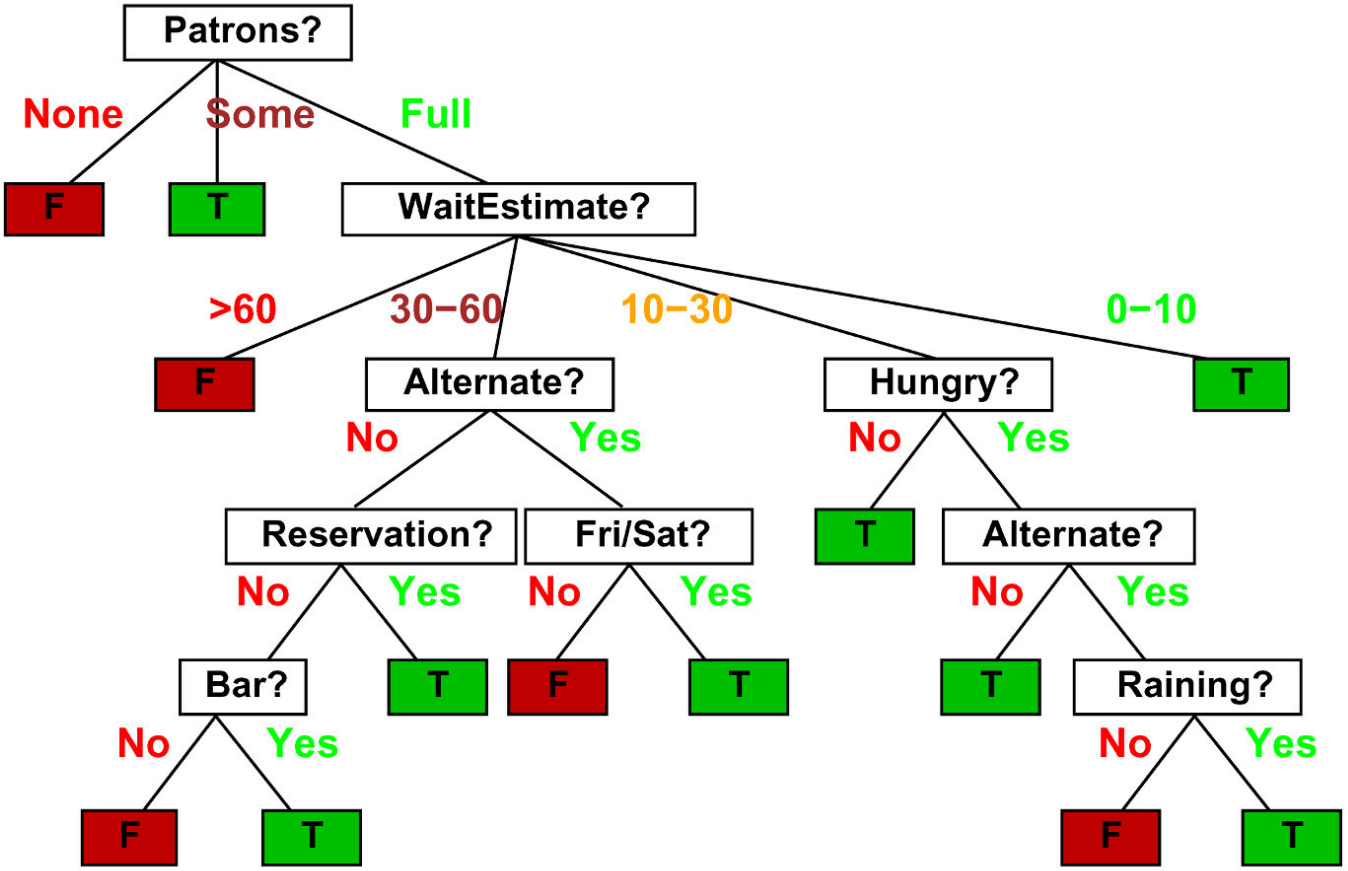
Task: Will I wait for a table?

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*

Classification of examples is positive (T) or negative (F)

Decision Trees!

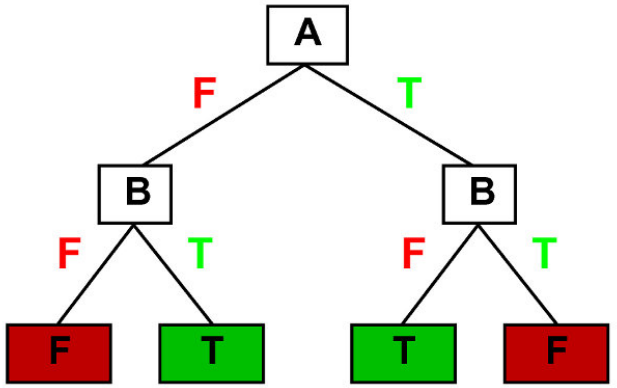
One possible representation for hypotheses
E.g., here is the "true" tree for deciding whether to wait:



Expressiveness of D-Trees

Decision trees can express any function of the input attributes.
E.g., for Boolean functions, truth table row $\rightarrow$ path to leaf:

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

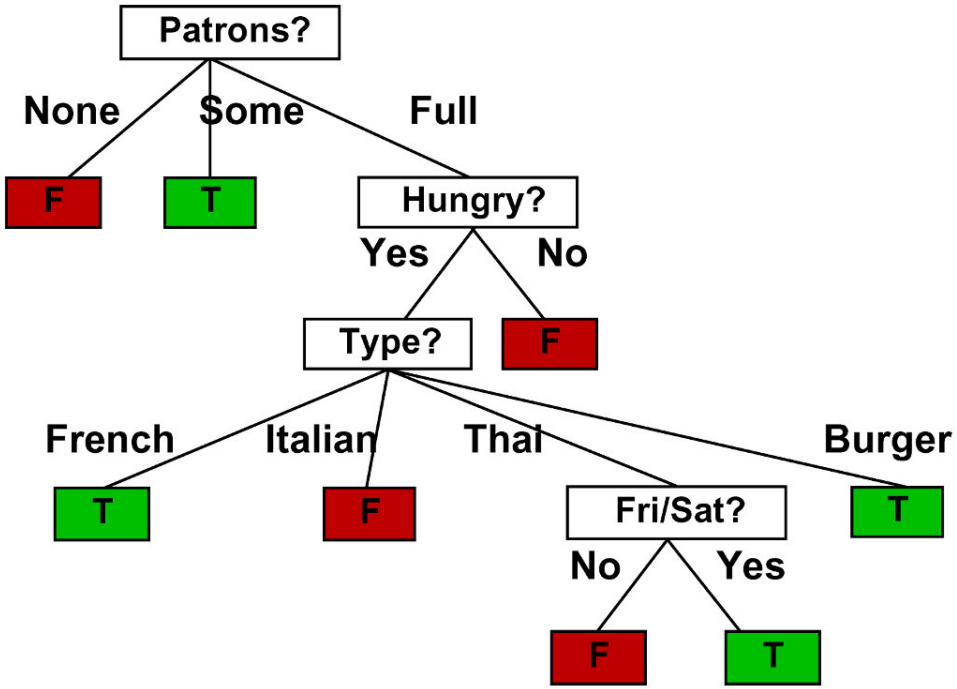


Trivially, there is a consistent decision tree for any training set w/ one path to leaf for each example (unless f nondeterministic in x) but it probably won't generalize to new examples

Prefer to find more **compact** decision trees

A learned decision tree

Decision tree learned from the 12 examples:



Substantially simpler than "true" tree—a more complex hypothesis isn't justified by small amount of data

Inductive Bias

- To learn, we **must** prefer some functions to others
  - **Selection bias**
    - use a **restricted** hypothesis space, e.g.:
      - linear separators
      - 2-level decision trees
  - **Preference bias**
    - use the whole concept space, but state a **preference** over concepts, e.g.:
      - *Lowest-degree* polynomial that separates the data
      - *shortest* decision tree that fits the data

Decision Tree Learning (ID3*)

Goal: Find a (small) tree consistent with *examples*

Function ID3(*examples*, *default*) **returns** a tree

if *examples* is empty

return tree(*default*)

else if all *examples* have same classification **or** no non-trivial splits are possible:

return tree(MODE(*examples*)))

Returns most frequent class label in examples

else:

best ← CHOOSE-ATTRIBUTE(*examples*)

t ← new tree with root test *best*

for each $value_i$ of best:

$examples_i$ ← {elements of *examples* with *best* = $value_i$}

subtree ← ID3(*examplesi*, MODE(*examples*)}

add branch to *t* with label $value_i$ and subtree *subtree*

return *t*

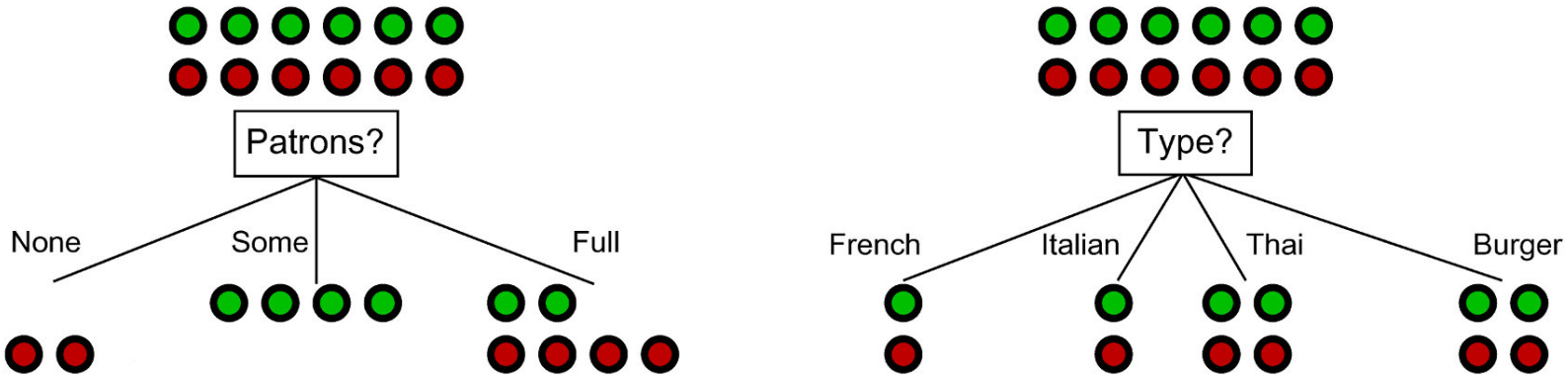
* Our algorithm's termination conditions differ in small ways from the original published ID3

Recap

- Inductive learning
  - Goal: generate a **hypothesis** – a function from **instances** described by **attributes** to an output – using **training examples**.
  - Requires **inductive bias**
    - a restricted **hypothesis space**, or preferences over hypotheses.
- Decision Trees
  - Simple representation of hypotheses, recursive learning algorithm
  - Prefer smaller trees!

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—gives **information** about the classification

Think/Pair/Share

How should we choose which attribute to split on next?

Think

Start

End

Think/Pair/Share

How should we choose which attribute to split on next?

Pair

Start

End

Think/Pair/Share

How should we choose which attribute to split on next?

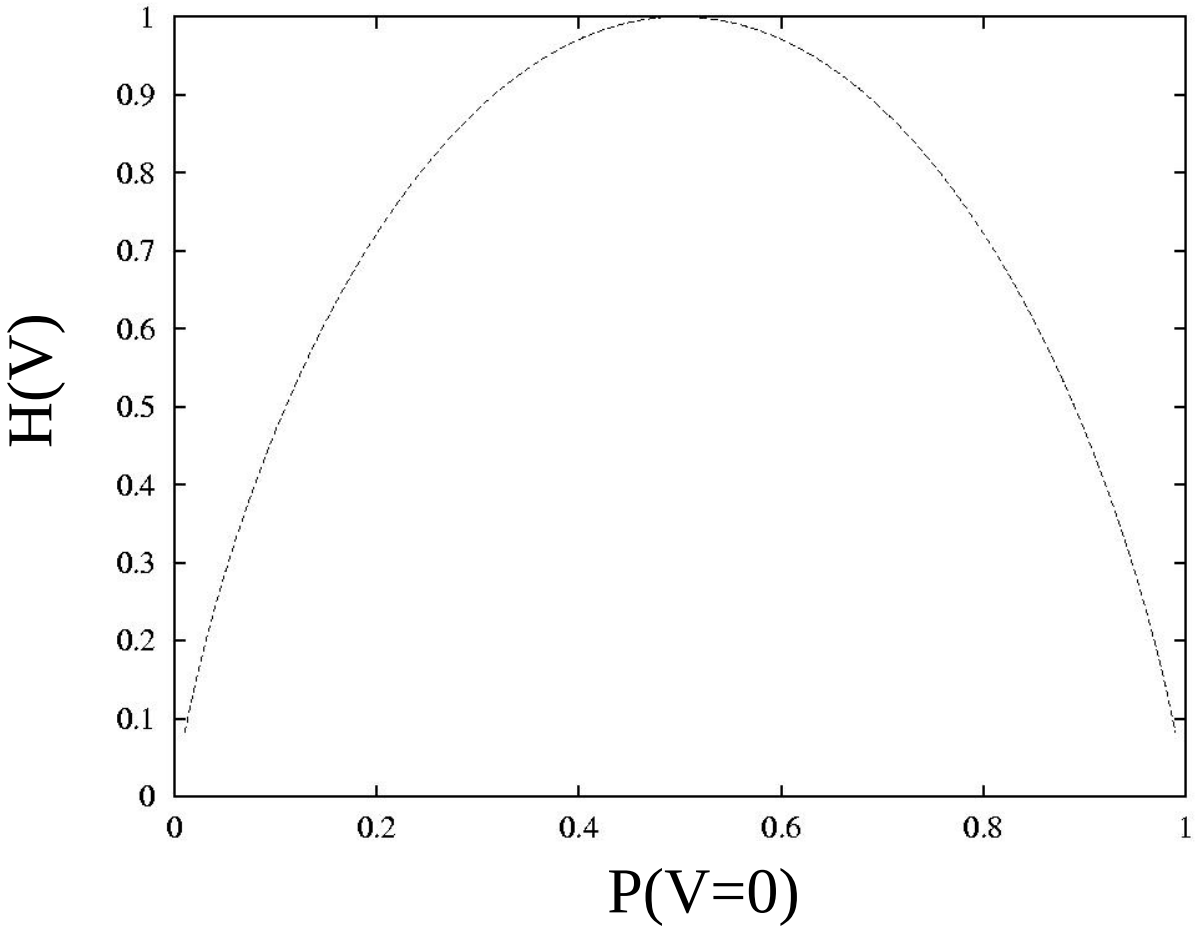
Share

Information

- Brief sojourn into information theory
  - (on board)

Entropy

The entropy H(V) of a Boolean random variable V as the probability of V = 0 varies from 0 to 1



Using Information

- The key question: how much information, on average, will I gain about the class by doing the split?
  - Choose attribute x_i that maximizes this expected value

- $InfoGain(x_i) = H_{prior} - \sum_v P(x_i = v) H(y|x_i = v)$
- Since H_{prior} is constant w.r.t. x_i, we can just choose attribute with minimum $\sum_v P(x_i = v) H(y|x_i = v)$

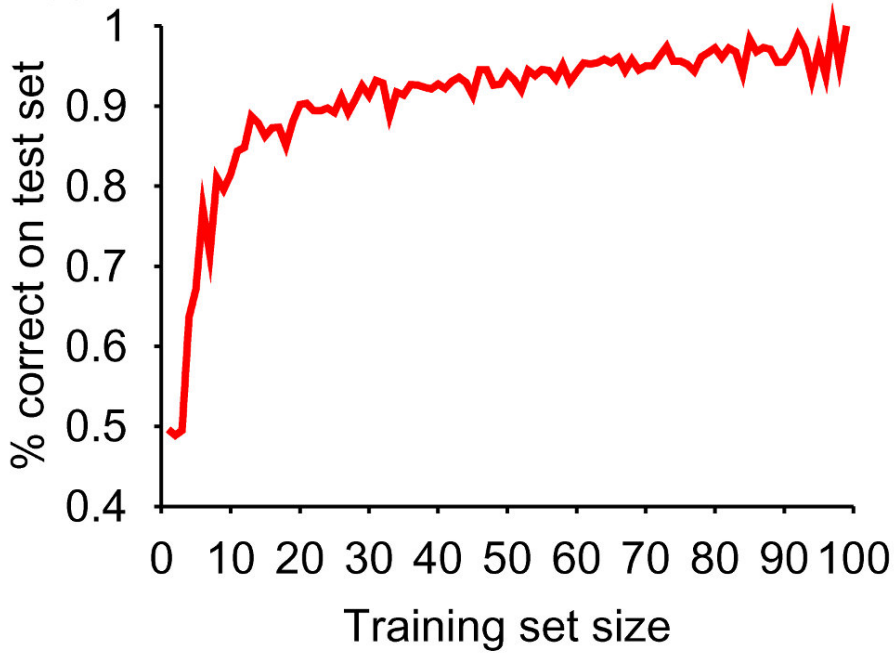
Measuring Performance

How do we know that $h \approx f$? (Hume's **Problem of Induction**)

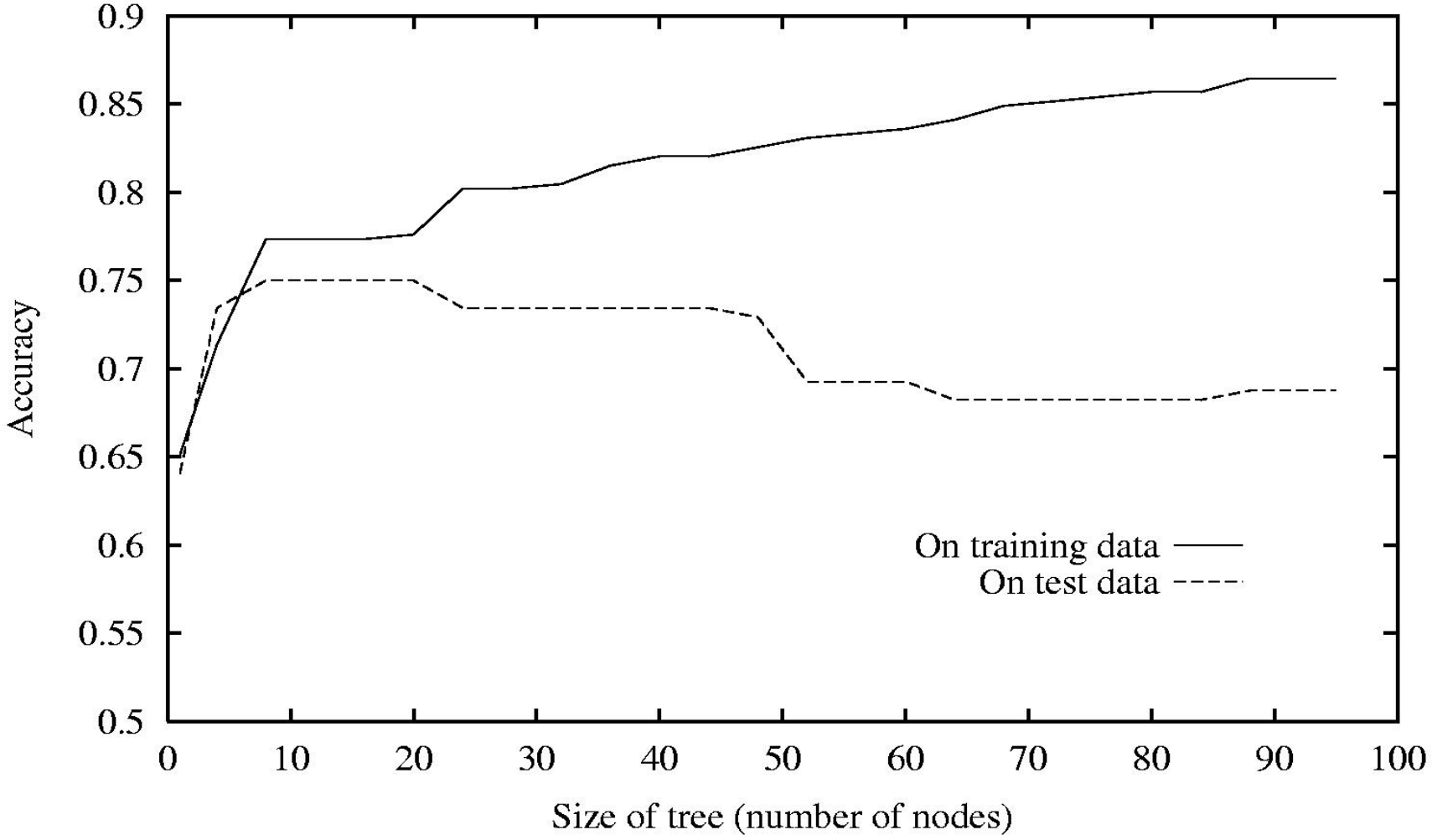
1) Use theorems of computational/statistical learning theory

2) Try h on a new test set of examples
(use **same distribution over example space** as training set)

Learning curve = % correct on test set as a function of training set size



Overfitting



Overfitting is due to “noise”

- Sources of noise:
  - Erroneous training data
    - concept variable incorrect (annotator error)
    - Attributes mis-measured
  - More significant:
    - Irrelevant attributes
    - Target function not realizable in attributes

Irrelevant attributes

- If many attributes are noisy, information gains can be spurious, e.g.:
  - 20 noisy attributes
  - 10 training examples
  - Expected # of different depth-3 trees that split the training data perfectly using *only* noisy attributes: **13.4**

Not realizable

- In general:
  - We can rarely measure well enough for **perfect** prediction
  - => Target function is not uniquely determined by attribute values
  - Target outputs appear to be "noisy"
    - Same attribute vector may yield distinct output values

Not realizable: Example

Humidity	EnjoySport
0.90	0
0.87	I
0.80	0
0.75	0
0.70	I
0.69	I
0.65	I
0.63	I

Decent hypothesis:

Humidity > 0.70 → No

Otherwise → Yes

Overfit hypothesis:

Humidity > 0.89 → No

Humidity > 0.80

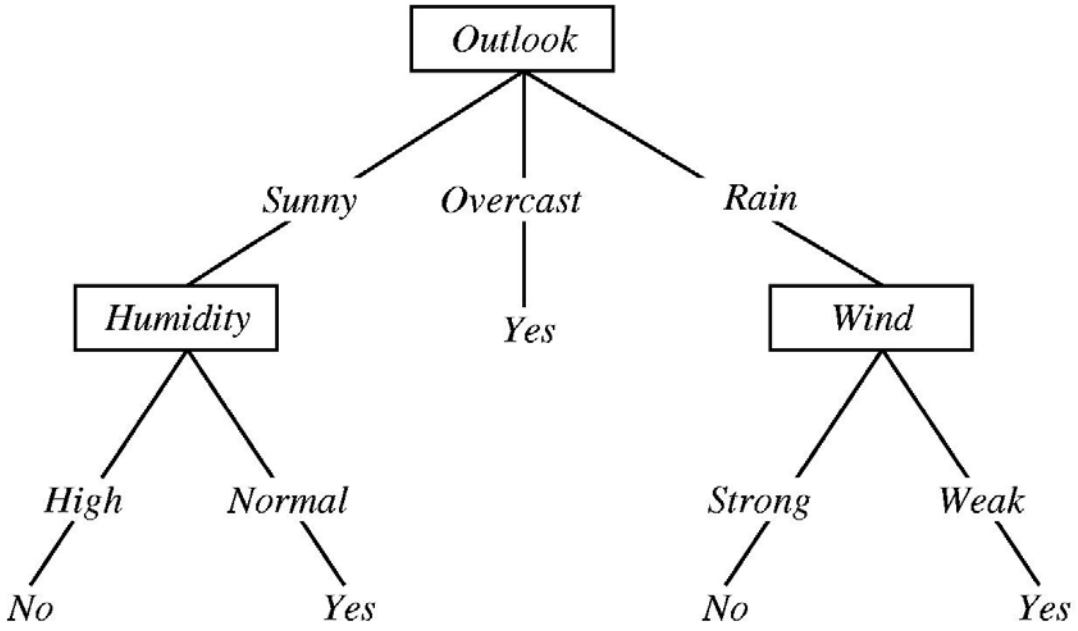
^ Humidity <= 0.89 → Yes

Humidity > 0.70

^ Humidity <= 0.80 → No

Humidity <= 0.70 → Yes

Overfitting in Decision Trees



Consider adding a noisy training example:

Sunny, Hot, Normal, Strong, PlayTennis=No

What effect on tree?

Avoiding Overfitting

- Approaches
  - Stop splitting when information gain is low or when split is not statistically significant.
  - Grow full tree and then **prune** it when done

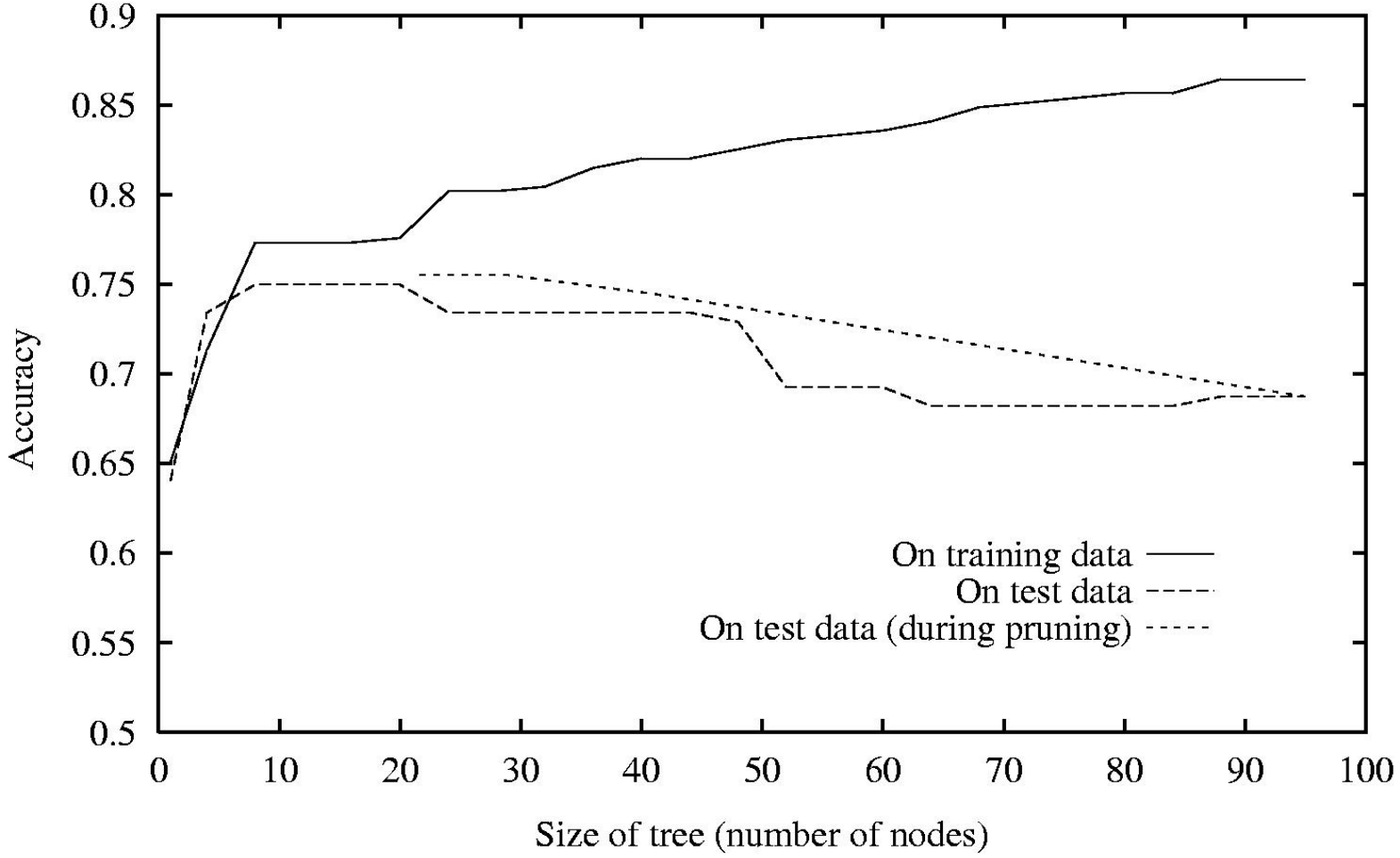
Reduced-Error Pruning

Split data into *training* and *validation* set

Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

Effect of Reduced Error Pruning



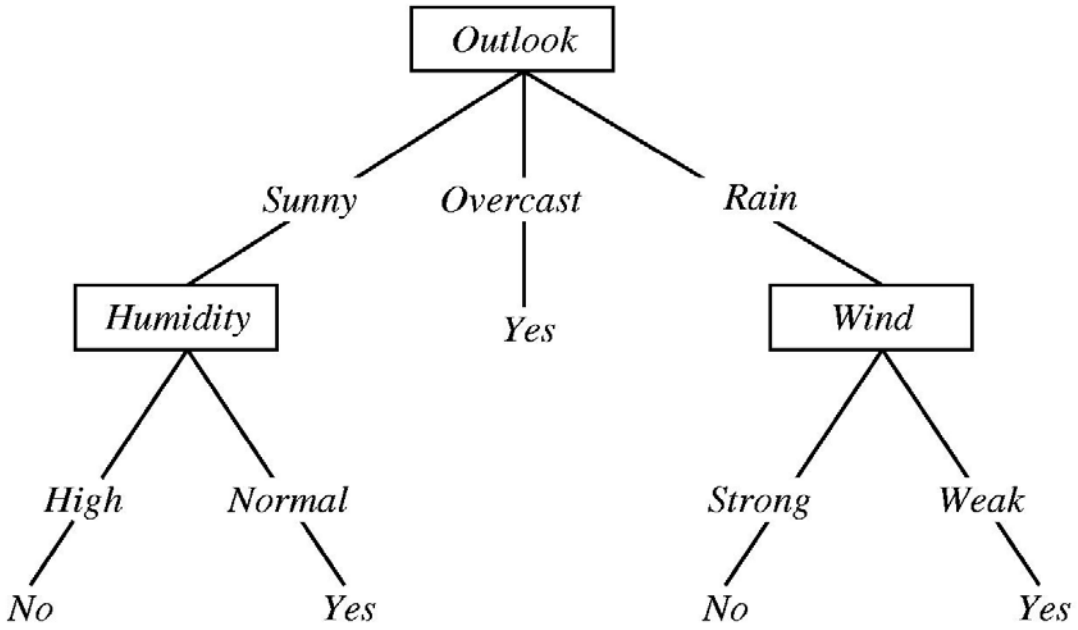
C4.5 Algorithm

- Builds a decision tree from labeled training data
- Generalizes simple "ID3" tree by
  - Prunes tree after building to improve generality
  - Allows missing attributes in examples
  - Allowing continuous-valued attributes

Rule post pruning

- Used in C4.5
- Steps
  1. Build the decision tree
  2. Convert it to a set of logical rules
  3. Prune each rule independently
  4. Sort rules into desired sequence for use

Converting A Tree to Rules



IF $(Outlook = Sunny)\ AND\ (Humidity = High)$
THEN $PlayTennis = No$

IF $(Outlook = Sunny)\ AND\ (Humidity = Normal)$
THEN $PlayTennis = Yes$

...

Other Odds and Ends

- Unknown Attribute Values?

Unknown Attribute Values

What if some examples are missing values of A?

Use training example anyway, sort through tree

- If node n tests A, assign most common value of A among other examples sorted to node n
- Assign most common value of A among other examples with same target value
- Assign probability p_i to each possible value v_i of A
  Assign fraction p_i of example to each descendant in tree

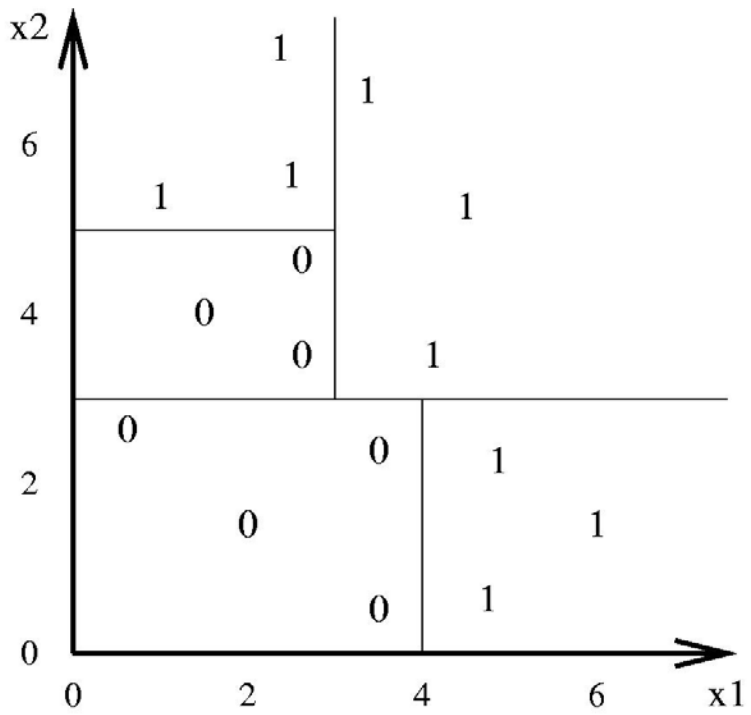
Classify new examples in same fashion

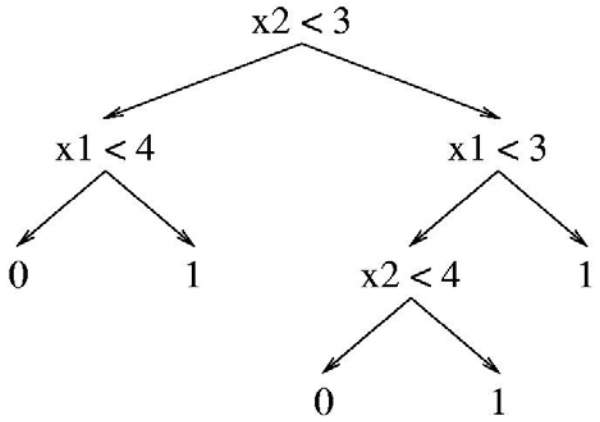
Odds and Ends

- Unknown Attribute Values?
- Continuous Attributes?

Decision Tree Boundaries

Decision trees divide the feature space into axis-parallel rectangles, and label each rectangle with one of the K classes.

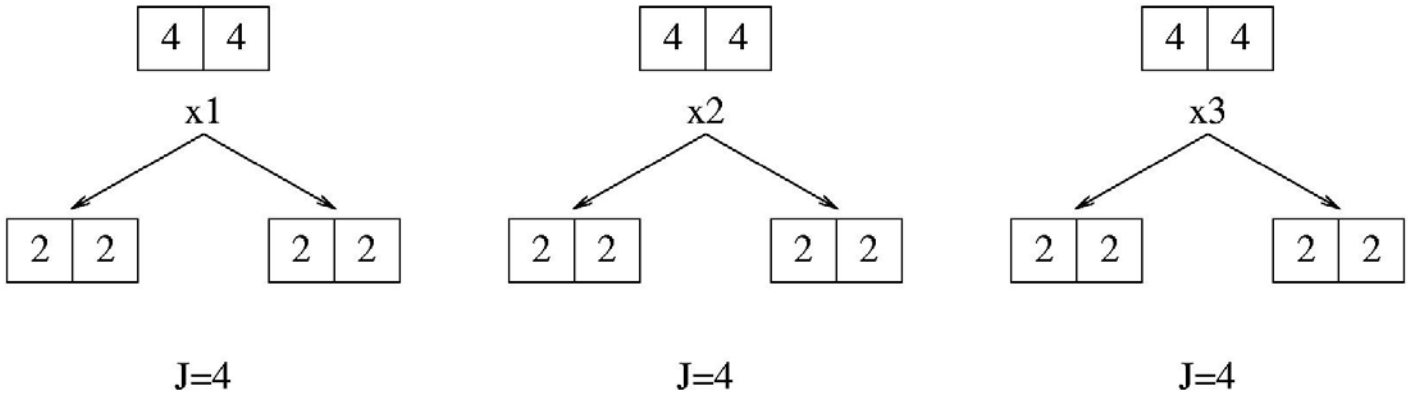




Learning Parity with Noise

When learning exclusive-or (2-bit parity), all splits look equally good. If extra random boolean features are included, they also look equally good. Hence, decision tree algorithms cannot distinguish random noisy features from parity features.

x_1	x_2	x_3	y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0



Decision Trees Bias

- How to solve 2-bit parity:
  - Two step "look-ahead," or
  - Split on *pairs* of attributes at once

- For *k*-bit parity, why not just do *k*-step look ahead? Or split on *k* attribute values?

=>*Parity functions are among the "victims" of the decision tree's inductive bias.*

Now we have choices

- Re-split continuous attributes?
- Handling unknown variables?
- Prune or not?
- Stopping criteria?
- Split selection criteria?
- Use look-ahead?

- In homework #2: one choice for each
- In practice, how to decide? An instance of *Model Selection*
  - In general, we could also select an H other than decision trees

Think/Pair/Share

We can do model selection using a 70% train, 30% validation split of our data. But can we do better?

Think

Start

End

Think/Pair/Share

We can do model selection using a 70% train, 30% validation split of our data. But can we do better?

Pair

Start

End

Think/Pair/Share

We can do model selection using a 70% train, 30% validation split of our data. But can we do better?

Share

10-fold Cross-Validation

- On board

Take away about decision trees

- Used as classifiers
- Supervised learning algorithms (ID3, C4.5)
- Good for situations where
  - Inputs, outputs are discrete
  - Interpretability is important
  - "We think the true function is a small tree"

Readings

- Decision Trees:
  - Induction of decision trees, Ross Quinlan (1986) (covers ID3)
    - https://link.springer.com/article/10.1007%2FBF00116251
      (may need to be on campus to access)
  - C4.5: Programs for Machine Learning (2014) (covers C4.5)
    https://books.google.com/books?hl=en&lr=&id=b3ujBQAAQBAJ&oi=fnd&pg=PP1&dq=c4.5&ots=sPanSTEtC4&sig=c2Np0fBu37b-Ie-dVUyhulPJsv4#v=onepage&q=c4.5&f=false
- Overfitting in Decision Trees
  - http://cse-wiki.unl.edu/wiki/index.php/Decision_Trees,_Overfitting,_and_Occam's_Razor
- Cross-Validation
  - https://en.wikipedia.org/wiki/Cross-validation_(statistics)