

A decorative graphic consisting of a thin yellow circle and a horizontal bar with a yellow-to-white gradient. A large black left square bracket is on the left side of the bar, and a large yellow right square bracket is on the right side.

Learning from Observations

Chapter 18
Sections 1-3

[Outline]

- Learning
- Hypothesis Spaces
- Learning Algorithms
 - K Nearest Neighbors
 - Decision Trees
 - Naïve Bayes
 - Not really described in the text well
- Training and Testing

[What is Learning]

- Memorizing something
- Learning facts through observation and exploration
- Generalizing a concept from experience

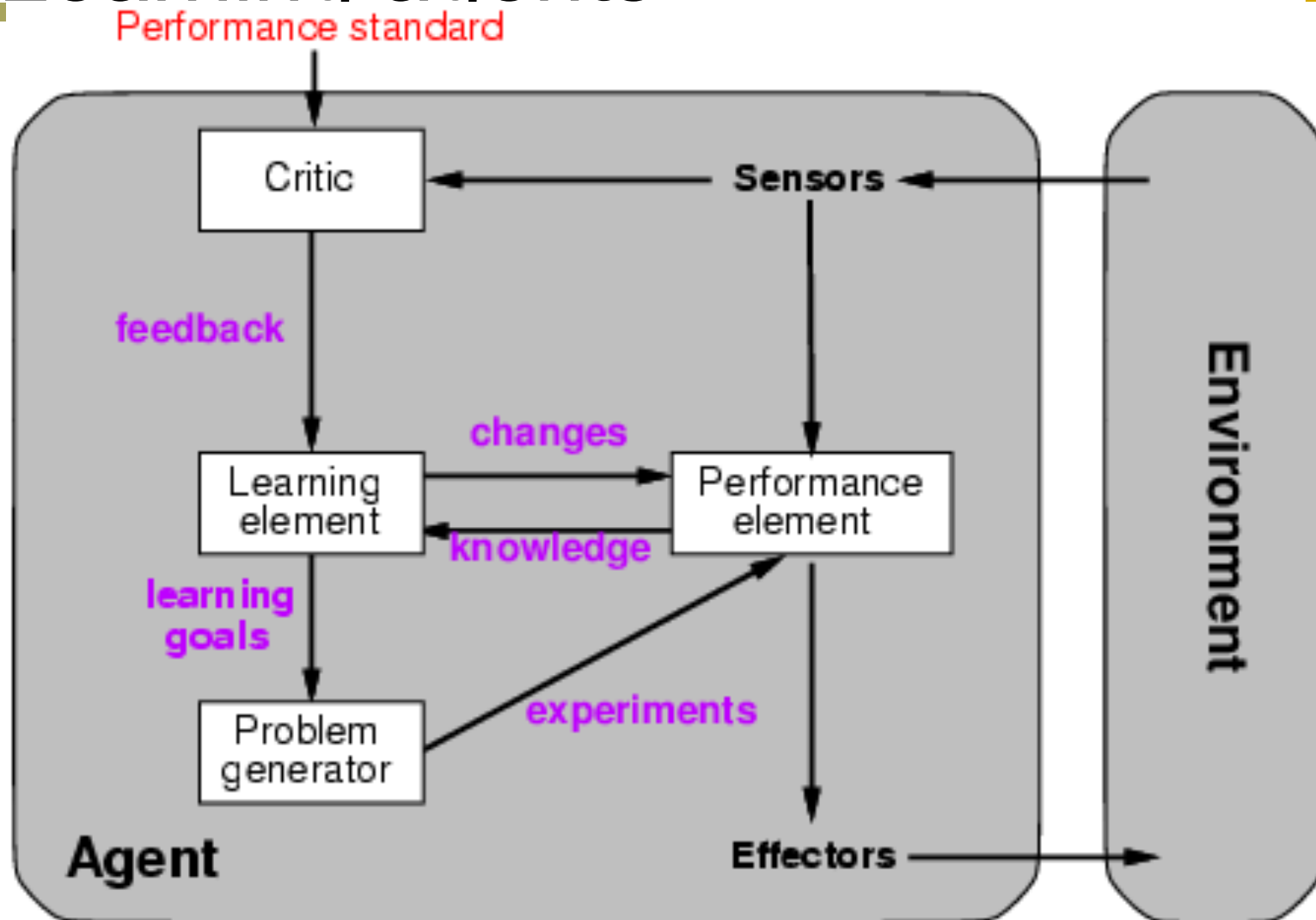
“Learning denotes changes in the system that are **adaptive** in the sense that they enable the system to do the task or tasks drawn from the **same population** more efficiently and more effectively the next time” – Herb Simon

[Why is it necessary?]

Three reasons:

- Unknown environment – need to deploy an agent in an unfamiliar territory
- Save labor – we may not have the resources to encode knowledge
- Can't explicitly encode knowledge – may lack the ability to articulate necessary knowledge.

[Learning agents]



[Learning element]

- Design of a learning element is affected by
 - Which components of the performance element are to be learned
 - What feedback is available to learn these components
 - What representation is used for the components
- Type of feedback:
 - **Supervised learning**: correct answers for each example
 - **Unsupervised learning**: correct answers not given
 - **Reinforcement learning**: occasional rewards

Induction

- Making predictions about the future based on the past.

If asked why we believe the sun will rise tomorrow, we shall naturally answer, “Because it has always risen every day.” We have a firm belief that it will rise in the future, because it has risen in the past. – Bertrand Russell

- Is induction sound? Why believe that the future will look similar to the past?

[Inductive learning]

- Simplest form: learn a function from examples

f is the target function

An example is a pair $(x, f(x))$

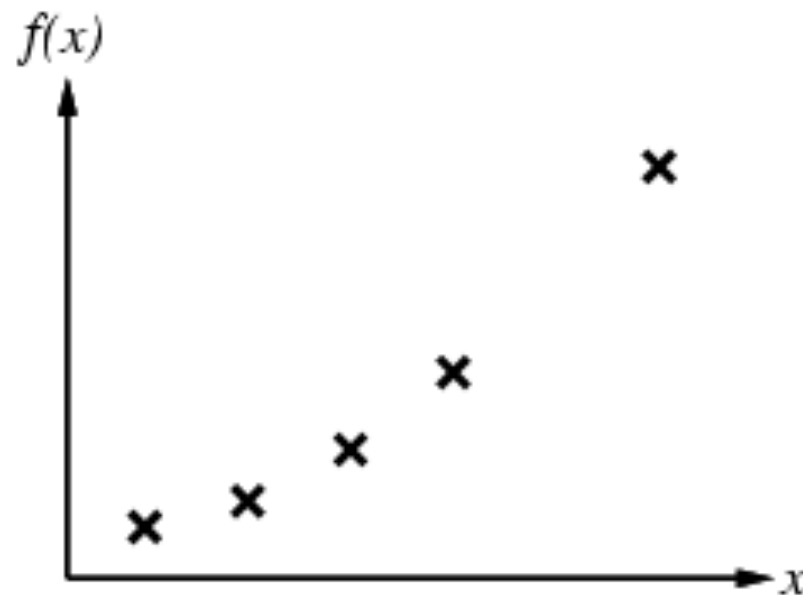
Problem: find a hypothesis h
such that $h \approx f$
given a training set of examples

This is a highly simplified model of real learning:

- Ignores prior knowledge
- Assumes examples are given

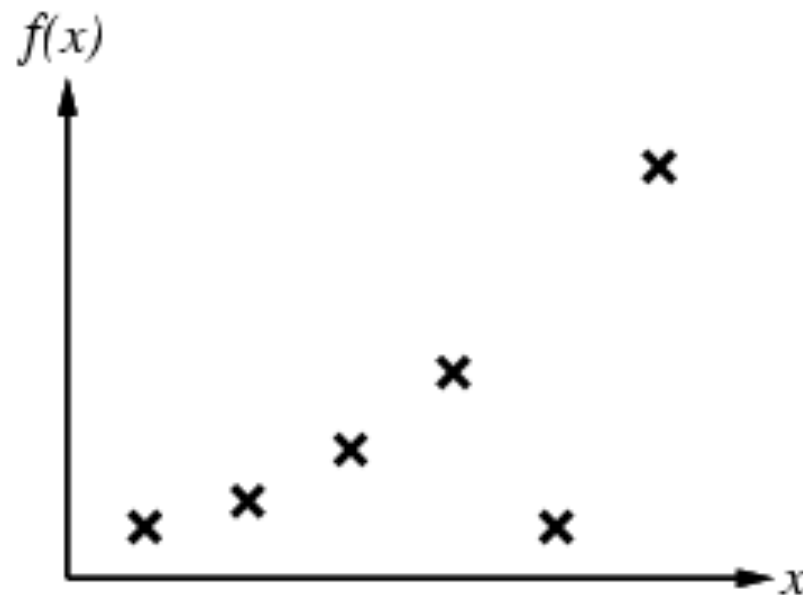
[Inductive learning method]

- Memorization
- Noise
 - Unreliable function
 - Unreliable sensors



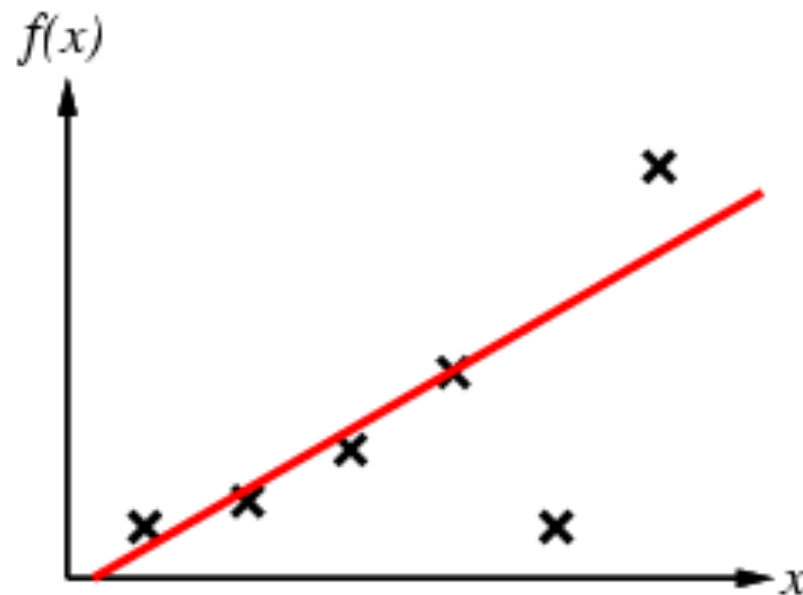
Inductive learning method

- Construct/adjust h to agree with f on training set
- (h is **consistent** if it agrees with f on all examples)
- E.g., curve fitting:



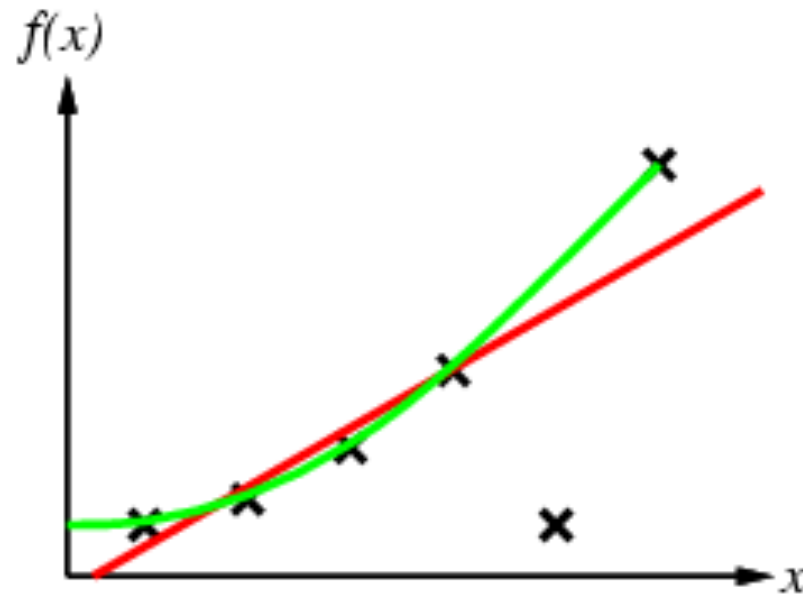
[Inductive learning method]

- Construct/adjust h to agree with f on training set
- (h is **consistent** if it agrees with f on all examples)
- E.g., curve fitting:



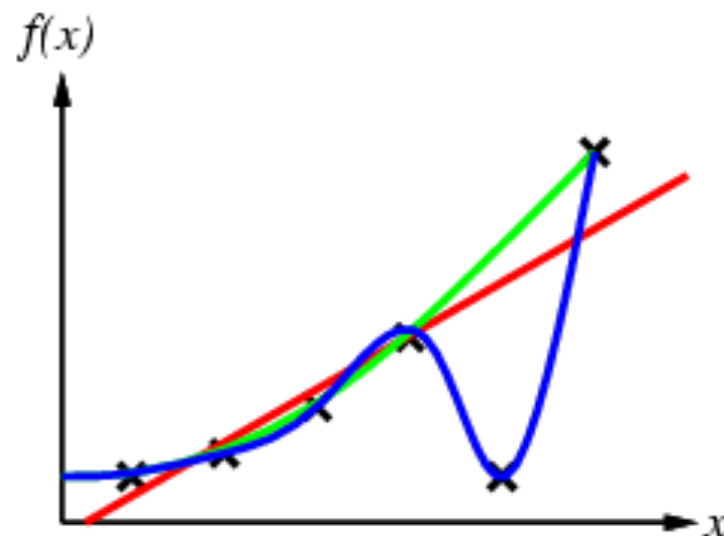
Inductive learning method

- Construct/adjust h to agree with f on training set
- (h is **consistent** if it agrees with f on all examples)
- E.g., curve fitting:



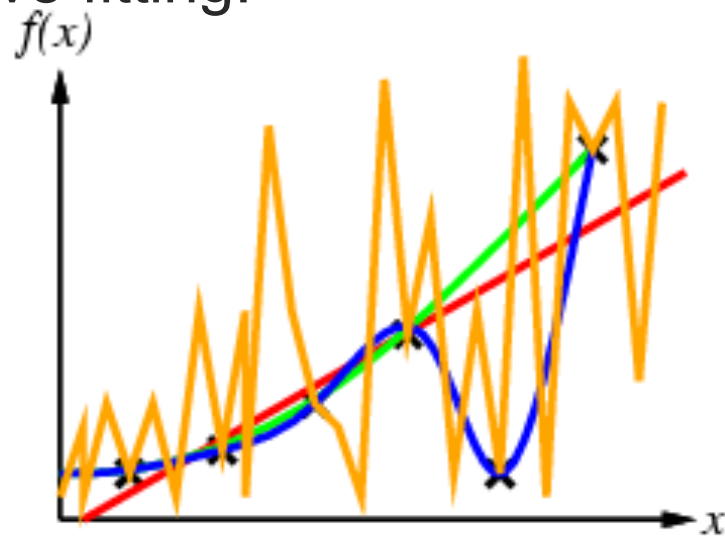
[Inductive learning method]

- Construct/adjust h to agree with f on training set
- (h is **consistent** if it agrees with f on all examples)
- E.g., curve fitting:



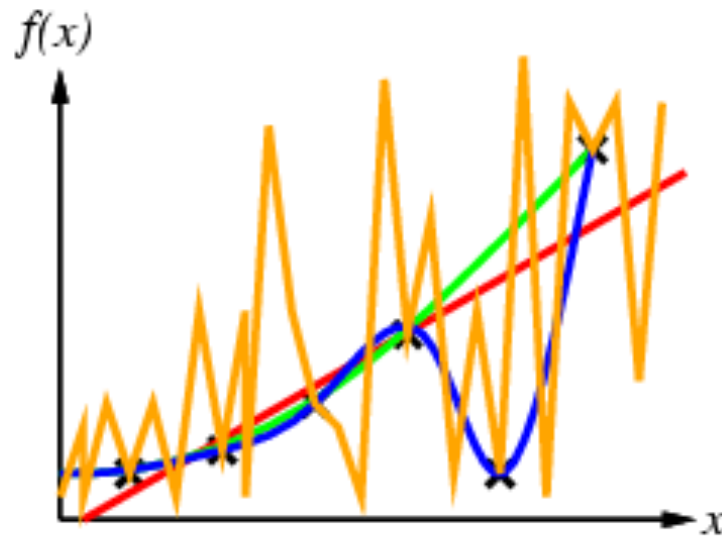
Inductive learning method

- Construct/adjust h to agree with f on training set
- (h is **consistent** if it agrees with f on all examples)
- E.g., curve fitting:



Inductive learning method

- Construct/adjust h to agree with f on training set
- (h is **consistent** if it agrees with f on all examples)
- E.g., curve fitting:



- Ockham's razor: prefer the simplest hypothesis consistent with data

[An application: Ad blocking]

The screenshot shows the CNN.com homepage. At the top left is the CNN.com logo. To its right is the text 'MEMBER SERVICES'. Further right is 'International Edition | Netscape' with a Netscape logo. Below this is a blue banner for 'THE SOUTH BEACH DIET™ ONLINE' with the text 'GET READY TO LOOK FABULOUS!' and a photo of a smiling woman. Below the banner is a search bar with 'SEARCH' on the left, 'The Web' and 'CNN.com' as options, and a 'Search' button. To the right of the search bar is 'Enhanced by: Google'. Below the search bar is a red 'Home Page' button and a list of categories: World, U.S., Weather, Business at CNNMoney, Sports at SI.com, Politics, Law, Technology, Science & Space, Health, Entertainment, Travel, Education, and Special Reports. To the right of the categories is the text 'Updated: 12:30 a.m. EST (05:30 GMT) March 15, 2004'. Below this is a large headline 'Israeli helicopters fire at Gaza targets' with a photo of a man in a dark jacket. To the right of the headline is a section 'MORE TOP STORIES' with a 'MOST POPULAR' button and a list of links: 'Terror-scarred Spain votes in Socialists', 'Son: Murder suspect the 'best dad'', 'CNN/Money: Microsoft facing European sanctions', 'Ashcroft discharged from hospital', 'SI.com: March Madness field unveiled', 'March Madness', '1794 silver dollar may be the country's first', 'Passion' stays on top at box office, and 'AMERICA VOTES 2004' Complete election coverage. At the bottom right are buttons for 'CNN RADIO', 'VIDEO', and 'MORE VIDEO'.

CNN.com

MEMBER SERVICES

International Edition | Netscape

MAKE CNN.com YOUR HOME PAGE

THE SOUTH BEACH DIET™ ONLINE

GET READY TO LOOK FABULOUS!

3 easy phases • 24h online support • Results, Results!

SEARCH

The Web CNN.com

Search

Enhanced by: Google

Home Page

World

U.S.

Weather

Business at CNNMoney

Sports at SI.com

Politics

Law

Technology

Science & Space

Health

Entertainment

Travel

Education

Special Reports

Updated: 12:30 a.m. EST (05:30 GMT) March 15, 2004

Israeli helicopters fire at Gaza targets

MORE TOP STORIES

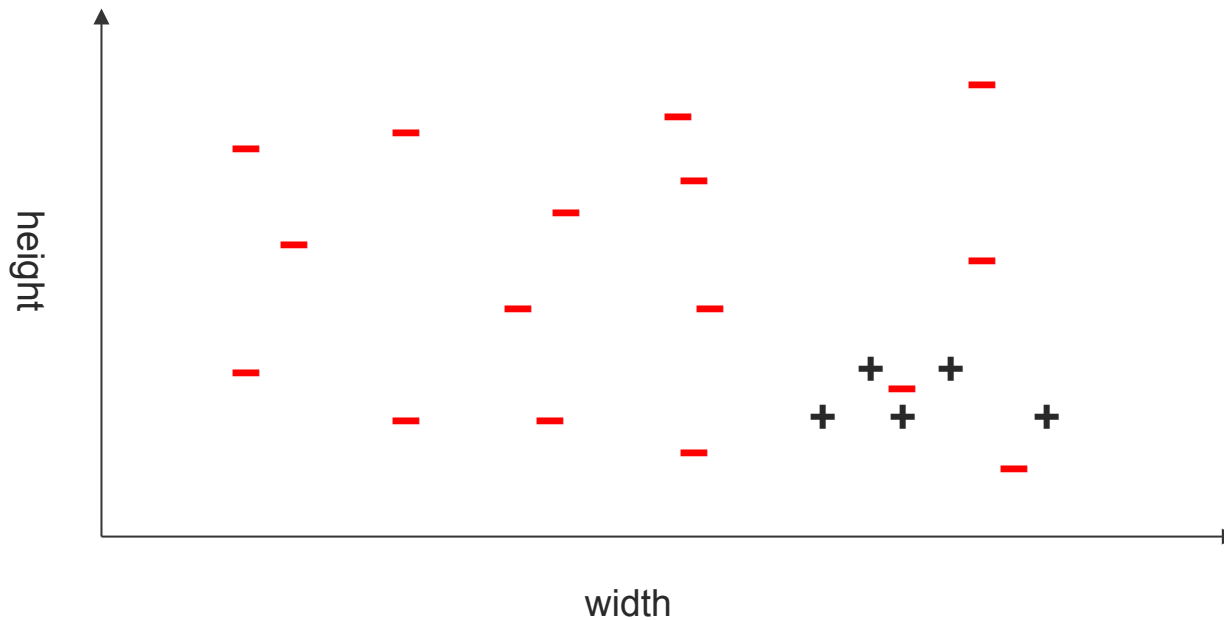
MOST POPULAR

- Terror-scarred Spain votes in Socialists | Video | Gallery
- Son: Murder suspect the 'best dad' | Video | Gallery
- CNN/Money: Microsoft facing European sanctions
- Ashcroft discharged from hospital
- SI.com: March Madness field unveiled | March Madness
- 1794 silver dollar may be the country's first
- 'Passion' stays on top at box office
- AMERICA VOTES 2004 Complete election coverage

CNN RADIO VIDEO MORE VIDEO

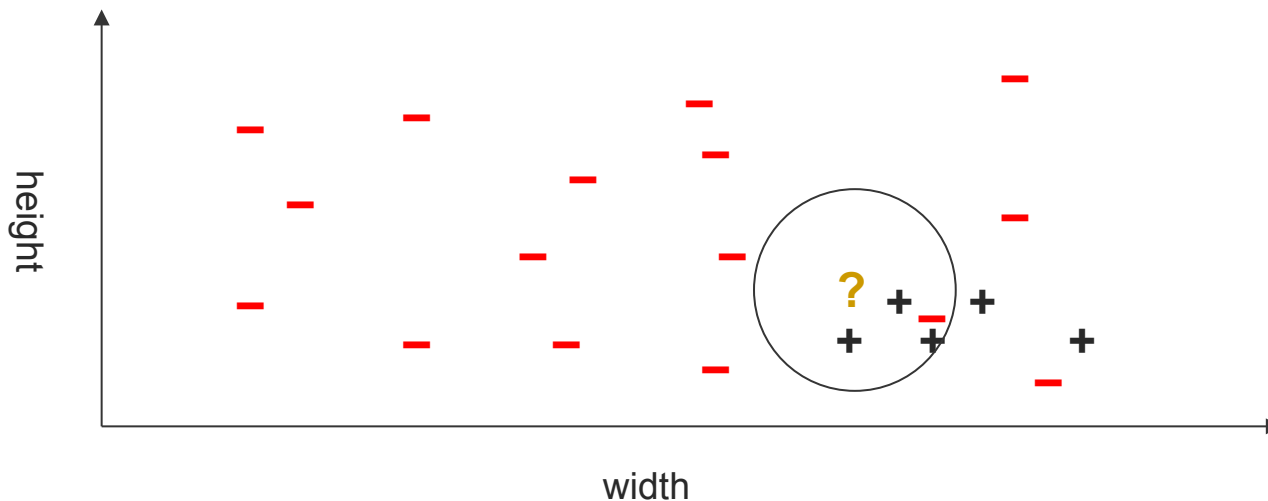
[Learning Ad blocking]

- Width and height of image
- Binary Classification: Ad or \neg Ad?



Nearest Neighbor

- A type of instance based learning
- Remember all of the past instances
- Use the nearest old data point as answer



- Generalize to kNN, that is take the average class of the closest k neighbors.

[Application: Eating out]

Problem: Decide on a restaurant, based on the following attributes:

1. Alternate: is there an alternative restaurant nearby?
2. Bar: is there a comfortable bar area to wait in?
3. Fri/Sat: is today Friday or Saturday?
4. Hungry: are we hungry?
5. Patrons: number of people in the restaurant (None, Some, Full)
6. Price: price range (\$, \$\$, \$\$\$)
7. Raining: is it raining outside?
8. Reservation: have we made a reservation?
9. Type: kind of restaurant (French, Italian, Thai, Burger)
10. WaitEstimate: estimated waiting time (0-10, 10-30, 30-60, >60)

Attribute representation

- Examples described by **attribute** or **feature values** (Boolean, discrete, continuous)
- E.g., situations where I will/won't wait for a table:

Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>Wait</i>
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)

[Bayes Rule]

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Which is shorthand for:

$$P(Y = y_i | X = x_j) = \frac{P(X = x_j | Y = y_i)P(Y = y_i)}{P(X = x_j)}$$

[Naïve Bayes Classifier]

- Calculate most probable function value

$$\begin{aligned} V_{\text{map}} &= \operatorname{argmax} P(v_j | a_1, a_2, \dots, a_n) \\ &= \operatorname{argmax} \frac{P(a_1, a_2, \dots, a_n | v_j) P(v_j)}{P(a_1, a_2, \dots, a_n)} \\ &= \operatorname{argmax} P(a_1, a_2, \dots, a_n | v_j) P(v_j) \end{aligned}$$

Naïve assumption: $P(a_1, a_2, \dots, a_n) = P(a_1)P(a_2) \dots P(a_n)$

[Naïve Bayes Algorithm]

NaïveBayesLearn(*examples*)

For each target value v_j

$P'(v_j) \leftarrow \text{estimate } P(v_j)$

For each attribute value a_i of each attribute a

$P'(a_i|v_j) \leftarrow \text{estimate } P(a_i|v_j)$

ClassfyingNewInstance(x)

$$v_{nb} = \underset{v_j \in V}{\operatorname{argmax}} P'(v_j) \prod_{a_i \in x} P'(a_i|v_j)$$

[An Example]

(due to MIT's open coursework slides)

f_1	f_2	f_3	f_4	y
0	1	1	0	1
0	0	1	1	1
1	0	1	0	1
0	0	1	1	1
0	0	0	0	1
1	0	0	1	0
1	1	0	1	0
1	0	0	0	0
1	1	0	1	0
1	0	1	1	0

$R_1(1,1) = 1/5$: fraction of all positive examples that have feature 1 = 1

$R_1(0,1) = 4/5$: fraction of all positive examples that have feature 1 = 0

$R_1(1,0) = 5/5$: fraction of all negative examples that have feature 1 = 1

$R_1(0,0) = 0/5$: fraction of all negative examples that have feature 1 = 0

Continue calculation of $R_2(1,0) \dots$

[An Example]

(due to MIT's open coursework slides)

f_1	f_2	f_3	f_4	y
0	1	1	0	1
0	0	1	1	1
1	0	1	0	1
0	0	1	1	1
0	0	0	0	1
1	0	0	1	0
1	1	0	1	0
1	0	0	0	0
1	1	0	1	0
1	0	1	1	0

(1,1) (0,1) (1,0) (0,0)

$$R_1 = 1/5, 4/5, 5/5, 0/5$$

$$R_2 = 1/5, 4/5, 2/5, 3/5$$

$$R_3 = 4/5, 1/5, 1/5, 4/5$$

$$R_4 = 2/5, 3/5, 4/5, 1/5$$

New $x = \langle 0, 0, 1, 1 \rangle$

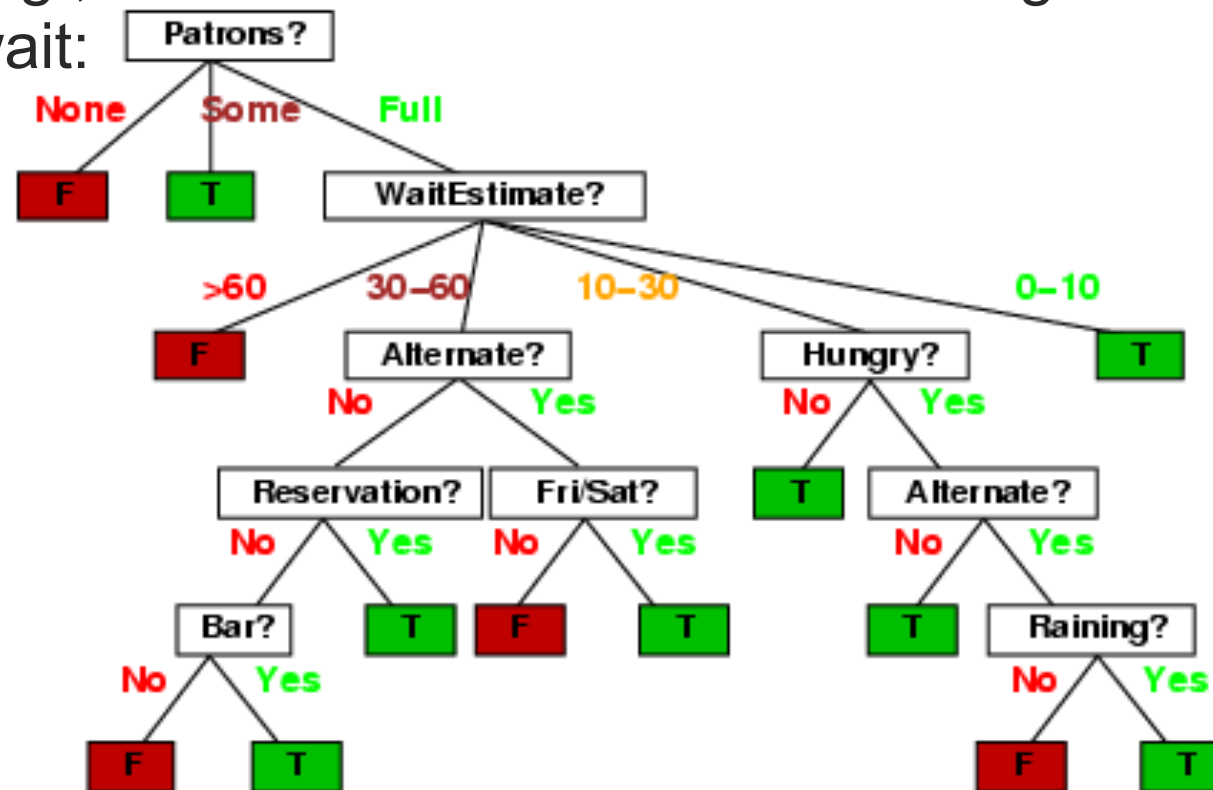
$$S(1) = R_1(0,1) * R_2(0,1) * R_3(1,1) * R_4(1,1) = .205$$

$$S(0) = R_1(0,0) * R_2(0,0) * R_3(1,0) * R_4(1,0) = 0$$

$S(1) > S(0)$, so predict $v = 1$.

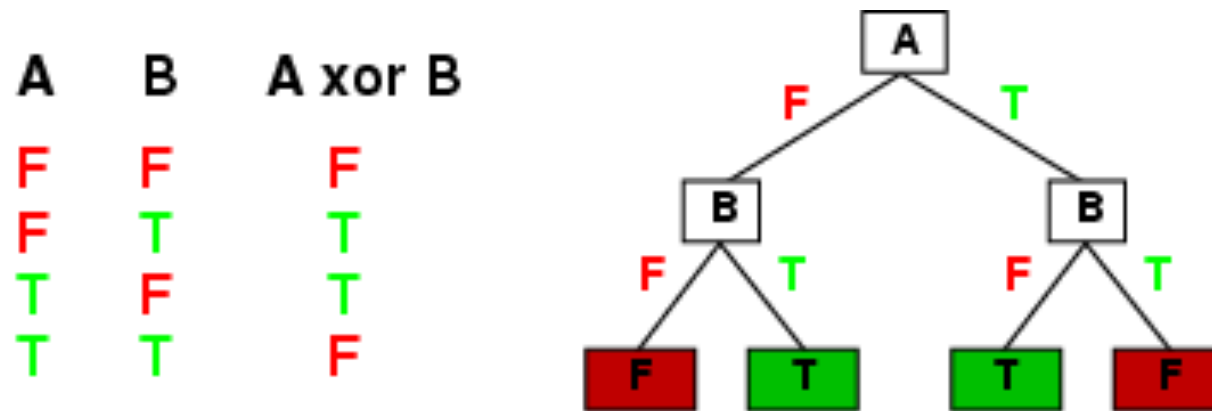
Decision trees

- Developed simultaneously by statistics and AI
- E.g., here is the “true” tree for deciding whether to wait:



Expressiveness

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



- Trivially, there is a consistent decision tree for any training set with 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

[Hypothesis spaces]

How many distinct decision trees with n Boolean attributes?

= number of Boolean functions

= number of distinct truth tables with 2^n rows = 2^{2^n}

- E.g., with 6 Boolean attributes, there are 18,446,744,073,709,551,616 trees

[Hypothesis spaces]

How many distinct decision trees with n Boolean attributes?

= number of Boolean functions

= number of distinct truth tables with 2^n rows = 2^{2^n}

- E.g., with 6 Boolean attributes, there are 18,446,744,073,709,551,616 trees

How many purely conjunctive hypotheses (e.g., $Hungry \wedge \neg Rain$)?

- Each attribute can be in (positive), in (negative), or out
 $\Rightarrow 3^n$ distinct conjunctive hypotheses
- More expressive hypothesis space
 - increases chance that target function can be expressed
 - increases number of hypotheses consistent with training set
 \Rightarrow may get worse predictions

[The best hypothesis]

- Find **best** function that models given data.
- How to define the best function?
 - Fidelity to the data – error on existing data: $E(h,D)$
 - Simplicity – how complicated is the solution: $C(h)$
 - One measure: how many possible hypotheses for the class?
- Inevitable tradeoff between complexity of hypothesis and degree of fit to the data
- Minimize $\alpha E(h,D) + (1-\alpha) C(h)$
 - Where α is a tuning parameter

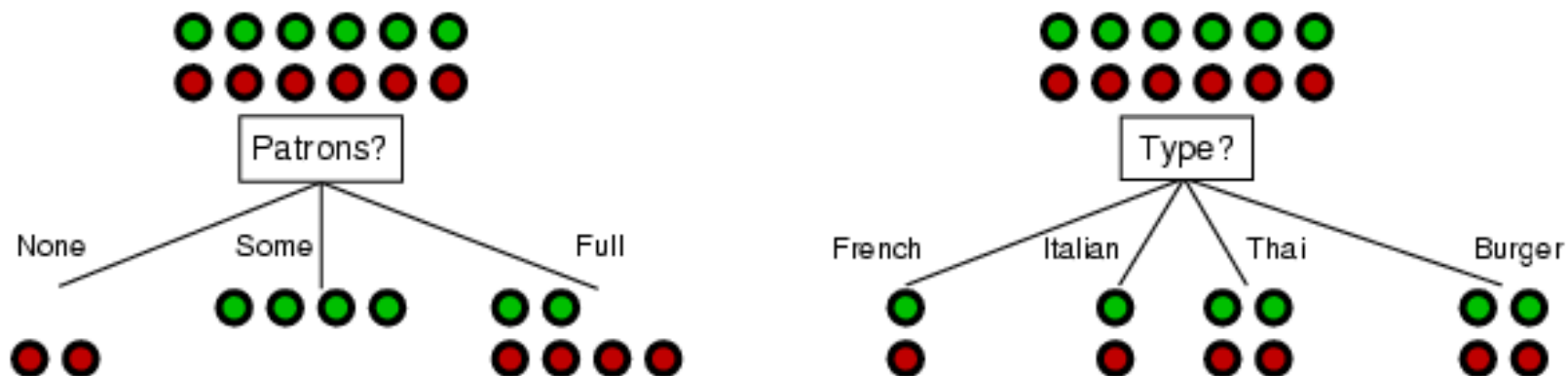
Decision tree learning

- 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
       $examples_i \leftarrow \{\text{elements of } examples \text{ with } best = v_i\}$ 
      subtree ← DTL(examplesi, 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 training set into subsets that are (ideally) "all positive" or "all negative"



- *Patrons?* is a better choice

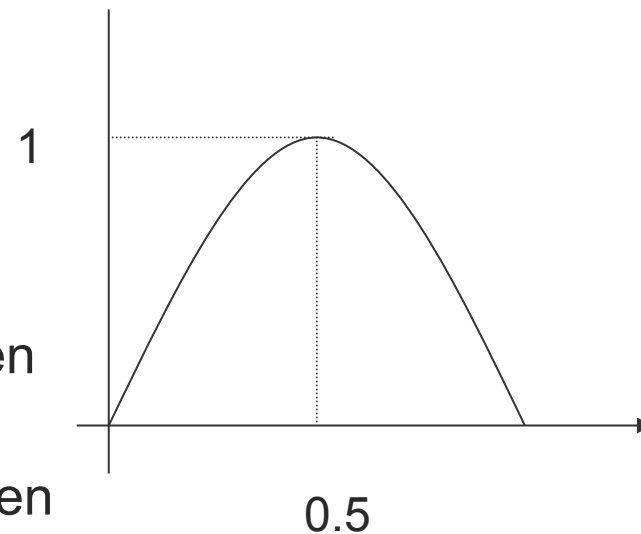
Information Content

- **Entropy** measures purity of sets of examples
 - Normally denoted $H(x)$
- Or as **information content**: the less you need to know (to determine class of new case), the more information you have
- With two classes (P,N):
 - $IC(S) = - (p/t) \log_2 (p/t) - (n/t) \log_2 (n/t)$
 - E.g., $p=9, n=5$;
 $IC([9,5]) = - (9/14) \log_2 (9/14) - (5/14) \log_2 (5/14)$
 $= 0.940$
 - Also, $IC([14,0])=0$; $IC([7,7])=1$

$$T \text{ (total)} = P + N$$

[Entropy curve]

- For p/t between 0 & 1, the 2-class entropy is
 - 0 when $p/(p+n)$ is 0
 - 1 when $p/(p+n)$ is 0.5
 - 0 when $p/(p+n)$ is 1
 - monotonically increasing between 0 and 0.5
 - monotonically decreasing between 0.5 and 1
 - When the data is pure, only need to send 1 bit



[Using information theory]

- To implement **Choose-Attribute** in the DTL algorithm
- Entropy:
$$I(P(v_1), \dots, P(v_n)) = \sum_{i=1} -P(v_i) \log_2 P(v_i)$$
- For a training set containing p positive examples and n negative examples:

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$

Information gain

- A chosen attribute A divides the training set E into subsets E_1, \dots, E_v according to their values for A , where A has v distinct values.

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

- Information Gain (IG) or reduction in entropy from the attribute test:

$$IG(A) = I\left(\frac{p}{p + n}, \frac{n}{p + n}\right) - remainder(A)$$

- Choose the attribute with the largest IG

Information gain

For the training set, $p = n = 6$, $I(6/12, 6/12) = 1$ bit

Consider the attributes *Patrons* and *Type* (and others too):

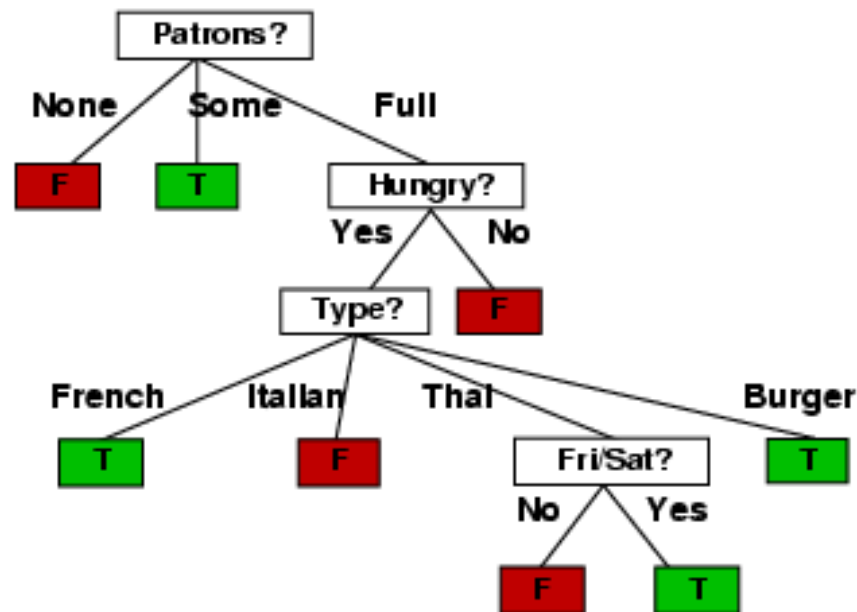
$$IG(Patrons) = 1 - \left[\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I\left(\frac{2}{6}, \frac{4}{6}\right) \right] = .0541 \text{ bits}$$

$$IG(Type) = 1 - \left[\frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right) + \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right) \right] = 0 \text{ bits}$$

Patrons has the highest IG of all attributes and so is chosen by the DTL algorithm as the root

[Example contd.]

- Decision tree learned from the 12 examples:



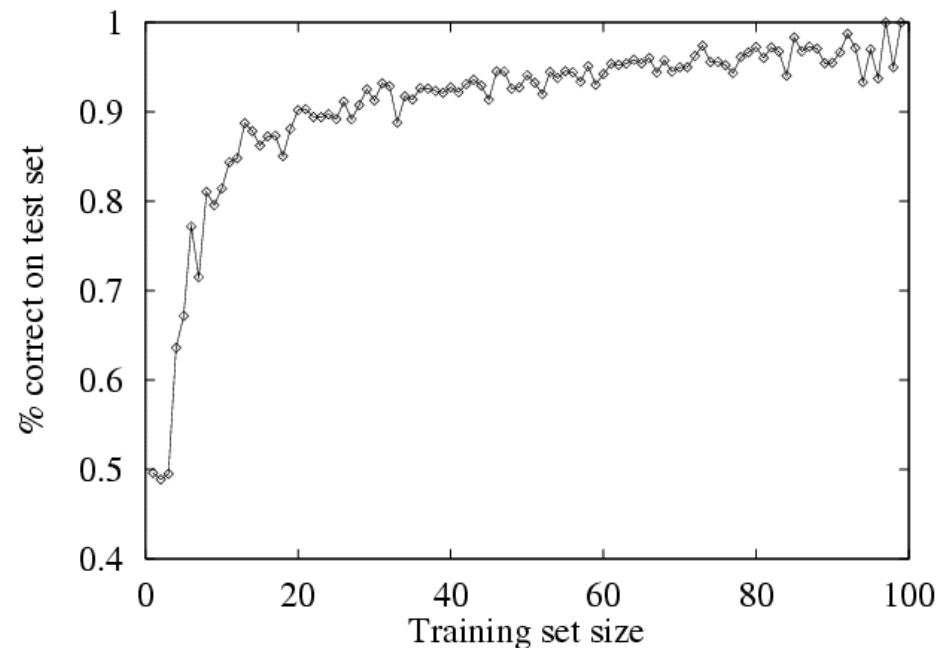
- Substantially simpler than “true” tree---a more complex hypothesis isn’t justified by small amount of data

Performance measurement

How do we know that $h \approx f$?

- Try h on a new **test set** of examples

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



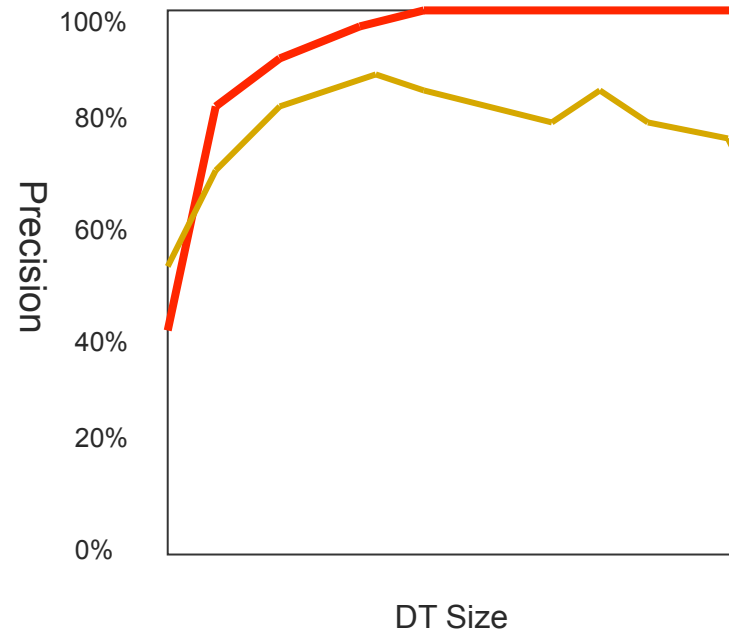
[Training and testing sets]

- Where does the test set come from?
 1. Collect a large set of examples
 2. Divide into **training** and **testing data**
 3. Train on training data, assess on testing
 4. Repeat 1-3 for different splits of the set.

- Same distribution
 - “Learning ... enable[s] the system to do the task or tasks drawn from the **same population**” – Herb Simon
 - To think about: Why?

[Overfitting]

- Better training performance = test performance?
- Nope. Why?
 1. Hypothesis too specific
 2. Models noise
- Pruning
 - Keep complexity of hypothesis low
 - Stop splitting when:
 1. IC below a threshold
 2. Too few data points in node



Test performance
Train performance

[Summary]

- Learning needed for unknown environments, lazy designers
- Learning agent = performance element + learning element
- For supervised learning, the aim is to find a simple hypothesis approximately consistent with training examples
- Decision tree learning using information gain
- Learning performance = prediction accuracy measured on test set