Lecture 17: Learning -1

Victor Lesser

CMPSCI 683
Fall 2004

Today's Lecture

- The structure of a learning agent
- Basic problems: bias, Ockham's razor, expressiveness
- Decision-tree algorithms

Commonsense Definition

Learning is change within a system that improves its performance

This admits a lot of different behaviors, but identifies the basic preconditions of learning:

- Learning systems must be capable of change
- Learning systems must do something differently as a result of the change

Why Should Systems Learn?

- A viable alternative to problem solving.
- Learning can simplify the complexity of problem solving.
  - **Replace procedural knowledge, inferencing, search with learned functions and policies**
- Learning increases efficiency, robustness, survivability, and autonomy of system.
  - **Key to operating in "open" environments**
- A learning program can become better than its teacher.

Characterizing Learning Systems

– What changes as a result of learning?

– How does the system find out change is needed?

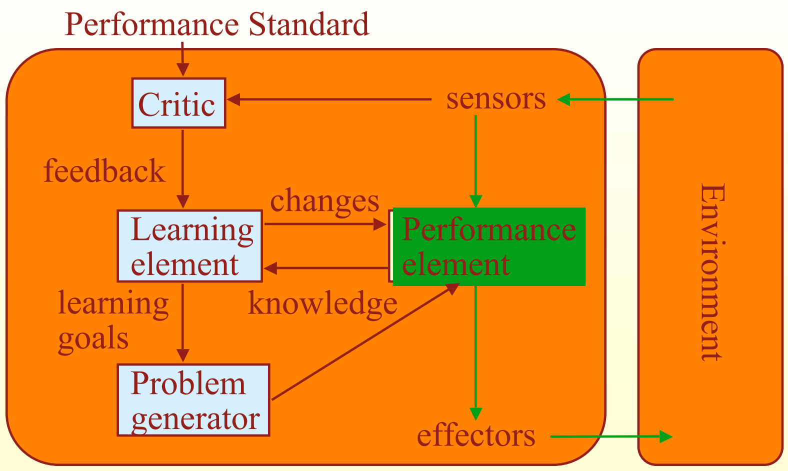
– How does the system localize the problem to find out what changes are necessary?

– What is the mechanism of change?

Available Feedback

- **Supervised learning**
  - Is told by a “teacher” what action is best in a specific situation
- **Reinforcement Learning**
  - Gets feedback about the consequences of a specific sequence of actions in a certain situation
  - Can also be thought of as supervised learning with a less informative feedback signal**.**
- **Unsupervised Learning**
  - No feedback about actions
  - Learns to predict future precepts given its previous precepts
  - Can’t learn what to do unless it already has a utility function

A Model of Learning Agents



Model of Learning Agent

- **Learning element — modifies performance element in response to feedback**
- **Critic — tells learning element how well agent is doing**
  - Fixed standard of performance
- **Problem generator — suggests actions that will lead to new and informative experiences**
  - Related to decision to acquire information

Design of Learning Element

Goals:

- *Learn better actions*
- *Speed up performance element*

- **Which *components* of the performance element are to be improved.**
- **What *representation* is used for those components.**
- **What *feedback* is available**
- **What *prior information* is available.**

Types of Learned Knowledge

- **A direct mapping from conditions on the current state to actions.**
- **Weighting of parameters of multi-attribute decision process**
- **A means to infer relevant properties of the world from the percept sequence.**
- **Information about the way the world evolves.**
  - Allow prediction of future events

Applicability of Learned Knowledge cont.

- **Information about the results of possible actions the agent can take**
- **Utility information indicating the desirability of world states.**
- **Action-value information indicating the desirability of particular actions in particular states.**
- **Goals that describe classes of states whose achievement maximizes the agent’s utility.**

Dimensions of Learning

- ***The type of training instances***
  - *the beginning data for the learning task.*
- ***The language used to represent knowledge.***
  - Specific training instances must be translated into this representation language
  - In some programs the training instances are in the same language as the internal knowledge base and this step is unnecessary.
- ***A set of operations on representations.***
  - Typical operations generalize or specialize existing knowledge, combine units of knowledge, or otherwise modify the program’s existing knowledge or the representation of the training instances.

Dimensions of Learning cont.

- ***The concept space.***
  - The operations that define a space of possible knowledge structures that is searched to find the appropriate characterization of the training instances and similar problems.
- ***The learning algorithms and heuristics employed to search the concept space.***
  - The order of the search and the use of heuristics to guide the search.

Types of Knowledge Representations for Learning

- numerical parameters
- decision trees
- formal grammars
- production rules
- logical theories
- graphs and networks
- frames and schemas
- computer programs (procedural encoding)

Learning Functions

All learning can be seen as learning the representation of a function

- **Choice of representation of a function**
  - Trade-off between expressiveness and efficiency
    - Is what you want representable?
    - Is what you want learnable (# of examples, cost of search)?
- **Choice of training data**
  - Correctly reflects past experiences
  - Correctly predicts future experiences
- **How to judge the goodness of the learned function**

Some Additional Thoughts

- **Importance of Prior Knowledge**
  - **Prior knowledge can significantly speed up learning process**
  - **EBL: explanation-based learning**
- **Learning as a search process**
  - Finding the “best” function
- **Incremental Process (on-line) vs. off-line**

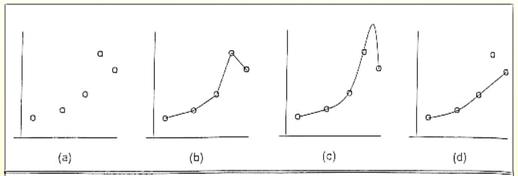
Inductive (Supervised) Learning

Let an example be *(x, f(x))*

- **Give a collection of examples of *f*, return a function *h* that approximates *f*.**
- **This function *h* is called a hypothesis:**
  - Feedback is relation between *f(x)* and *h(x)*
  - *(x, f(x))* could only be approximately correct
    - Noise, missing components

Problems

- Many hypotheses *h's* are **approximately** consistent with the training set
- Curve-fitting ...



In (a) we have some example (input, output) pairs. In (b), (c), and (d) we have three hypotheses for functions from which these examples could be drawn.

- A preference for one hypothesis over another beyond consistency is called **Bias**.

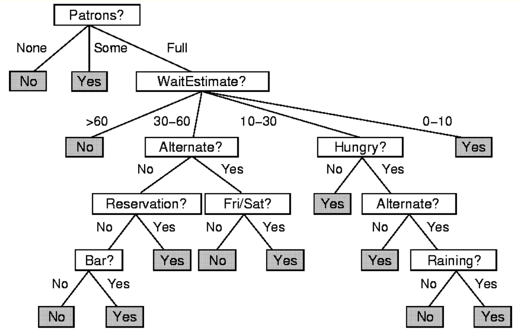
Ockham's Razor

- "Simple" hypotheses that are consistent with data are preferred
- We want to maximize some metric of *consistency* and *simplicity* in the choice of the most appropriate function

Learning Classification Decision Trees

- **Restricted representation of logical sentences**
  - Boolean functions
- **Takes as input situation described by a set of properties and outputs a "yes/no" decision**
- **Tree of property value tests**
  - Terminals are decisions

Learn, based on conditions of the situation, whether to wait at a restaurant for a table



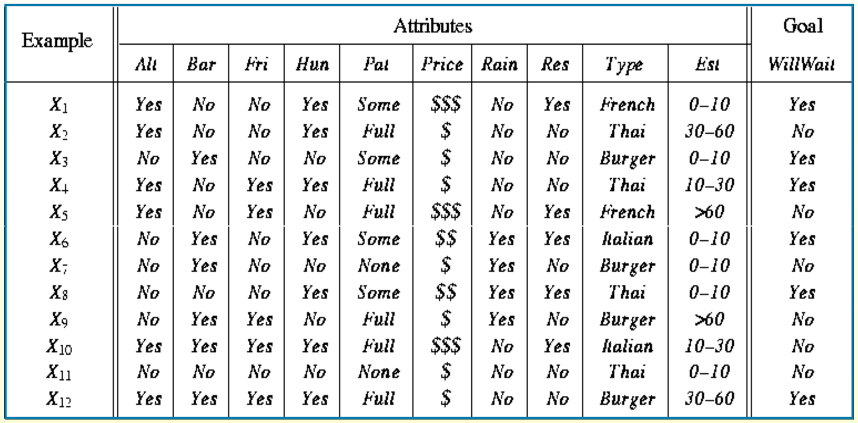
Decision trees

- A (classification) decision tree takes as input a situation described by a set of attributes and returns a “decision.”
- Can express any boolean function of the input attributes.
- How to choose between equally consistent trees

Example: Waiting for a table

- Alternate
- Bar
- Fri/Sat
- Hungry
- Patrons (None, Some, Full)
- Price ($, $$, $$$)
- Raining
- Reservation
- Type (French, Italian, Thai, Burger)
- WaitEstimate (0-10, 10-30, 30-60, >60)

Inducing Decision Trees from Examples



Example	Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	*Yes*
X_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	*No*
X_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	*Yes*
X_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*10–30*	*Yes*
X_5	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	*No*
X_6	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	*Yes*
X_7	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	*No*
X_8	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	*Yes*
X_9	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	*No*
X_{10}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	*No*
X_{11}	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	*No*
X_{12}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	*Yes*

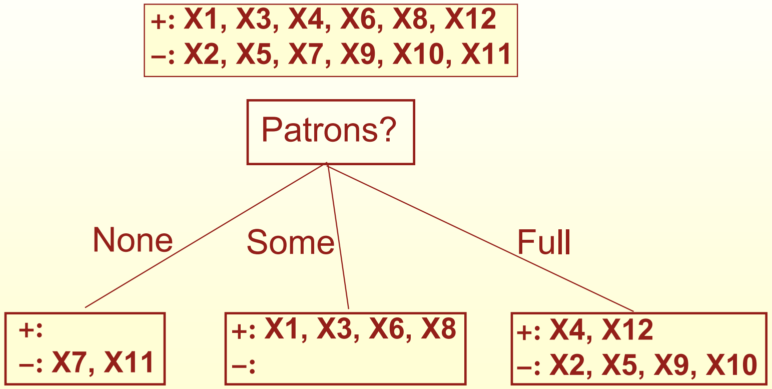
Constructing the Decision Tree

Construct a root node that includes all the examples, then for each node:

1. if there are both positive and negative examples, choose the best attribute to split them.
2. if all the examples are pos (neg) answer yes (no).
3. if there are no examples for a case (no observed examples) then choose a default based on the majority classification at the parent.
   - **Case of raining under hungry- yes,alternate - yes**
4. if there are no attributes left but we have both pos and neg examples, this means that the selected features are not sufficient for classification or that there is error in the examples. (can use majority vote.)

Splitting the Examples

- A perfect attribute divides the examples into sets that are all positive and negative



+: X1, X3, X4, X6, X8, X12
–: X2, X5, X7, X9, X10, X11

Patrons?

None

Some

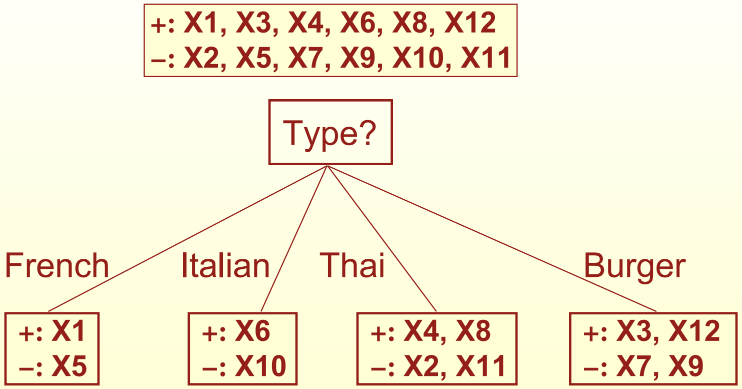
Full

+:
–: X7, X11

+: X1, X3, X6, X8
–:

+: X4, X12
–: X2, X5, X9, X10

Splitting Examples cont.



+: X1, X3, X4, X6, X8, X12
–: X2, X5, X7, X9, X10, X11

Type?

French

Italian

Thai

Burger

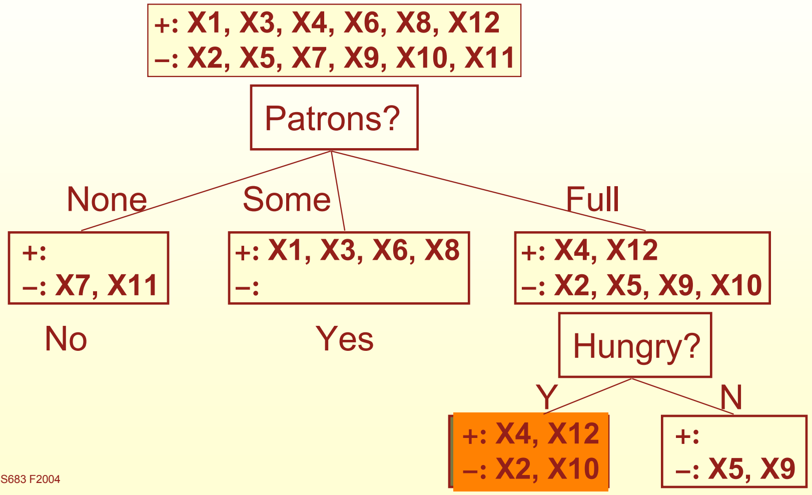
+: X1
–: X5

+: X6
–: X10

+: X4, X8
–: X2, X11

+: X3, X12
–: X7, X9

Splitting Examples cont.



+: X1, X3, X4, X6, X8, X12
–: X2, X5, X7, X9, X10, X11

Patrons?

None

Some

Full

+:
–: X7, X11

+: X1, X3, X6, X8
–:

+: X4, X12
–: X2, X5, X9, X10

No

Yes

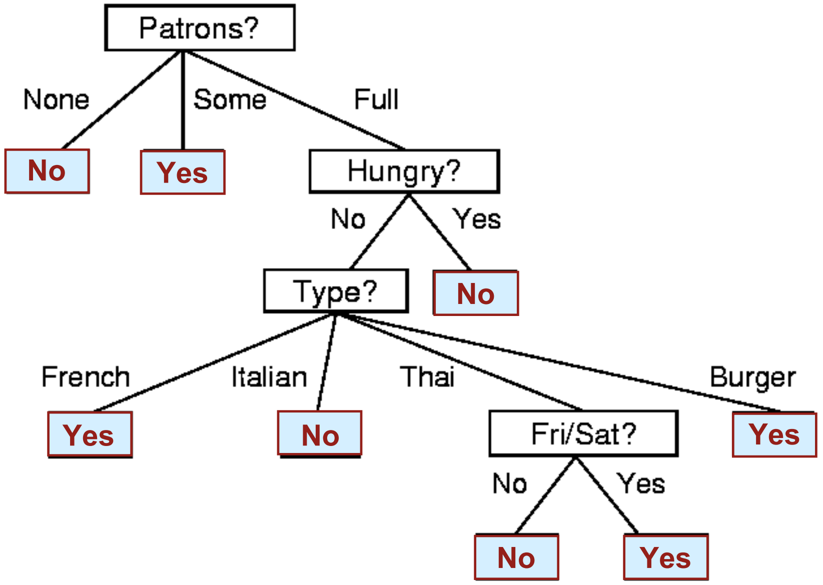
Hungry?

Y

N

+: X4, X12
–: X2, X10

+:
–: X5, X9



Decision Tree Algorithm

- Basic idea is to build the tree greedily.
  - **Decisions once made are not revised**
  - **No search**
- Choose “most significant attribute” to be the root. Then split the dataset in two halves, and recurse.
- Define “significance” using information theory (based on information gain or “entropy”).

Finding the *smallest* decision tree is an intractable problem

Expressions of Decision Tree

- **Any Boolean function can be written as a decision tree**
  - *∀r Patrons(r,Full)* Λ *WaitEstimate(r,10-30)* Λ *Hungry(r,N)* $\Rightarrow$ *WillWait(r)*
  - Row of truth table path in decision tree
  - 2^n rows given n literals, 2^{2^n} functions

Limits on Expressability

- **Cannot use decision tree to represent tests that refer to two or more different objects**
- ***∃r_2 Nearby(r_2,r)* Λ *Price(r,p)* Λ *Price(r_2,p_2)* Λ *Cheaper(p_2,p)***
- **New Boolean attribute: *CheaperRestaurantNearby* but intractable to add all such attributes**
- **Some truth tables cannot be compactly represented in decision tree**
  - Parity function
    - **returns 1 if and only if an even number of inputs are 1**
    - **exponentially large decision tree will be needed.**
  - Majority function
    - **which returns 1 if more than half of its inputs are 1.**

Choosing the Best Attribute Based on Information Theory

- **Expected amount of information provided by an attribute**
  - Similar to the concept of value of perfect information
- **Amount of information content in a set of examples**
  - V_i is the possible answers, *p* positive, *n negative*

$$I(P(v_1),\ldots,P(v_n)) = \sum_{i=1}^{n} -P(v_i)\log_2 P(v_i)$$

  - *Example 12 cases, 6 pos, 6 neg; information 1 bit*

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

  - *v number of attributes of Attribute A*

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

Choosing the Best Attribute Based on Information Theory cont.

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

$$Gain(Patrons) = 1 - \left[\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I\left(\frac{2}{6}, \frac{4}{6}\right) \right] \approx 0.541 bits$$

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

Example *(Quinlan '83)*

CLASS	HEIGHT	HAIR	EYES
−	SHORT	BLOND	BROWN
−	TALL	DARK	BROWN
+	TALL	BLOND	BLUE
−	TALL	DARK	BLUE
−	SHORT	DARK	BLUE
+	TALL	RED	BLUE
−	TALL	BLOND	BROWN
+	SHORT	BLOND	BLUE

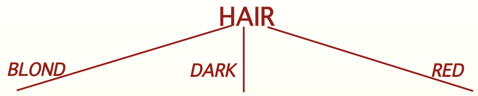
	HEIGHT	
	SHORT	*TALL*
+	1	2
−	2	3

	HAIR		
	BLOND	*DARK*	*RED*
+	2	0	1
−	2	3	0

	EYES	
	BROWN	*BLUE*
+	0	3
−	3	2

Partition on hair gives least Impurity

Example *(Quinlan '83) cont.*



CLASS	HEIGHT	EYES
-	SHORT	BROWN
+	TALL	BLUE
-	TALL	BROWN
+	SHORT	BLUE

CLASS	HEIGHT	EYES
-	TALL	BROWN
-	TALL	BLUE
-	SHORT	BLUE

CLASS	HEIGHT	EYES
+	TALL	BLUE

HEIGHT	
SHORT	*TALL*
1	1
1	1

$\frac{2}{4}\mathbf{I}(1,1) + \frac{2}{4}\mathbf{I}(1,1)$

1- 0.6931

EYES	
BROWN	*BLUE*
0	2
2	0

$\frac{2}{4}\mathbf{I}(0,2) + \frac{2}{4}\mathbf{I}(2,0)$

1- 0.0

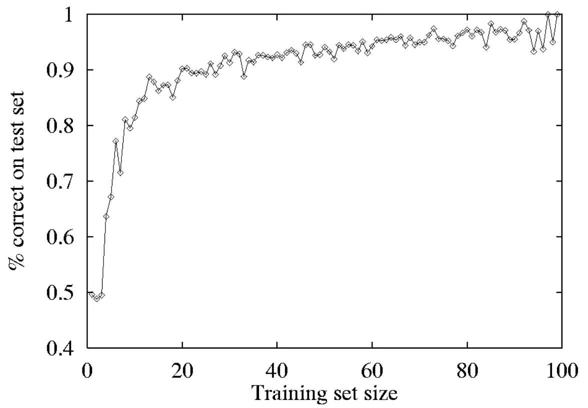
EYES ARE BETTER ATTRIBUTE

Performance Measurement

- How do we measure how close our hypothesis is to f()?
- Try h() on a test set
- Learning curve: Measure % correct predictions on the test set as a function of the size of the training set.

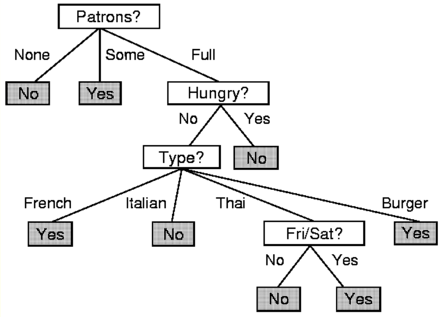
Assessing the Performance of the Learning Algorithm

- **Randomly divide available examples into test and training set**



A learning curve for the decision tree algorithm on 100 randomly generated examples in the restaurant domain. The graph summarizes 20 trials.

Full Learned Decision Tree



- **How “correct” is this?**
  - Can we even judge this idea?
  - Not all attributes used
- **How does the number of examples seen relate to the likelihood of “correctness”?**

Noise and Overfitting

- Finding meaningless “regularities” in the data.
- With enough attributes, you’re likely to find one which captures some of the noise in your data.
- One solution is to prune the tree. Collapse subtrees which provide only minor improvements
  - Using information gain as a criteria

Broadening the applicability - Missing Data

- Handling examples with missing data
  - **Add new attribute value -”unknown”**
  - **Instantiated example with all possible values of missing attribute but assign weights to each instance based on likelihood of missing value being a particular value given the distribution of examples in the parent node**
    - **Modify decision tree algorithm to take into account weighting**

Broadening the applicability - Multivalued Attributes

- Handling multivalued (large) attributes and classification
  - **Need another measure of information gain**
  - **Information gain measure gives inappropriate indication of attributed usefulness because of likelihood of singleton values**
  - **Gain ratio**
    - **Gain over intrinsic information content**

Broadening the Applicability - Continuous-Valued attributes

- Continuous-valued attributes
  - **Discretize**
    - **Example $,$$, $$$**
  - **Preprocess to find out which ranges give the most useful information for classification purposes**
- Incremental construction

Next Lecture

- **The version space algorithm**
- **Neural Networks**