



Lecture 33 of 42

Introduction to Machine Learning Discussion: BNJ

Monday, 12 November 2007

William H. Hsu
Department of Computing and Information Sciences, KSU

KSOL course page: <http://snipurl.com/v9v3>
Course web site: <http://www.kddresearch.org/Courses/Fall-2007/CIS730>
Instructor home page: <http://www.cis.ksu.edu/~bhsu>

Reading for Next Class:
Section 18.3, Russell & Norvig 2nd edition



Lecture Outline

- Today's Reading: Sections 18.1 – 18.2, R&N 2e
- Next Monday's Reading: Section 18.3, R&N 2e
- Machine Learning
 - * Definition
 - * Supervised learning and hypothesis space
- Brief Tour of Machine Learning
 - * A case study
 - * A taxonomy of learning
 - * Specification of learning problems
- Issues in Machine Learning
 - * Design choices
 - * The performance element: intelligent systems
- Some Applications of Learning
 - * Database mining, reasoning (inference/decision support), acting
 - * Industrial usage of intelligent systems





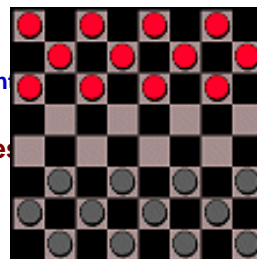
Rule and Decision Tree Learning

- **Example: Rule Acquisition from Historical Data**
- **Data**
 - * Patient 103 (time = 1): Age 23, First-Pregnancy: no, Anemia: no, Diabetes: no, Previous-Premature-Birth: no, Ultrasound: *unknown*, Elective C-Section: *unknown*, Emergency-C-Section: *unknown*
 - * Patient 103 (time = 2): Age 23, First-Pregnancy: no, Anemia: no, Diabetes: yes, Previous-Premature-Birth: no, Ultrasound: abnormal, Elective C-Section: no, Emergency-C-Section: *unknown*
 - * Patient 103 (time = n): Age 23, First-Pregnancy: no, Anemia: no, Diabetes: no, Previous-Premature-Birth: no, Ultrasound: *unknown*, Elective C-Section: no, Emergency-C-Section: YES
- **Learned Rule**
 - * IF _____ no previous vaginal delivery, AND abnormal 2nd trimester ultrasound, AND malpresentation at admission, AND no elective C-Section THEN probability of emergency C-Section is 0.6
 - * Training set: 26/41 = 0.634
 - * Test set: 12/20 = 0.600



Specifying A Learning Problem

- **Learning = Improving with Experience at Some Task**
 - * Improve over task T ,
 - * with respect to performance measure P ,
 - * based on experience E .
- **Example: Learning to Play Checkers**
 - * T : play games of checkers
 - * P : percent of games won in world tournament
 - * E : opportunity to play against self
- **Refining the Problem Specification: Issues**
 - * What experience?
 - * What *exactly* should be learned?
 - * How shall it be *represented*?
 - * What specific algorithm to learn it?
- **Defining the Problem Milieu**
 - * Performance element: How shall the results of learning be applied?
 - * How shall the performance element be evaluated? The learning system?





Example: Learning to Play Checkers

- **Type of Training Experience**
 - * Direct or indirect?
 - * Teacher or not?
 - * Knowledge about the game (e.g., openings/endgames)?
- **Problem: Is Training Experience *Representative* (of Performance Goal)?**
- **Software Design**
 - * Assumptions of the learning system: *legal* move generator exists
 - * Software requirements: generator, evaluator(s), parametric target function
- **Choosing a Target Function**
 - * *ChooseMove*: $Board \rightarrow Move$ (action selection function, or *policy*)
 - * $V: Board \rightarrow R$ (board evaluation function)
 - * Ideal target V ; approximated target
 - * Goal of learning process: operational description (approximation) of V



A Target Function for Learning to Play Checkers

- **Possible Definition**
 - * If b is a final board state that is won, then $V(b) = 100$
 - * If b is a final board state that is lost, then $V(b) = -100$
 - * If b is a final board state that is drawn, then $V(b) = 0$
 - * If b is not a final board state in the game, then $V(b) = V(b')$ where b' is the best final board state that can be achieved starting from b and playing optimally until the end of the game
 - * *Correct values, but not operational*
- **Choosing a Representation for the Target Function**
 - * Collection of rules?
 - * Neural network?
 - * Polynomial function (e.g., linear, quadratic combination) of board features?
 - * Other?
- **A Representation for Learned Function**
 - * $\hat{V}(b) = w_0 + w_1 bp(b) + w_2 rp(b) + w_3 bk(b) + w_4 rk(b) + w_5 bt(b) + w_6 rt(b)$
 - * bp/rp = number of black/red pieces; bk/rk = number of black/red kings;
= number of black/red pieces *threatened* (can be taken on next turn)



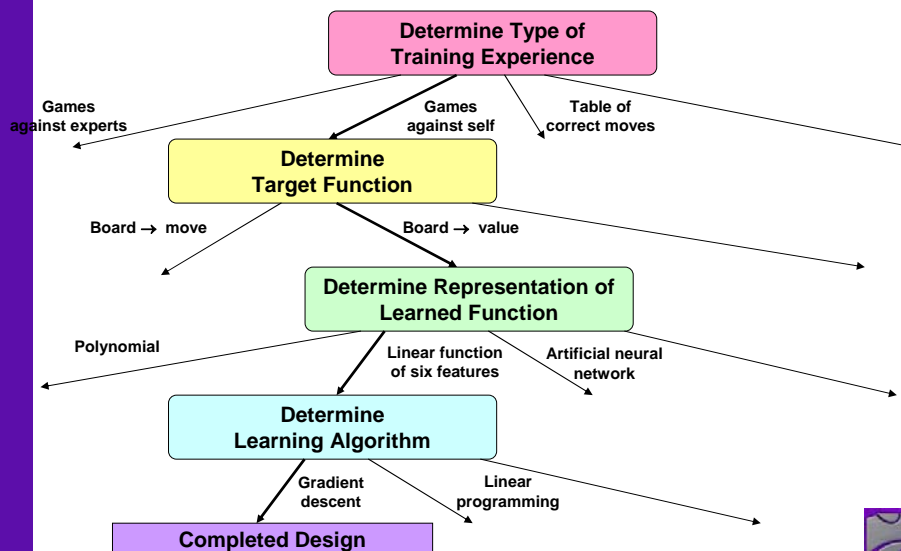
A Training Procedure for Learning to Play Checkers

- **Obtaining Training Examples**
 - * $V(b)$ the target function
 - * $\hat{V}(b)$ the learned function
 - * $V_{train}(b)$ the training value
- **One Rule For Estimating Training Values:**
 - * $V_{train}(b) \leftarrow \hat{V}(Successor(b))$
- **Choose Weight Tuning Rule**
 - * **Least Mean Square (LMS) weight update rule:**
REPEAT
 - Select a training example b at random
 - Compute the $error(b)$ for this training example
 $error(b) = V_{train}(b) - \hat{V}(b)$
 - For each board feature f_i , update weight w_i as follows:
 $w_i \leftarrow w_i + c \cdot f_i \cdot error(b)$

where c is a small, constant factor to adjust the learning rate

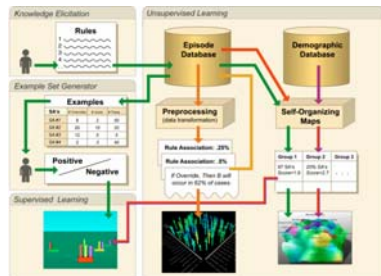


Design Choices for Learning to Play Checkers





Interesting Applications



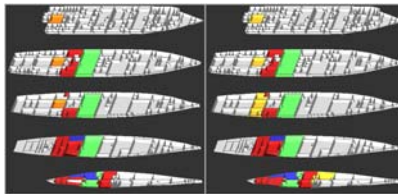
NCSA D2K - <http://alg.ncsa.uiuc.edu>

Database Mining



Cartia ThemeScapes - <http://www.cartia.com>

Reasoning (Inference, Decision Support)



DC-ARM - <http://www.kbs.ai.uiuc.edu>



Planning, Control



Example: Learning A Concept (*EnjoySport*) from Data

- **Specification for Training Examples**
 - * Similar to a data type definition
 - * 6 variables (*aka attributes, features*):
Sky, Temp, Humidity, Wind, Water, Forecast
 - * Nominal-valued (symbolic) attributes - enumerative data type
- **Binary (Boolean-Valued or H-Valued) Concept**
- **Supervised Learning Problem: *Describe the General Concept***

Example	Sky	Air Temp	Humidity	Wind	Water	Forecast	Enjoy Sport
0	Sunny	Warm	Normal	Strong	Warm	Same	Yes
1	Sunny	Warm	High	Strong	Warm	Same	Yes
2	Rainy	Cold	High	Strong	Warm	Change	No
3	Sunny	Warm	High	Strong	Cool	Change	Yes



Representing Hypotheses

- **Many Possible Representations**
- **Hypothesis h : Conjunction of Constraints on Attributes**
- **Constraint Values**
 - * Specific value (e.g., *Water = Warm*)
 - * Don't care (e.g., "*Water = ?*")
 - * No value allowed (e.g., "*Water = ∅*")
- **Example Hypothesis for *EnjoySport***

* Sky	AirTemp	Humidity	Wind	Water
Forecast			<Sunny ?	?
Strong ?	Same>			

 - * Is this consistent with the training examples?
 - * What are some hypotheses that are consistent with the examples?



Typical Concept Learning Tasks

- **Given**
 - * Instances X : possible days, each described by attributes *Sky, AirTemp, Humidity, Wind, Water, Forecast*
 - * Target function $c \equiv \text{EnjoySport}: X \rightarrow H \equiv \{\{\text{Rainy, Sunny}\} \times \{\text{Warm, Cold}\} \times \{\text{Normal, High}\} \times \{\text{None, Mild, Strong}\} \times \{\text{Cool, Warm}\} \times \{\text{Same, Change}\}\} \rightarrow \{0, 1\}$
 - * Hypotheses H : conjunctions of literals (e.g., <?, *Cold, High, ?, ?, ?>*)
 - * Training examples D : positive and negative examples of the target function

$$\langle x_1, c(x_1) \rangle, \dots, \langle x_m, c(x_m) \rangle$$

- **Determine**
 - * Hypothesis $h \in H$ such that $h(x) = c(x)$ for all $x \in D$
 - * Such h are consistent with the training data
- **Training Examples**
 - * Assumption: no missing X values
 - * Noise in values of c (contradictory labels)?



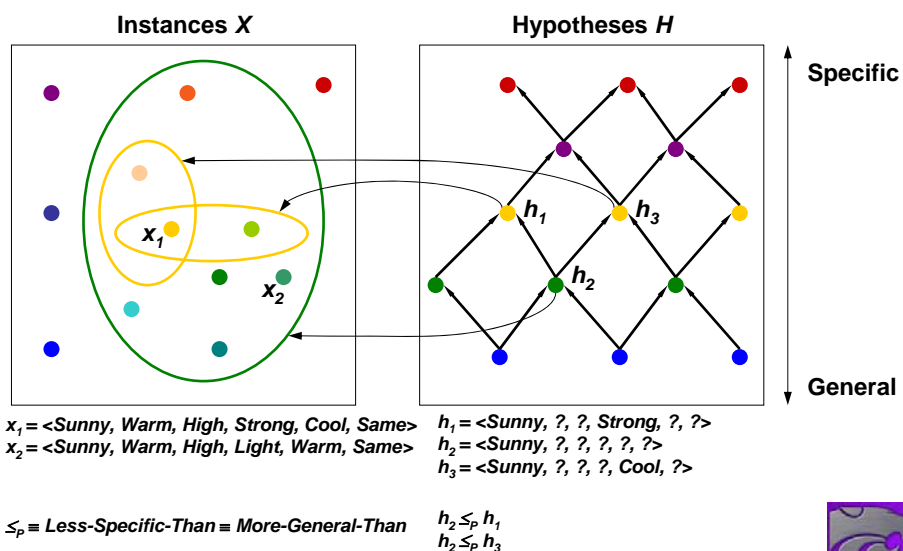


Inductive Learning Hypothesis

- Fundamental Assumption of Inductive Learning
- Informal Statement
 - * Any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples
 - * Definitions deferred: *sufficiently large, approximate well, unobserved*
- Formal Statements, Justification, Analysis
 - * Statistical (Mitchell, Chapter 5; statistics textbook)
 - * Probabilistic (R&N, Chapters 14-15 and 19; Mitchell, Chapter 6)
 - * Computational (R&N, Section 18.6; Mitchell, Chapter 7)
- More on This Topic: *Machine Learning and Pattern Recognition* (CIS732)
- Next: How to *Find* This Hypothesis?



Instances, Hypotheses, and the Partial Ordering *Less-Specific-Than*



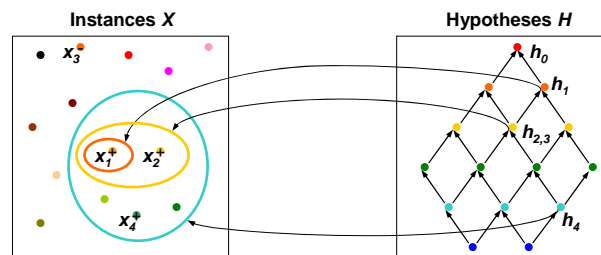


Find-S Algorithm

1. Initialize h to the most specific hypothesis in H
 H : the hypothesis space (partially ordered set under relation *Less-Specific-Than*)
2. For each positive training instance x
 For each attribute constraint a_i in h
 IF the constraint a_i in h is satisfied by x
 THEN do nothing
 ELSE replace a_i in h by the next more general constraint that is satisfied by x
3. Output hypothesis h



Hypothesis Space Search by Find-S



$x_1 = \langle \text{Sunny, Warm, Normal, Strong, Warm, Same} \rangle, +$
 $x_2 = \langle \text{Sunny, Warm, High, Strong, Warm, Same} \rangle, +$
 $x_3 = \langle \text{Rainy, Cold, High, Strong, Warm, Change} \rangle, -$
 $x_4 = \langle \text{Sunny, Warm, High, Strong, Cool, Change} \rangle, +$

$h_1 = \langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$
 $h_2 = \langle \text{Sunny, Warm, Normal, Strong, Warm, Same} \rangle$
 $h_3 = \langle \text{Sunny, Warm, ?, Strong, Warm, Same} \rangle$
 $h_4 = \langle \text{Sunny, Warm, ?, Strong, Warm, Same} \rangle$
 $h_5 = \langle \text{Sunny, Warm, ?, Strong, ?, ?} \rangle$

- Shortcomings of *Find-S*
 - * Can't tell whether it has learned concept
 - * Can't tell when training data inconsistent
 - * Picks a maximally specific h (why?)
 - * Depending on H , there might be several!





Version Spaces

- **Definition: Consistent Hypotheses**
 - * A hypothesis h is consistent with a set of training examples D of target concept c if and only if $h(x) = c(x)$ for each training example $\langle x, c(x) \rangle$ in D .
 - * $Consistent(h, D) \equiv \forall \langle x, c(x) \rangle \in D. h(x) = c(x)$
- **Definition: Version Space**
 - * The version space $VS_{H,D}$, with respect to hypothesis space H and training examples D , is the subset of hypotheses from H consistent with all training examples in D .
 - * $VS_{H,D} \equiv \{ h \in H \mid Consistent(h, D) \}$



Candidate Elimination Algorithm [1]

1. **Initialization**
 - $G \leftarrow$ (singleton) set containing most general hypothesis in H , denoted $\langle \langle ?, \dots, ? \rangle \rangle$
 - $S \leftarrow$ set of most specific hypotheses in H , denoted $\langle \langle \emptyset, \dots, \emptyset \rangle \rangle$
2. **For each training example d**
 - If d is a positive example (*Update-S*)
 - Remove from G any hypotheses inconsistent with d
 - For each hypothesis s in S that is not consistent with d
 - Remove s from S
 - Add to S all minimal generalizations h of s such that
 1. h is consistent with d
 2. Some member of G is more general than h
 (These are the greatest lower bounds, or *meets*, $s \vee d$, in $VS_{H,D}$)
 - Remove from S any hypothesis that is more general than another hypothesis in S (remove any dominated elements)





Candidate Elimination Algorithm [2]

(continued)

If d is a negative example (*Update-G*)

Remove from S any hypotheses inconsistent with d

For each hypothesis g in G that is not consistent with d

Remove g from G

Add to G all minimal specializations h of g such that

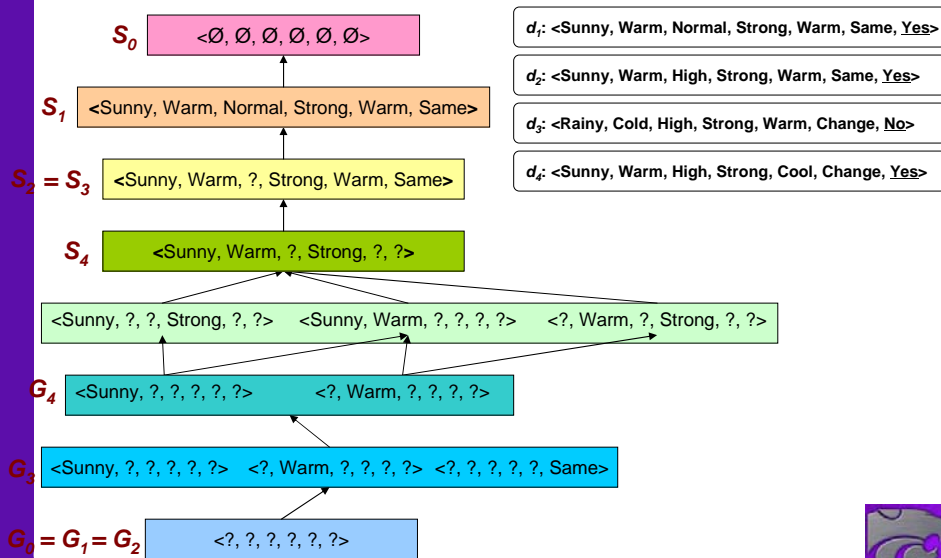
1. h is consistent with d
2. Some member of S is more specific than h

(These are the least upper bounds, or *joins*, $g \wedge d$, in $VS_{H,D}$)

Remove from G any hypothesis that is less general than another hypothesis in G (remove any dominating elements)



Example Trace





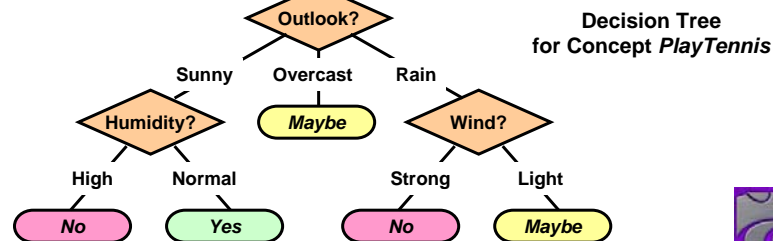
An Unbiased Learner

- Example of A Biased H
 - * Conjunctive concepts with don't cares
 - * What concepts can H not express? (Hint: what are its syntactic limitations?)
- Idea
 - * Choose H' that expresses every teachable concept
 - * i.e., H' is the power set of X
 - * Recall: $|A \rightarrow B| = |B|^{|A|}$ ($A = X$; $B = \{\text{labels}\}$; $H' = A \rightarrow B$)
 - * $\{\{\text{Rainy, Sunny}\} \times \{\text{Warm, Cold}\} \times \{\text{Normal, High}\} \times \{\text{None, Mild, Strong}\} \times \{\text{Cool, Warm}\} \times \{\text{Same, Change}\}\} \rightarrow \{0, 1\}$
- An Exhaustive Hypothesis Language
 - * Consider: H' = disjunctions (\vee), conjunctions (\wedge), negations (\neg) over previous H
 - * $|H'| = 2^{(2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 2 \cdot 2)} = 2^{96}$; $|H| = 1 + (3 \cdot 3 \cdot 3 \cdot 4 \cdot 3 \cdot 3) = 973$



Decision Trees

- Classifiers: Instances (Unlabeled Examples)
- Internal Nodes: Tests for Attribute Values
 - * Typical: equality test (e.g., "Wind = ?")
 - * Inequality, other tests possible
- Branches: Attribute Values
 - * One-to-one correspondence (e.g., "Wind = Strong", "Wind = Light")
- Leaves: Assigned Classifications (Class Labels)
- Representational Power: Propositional Logic (*Why?*)





Example: Decision Tree to Predict C-Section Risk

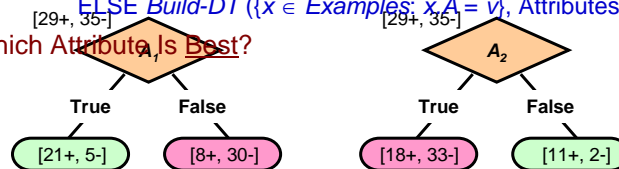
- Learned from Medical Records of 1000 Women
- Negative Examples are Cesarean Sections
 - * Prior distribution: [833+, 167-] 0.83+, 0.17-
 - * *Fetal-Presentation* = 1: [822+, 116-] 0.88+, 0.12-
 - ⇒ *Previous-C-Section* = 0: [767+, 81-] 0.90+, 0.10-
 - *Primiparous* = 0: [399+, 13-] 0.97+, 0.03-
 - *Primiparous* = 1: [368+, 68-] 0.84+, 0.16-
 - *Fetal-Distress* = 0: [334+, 47-] 0.88+, 0.12-
 - *Birth-Weight* ≥ 3349 0.95+, 0.05-
 - *Birth-Weight* < 3347 0.78+, 0.22-
 - *Fetal-Distress* = 1: [34+, 21-] 0.62+, 0.38-
 - ⇒ *Previous-C-Section* = 1: [55+, 35-] 0.61+, 0.39-
 - * *Fetal-Presentation* = 2: [3+, 29-] 0.11+, 0.89-
 - * *Fetal-Presentation* = 3: [8+, 22-] 0.27+, 0.73-



Decision Tree Learning: Top-Down Induction (*ID3*)

- Algorithm *Build-DT* (*Examples, Attributes*)
 - IF all examples have the same label THEN RETURN (leaf node with *label*)
 - ELSE
 - IF set of attributes is empty THEN RETURN (leaf with *majority label*)
 - ELSE
 - Choose best attribute *A* as root
 - FOR each value *v* of *A*
 - Create a branch out of the root for the condition *A = v*
 - IF $\{x \in \text{Examples}: x.A = v\} = \emptyset$ THEN RETURN (leaf with *majority label*)
 - ELSE *Build-DT* ($\{x \in \text{Examples}: x.A = v\}, \text{Attributes} \sim \{A\}$)

- But Which Attribute Is Best?





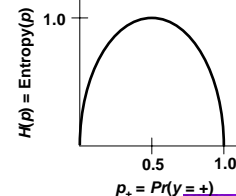
Choosing the “Best” Root Attribute

- **Objective**
 - * Construct a decision tree that is as small as possible (Occam’s Razor)
 - * Subject to: consistency with labels on training data
- **Obstacles**
 - * Finding the *minimal* consistent hypothesis (i.e., decision tree) is NP-hard (D’oh!)
 - * Recursive algorithm (*Build-DT*)
 - ⇒ A greedy heuristic search for a simple tree
 - ⇒ Cannot guarantee optimality (D’oh!)
- **Main Decision: Next Attribute to Condition On**
 - * Want: attributes that split examples into sets that are relatively pure in one label
 - * Result: closer to a leaf node
 - * Most popular heuristic



Entropy: Intuitive Notion

- **A Measure of Uncertainty**
 - * **The Quantity**
 - ⇒ Purity: how close a set of instances is to having just one label
 - ⇒ Impurity (disorder): how close it is to total uncertainty over labels
 - * **The Measure: Entropy**
 - ⇒ Directly proportional to impurity, uncertainty, irregularity, surprise
 - ⇒ Inversely proportional to purity, certainty, regularity, redundancy
- **Example**
 - * For simplicity, assume $H = \{0, 1\}$, distributed according to $Pr(y)$
 - ⇒ Can have (more than 2) discrete class labels
 - ⇒ Continuous random variables: differential entropy
 - * **Optimal purity for y : either**
 - ⇒ $Pr(y = 0) = 1, Pr(y = 1) = 0$
 - ⇒ $Pr(y = 1) = 1, Pr(y = 0) = 0$
 - * **What is the least pure probability distribution?**
 - ⇒ $Pr(y = 0) = 0.5, Pr(y = 1) = 0.5$
 - ⇒ Corresponds to maximum impurity/uncertainty/irregularity/surprise
 - ⇒ Property of entropy: concave function (“concave downwards”)





Entropy: Information Theoretic Definition

- **Components**
 - * D : a set of examples $\{ \langle x_1, c(x_1) \rangle, \langle x_2, c(x_2) \rangle, \dots, \langle x_m, c(x_m) \rangle \}$
 - * $p_+ = Pr(c(x) = +)$, $p_- = Pr(c(x) = -)$
- **Definition**
 - * H is defined over a probability density function p
 - * D contains examples whose frequency of + and - labels indicates p_+ and p_- for the observed data
 - * The entropy of D relative to c is:

$$H(D) \equiv -p_+ \log_b(p_+) - p_- \log_b(p_-)$$
- **What Units is H Measured In?**
 - * Depends on the base b of the log (bits for $b = 2$, nats for $b = e$, etc.)
 - * A single bit is required to encode each example in the worst case ($p_+ = 0.5$)
 - * If there is less uncertainty (e.g., $p_+ = 0.8$), we can use less than 1 bit each

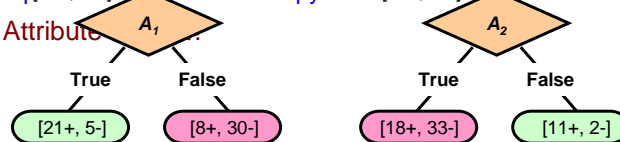


Information Gain: Information Theoretic Definition

- **Partitioning on Attribute Values**
 - * Recall: a partition of D is a collection of disjoint subsets whose union is D
 - * Goal: *measure the uncertainty removed by splitting on the value of attribute A*
- **Definition**
 - * The information gain of D relative to attribute A is the expected reduction in entropy due to splitting ("sorting") on A :

$$Gain(D, A) \equiv -H(D) - \sum_{v \in \text{values}(A)} \left[\frac{|D_v|}{|D|} \cdot H(D_v) \right]$$
 - where D_v is $\{x \in D : x.A = v\}$, the set of examples in D where attribute A has value v
 - * Idea: partition on A : scale entropy to the size of each subset D_v

Which Attribute

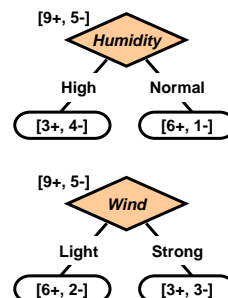




Constructing A Decision Tree for *PlayTennis* using ID3 [1]

Selecting The Root Attribute

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
1	Sunny	Hot	High	Light	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Light	Yes
4	Rain	Mild	High	Light	Yes
5	Rain	Cool	Normal	Light	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Light	No
9	Sunny	Cool	Normal	Light	Yes
10	Rain	Mild	Normal	Light	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Light	Yes
14	Rain	Mild	High	Strong	No



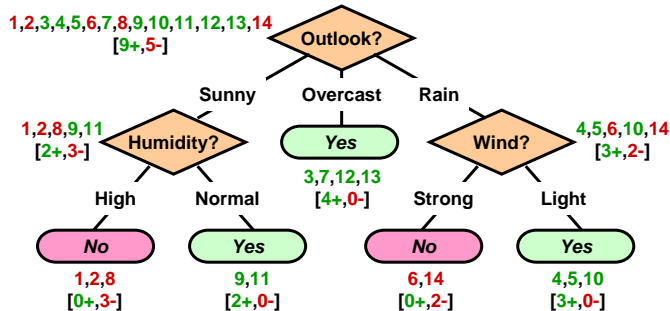
Prior (unconditioned) distribution: 9+, 5-

- * $H(D) = -(9/14) \lg(9/14) - (5/14) \lg(5/14)$ bits = 0.94 bits
- * $H(D, \text{Humidity} = \text{High}) = -(3/7) \lg(3/7) - (4/7) \lg(4/7)$ = 0.985 bits
- * $H(D, \text{Humidity} = \text{Normal}) = -(6/7) \lg(6/7) - (1/7) \lg(1/7)$ = 0.592 bits
- * $\text{Gain}(D, \text{Humidity}) = 0.94 - (7/14) * 0.985 + (7/14) * 0.592 = 0.151$ bits
- * Similarly, $\text{Gain}(D, \text{Wind}) = 0.94 - (8/14) * 0.811 + (6/14) * 1.0 = 0.048$ bits



Constructing A Decision Tree for *PlayTennis* using ID3 [2]

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
1	Sunny	Hot	High	Light	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Light	Yes
4	Rain	Mild	High	Light	Yes
5	Rain	Cool	Normal	Light	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Light	No
9	Sunny	Cool	Normal	Light	Yes
10	Rain	Mild	Normal	Light	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Light	Yes
14	Rain	Mild	High	Strong	No





Summary Points

- Taxonomies of Learning
- Definition of Learning: Task, Performance Measure, Experience
- Concept Learning as Search through H
 - * Hypothesis space H as a state space
 - * Learning: finding the correct hypothesis
- General-to-Specific Ordering over H
 - * Partially-ordered set: Less-Specific-Than (More-General-Than) relation
 - * Upper and lower bounds in H
- Version Space Candidate Elimination Algorithm
 - * S and G boundaries characterize learner's uncertainty
 - * Version space can be used to make predictions over unseen cases
- Learner Can Generate Useful Queries
- Next Tuesday: When and Why Are Inductive Leaps Possible?



Terminology

- Supervised Learning
 - * Concept - function from observations to categories (so far, boolean-valued: +/-)
 - * Target (function) - true function f
 - * Hypothesis - proposed function h believed to be similar to f
 - * Hypothesis space - space of all hypotheses that can be generated by the learning system
 - * Example - tuples of the form $\langle x, f(x) \rangle$
 - * Instance space (aka example space) - space of all possible examples
 - * Classifier - discrete-valued function whose range is a set of class labels
- The Version Space Algorithm
 - * Algorithms: *Find-S*, *List-Then-Eliminate*, candidate elimination
 - * Consistent hypothesis - one that correctly predicts observed examples
 - * Version space - space of all currently consistent (or *satisfiable*) hypotheses
- Inductive Learning

