Inductive Learning or Concept Learning

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

Inductive learning: system tries to induce a "general rule" from a set of observed instances.

Supervised learning: learning algorithm is given the correct value of the function for particular inputs, and changes its representation of the function to try to match the information provided by the feedback.

An **example** is a pair $(x, f(x))$, where x is the input and $f(x)$ is the output of the function applied to x.

Example: Work or Play?

outlook	*temp*	*humidity*	*windy*	**Saturday plan**
sunny	hot	high	false	cs472
sunny	hot	high	true	cs472
overcast	hot	high	false	soccer
rain	mild	high	false	soccer
rain	cool	normal	false	soccer
rain	cool	normal	true	cs472

- Each input observation, x, is a Satur*day*, described by the features *outlook, temp, humidity, windy*
- The **target concept**, f: *day* $\rightarrow$ {soccer, cs472}

Classification Tasks

Learning a discrete-valued function is called **classification**.

Steering a vehicle: image in windshield → direction to turn the wheel

Medical diagnosis: patient symptoms → has disease/ does not have disease

Forensic hair comparison: image of two hairs → match or not

Stock market prediction: closing price of last few days → market will go up or down tomorrow

Noun phrase coreference: description of two noun phrases in a document → do they refer to the same real world entity

Building Classifiers

1. Learn about the domain, write a program that maps inputs to outputs (eg., rule-based medical diagnosis systems).
2. Automate the process using data in the form of observations $(x_i, f(x_i))$.
   cholesterol=170,bp=170/95,... → heart disease = N
   cholesterol=250,bp=170/95,... → heart disease = Y

Inductive Learning

Given: collection of examples

Return: a function h (*hypothesis*) that approximates f (*target concept*).

OR

Given: a universe of objects described by a collection of attributes each labeled with one of a discrete number of classes

Return: a classification "rule" that can determine the class of any object from its attributes' values

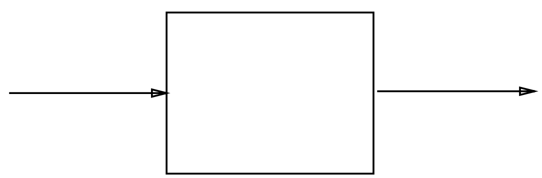
Inductive learning hypothesis: any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over any other unobserved examples.

Assumptions for Inductive Learning Algorithms:

- The training sample represents the population
- The input features permit discrimination

Inductive Learning

System tries to induce a general rule from a set of observed instances.



The *hypothesis* produced is sometimes called the *concept description* — essentially a program that can be used to classify subsequent instances.

k-nearest neighbor

Also called instance-based Learning; case-based learning.

A: set of features/attributes, $A_1, \ldots, A_n$ that describe the problem

$x = x_{a_1} x_{a_2} \ldots x_{a_n}$, where x_{a_i} is the value of feature A_i in example x

$f(x) : x \rightarrow c \epsilon C = \{c_1, \ldots, c_m\}$

The *case base* is the set of training examples $(x_1, f(x_1)), (x_2, f(x_2)), \ldots$

k-nearest neighbor algorithm for computing $f(x)$:

1. Compare new example, x, to each case, y, in the case base and calculate for each pair:

$$sim(x,y) = \sum_{i=1}^{n} match(x_{a_i}, y_{a_i})$$

where $match(a,b)$ is a function that returns 1 if a and b are equal and 0 otherwise.

2. Let $R =$ the top k cases ranked according to sim
3. Return as $f(x)$ the class, c, that wins the majority vote among $f(R_1), f(R_2), \ldots, f(R_{|k|})$. Handle ties randomly.

Types of Attributes

1. Symbolic (nominal) – $EyeColor \in \{brown, blue, green\}$
2. Boolean – $anemic \in \{TRUE, FALSE\}$
3. Numeric (Integer, Real) – $age \in [0, 105]$

How do we compute the similarity between $EyeColor = brown$ and $EyeColor = green$?

Example of case retrieval for k-nn

outlook	temp	humidity	windy	plan	*sim*
sunny	hot	high	false	cs472	
sunny	hot	high	true	cs472	
overcast	hot	high	false	soccer	
overcast	mild	normal	true	football	
rain	mild	high	false	soccer	
rain	cool	normal	false	soccer	

A: outlook, temp, humidity, windy

$k = 1$, $C =$ {soccer, cs472 football}

test case: $X =$ sunny cool high false

k-Nearest Neighbor Algorithm

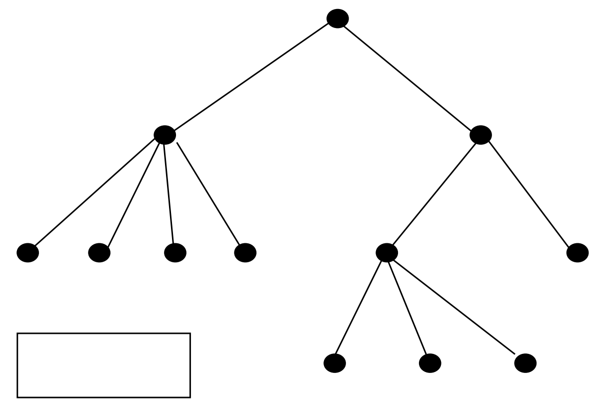
1. Memorizes all observed instances and their class
2. Is this rote learning?
3. Is this really learning?
4. When does the induction take place?

Advantages and Disadvantages

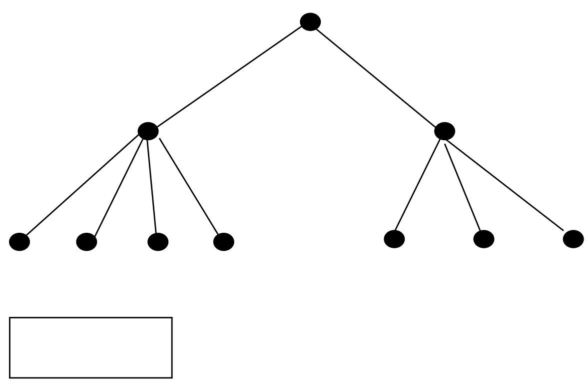
What constitutes the concept description?

Poisonous Mushroom Decision Tree

Concept description: decision trees



Another Poisonous Mushroom Decision Tree?



Finding a Decision Tree

Goal: find the best decision tree
where *best* means the smallest tree consistent with data

Ockham's Razor: all other things being equal, choose the simplest

Problem: goal is computationally intractable

Solution: use heuristic search

Top Down Induction of Decision Trees

If all instances from same class
  then tree is leaf with that class name
  else
    pick test for decision node
    partition instances by test outcome
    construct one branch for each possible outcome
    build subtrees recursively

Example

CS Major Database

Height	Eyes	Class
short	brown	hacker
tall	blue	theoretician
tall	brown	hacker
short	blue	theoretician

A Concept Learning Task

Day	Outlook	Temp	Humidity	Wind	Play-Tennis?
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Characteristics of Tests

Let $\mid P \mid = 20, \mid N \mid = 20$

A Boolean test splits the data into two subsets, U_1 and U_2

The best test: $U_1 = P$ and $U_2 = N$

The worst test: $U_1 = \frac{1}{2}P + \frac{1}{2}N$ and $U_2 = \frac{1}{2}P + \frac{1}{2}N$

Information Gain

average disorder =

$$\sum_{b=1}^{nbranches} \frac{n_b}{n_t} * Disorder(b)$$

average disorder =

$$\sum_{b=1}^{nbranches} \frac{n_b}{n_t} * (\sum_{c}^{nclasses} -\frac{n_{bc}}{n_b} \log_2(\frac{n_{bc}}{n_b}))$$

n_b is the number of instances in branch b

n_t is the total number of instances

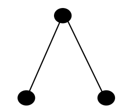
n_{bc} is the number of instances in branch b of class c

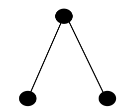
Disorder Term

Disorder = $(\sum_c^{nclasses} -\frac{n_{bc}}{n_b} \log_2(\frac{n_{bc}}{n_b}))$

Average disorder =

$$\sum_{b=1}^{nbranches} \frac{n_b}{n_t} * disorder(b)$$





Calculation for Attribute Humidity

branch	value	n_{bp}	n_{bn}	disorder
1	high	3	4	.99
2	normal	6	1	.58

Disorder(high) = $-\frac{3}{7}log_2(\frac{3}{7}) - \frac{4}{7}log_2(\frac{4}{7}) = .99$

Disorder(normal)= $-\frac{6}{7}log_2(\frac{6}{7}) - \frac{1}{7}log_2(\frac{1}{7}) = .58$

Average Disorder of Humidity =

$\frac{7}{14}Disorder(high) + \frac{7}{14}Disorder(normal) =$

$\frac{7}{14}(.99) + \frac{7}{14}(.58) = .79$

Selection of Attribute

Attribute	Average Disorder
outlook	0.69
temperature	0.91
humidity	0.79
windy	0.89

Information Gain and Entropy

- S is a sample of training examples
- p is the proportion of positive examples in S
- n is the proportion of negative examples in S
- Entropy (our *Disorder*) measures the impurity of S

$$Entropy(S) \equiv -p \log_2 p - n \log_2 n$$

Information Gain measures the expected reduction in entropy caused by partitioning the examples according to attribute A.

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|\, S_v \,|}{|\, S \,|} Entropy(S_v)$$

Decision Trees

Goal: Construct a decision tree that agrees (is consistent) with the training set.

Trivial solution: construct a decision tree that has one path to a leaf for every example.

Problem with trivial solution?

Non-trivial solution: find a concise decision tree that agrees with the training data.

Appropriate Problems for Decision Tree Learning

- Instances represented by attribute-value pairs
- Target function has a discrete number of output values
- Disjunctive descriptions may be required