CSC 480: Artificial Intelligence

Franz J. Kurfess

Visiting Professor
Department of Computer Science and Mathematics
Munich University of Applied Sciences
Germany

Professor
Computer Science Department
California Polytechnic State University
San Luis Obispo, CA, U.S.A.

Course Overview

- **Introduction**
- **Intelligent Agents**
- **Search**
  - problem solving through search
  - uninformed search
  - informed search
- **Games**
  - games as search problems
- **Knowledge and Reasoning**
  - reasoning agents
  - propositional logic
  - predicate logic
  - knowledge-based systems
- **Learning**
  - PAC learning
  - learning from observation
  - neural networks
- **Conclusions**

Chapter Overview
Learning

- **Motivation**
- **Objectives**
- **Learning from Observation**
  - Learning Agents
  - Inductive Learning
  - Learning Decision Trees
- **Computational Learning Theory**
  - Probably Approximately Correct (PAC) Learning
- **Learning in Neural Networks**
  - Neurons and the Brain
  - Neural Networks
  - Perceptrons
  - Multi-layer Networks
  - Deep Learning
  - Applications
- **Important Concepts and Terms**
- **Chapter Summary**

Motivation

- **learning is important for agents**
  - unknown environments
  - changes
  - performance improvement
- **the capability to learn is essential for the autonomy of an agent**
  - flexible decision making
- **learning vs. knowledge transfer**
  - training an agent via examples can be more efficient, but less transparent
  - extraction of knowledge from the examples, and transfer to the agent
- **agents capable of learning can improve their performance**
  - but may require experimentation

Objectives

- be aware of the necessity of learning for autonomous agents
- understand the basic principles and limitations of inductive learning from examples
- apply decision tree learning to deterministic problems characterized by Boolean functions
- understand the basic learning methods of perceptrons and multi-layer neural networks
- know the main advantages and problems of learning in neural networks

Learning

- **an agent tries to improve its behavior through**
  - observation
    - learning from experience
      - memorization of past percepts, states, and actions
      - generalizations, identification of similar experiences
    - forecasting
      - prediction of changes in the environment
  - reasoning
    - performance improvement through generation of new knowledge
    - theories
      - generation of complex models based on observations and reasoning
  - reflection
    - analysis of past behavior

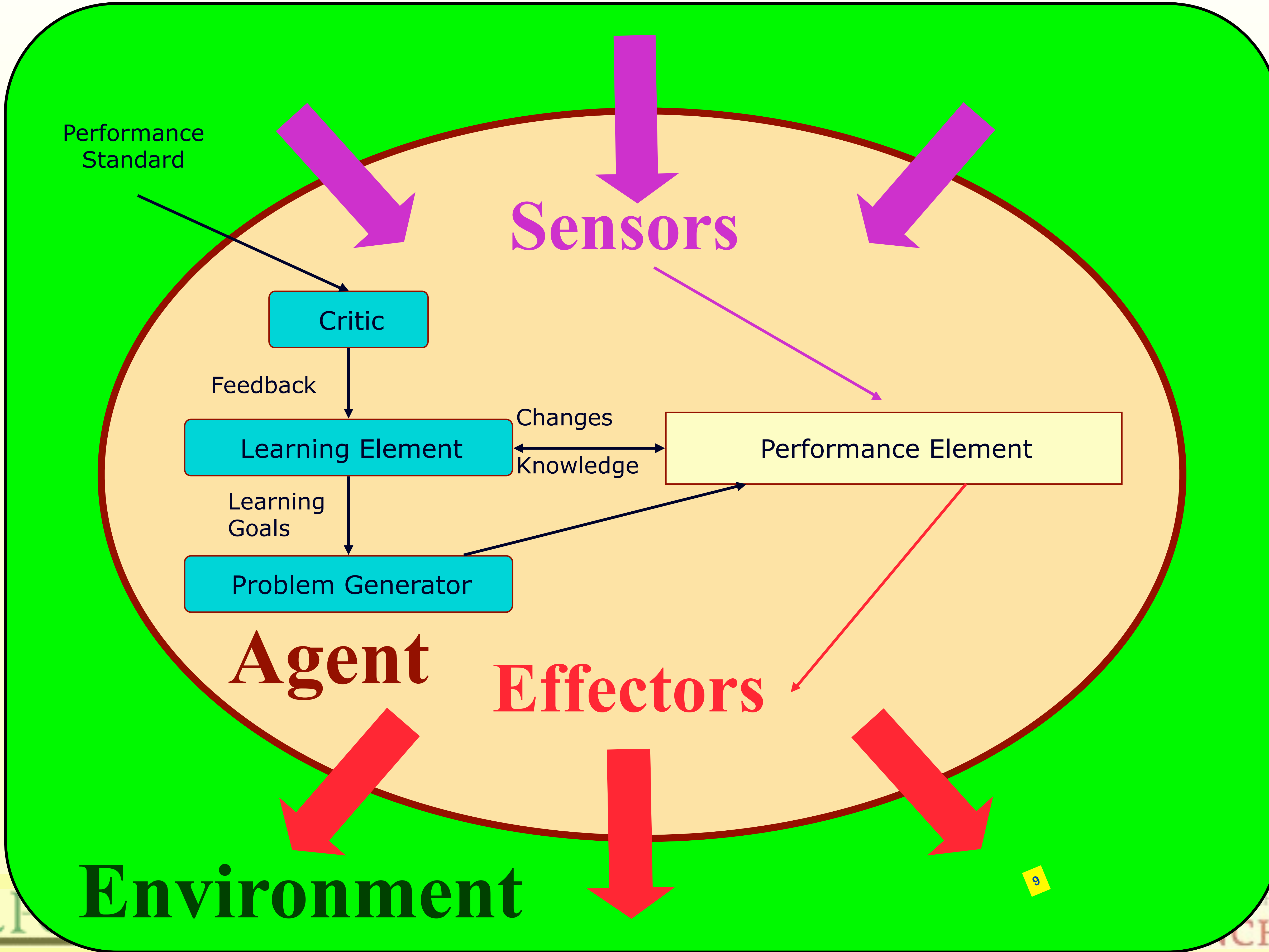
Learning from Observation

Learning Agents
Inductive Learning
Learning Decision Trees

Learning Agents

- **based on previous agent designs, such as reflexive, model-based, goal-based agents**
  - those aspects of agents are encapsulated into the performance element of a learning agent
- **a learning agent has an additional learning element**
  - usually used in combination with a critic and a problem generator for better learning
- **most agents learn from examples**
  - inductive learning

Learning Agent Model



Forms of Learning

- **supervised learning**
  - an agent tries to find a function that matches examples from a sample set
    - each example provides an input together with the correct output
  - a teacher provides feedback on the outcome
    - the teacher can be an outside entity, or part of the environment
- **un-supervised learning**
  - the agent tries to learn from patterns without corresponding output values
- **reinforcement learning**
  - the agent does not know the exact output for an input, but it receives feedback on the desirability of its behavior
    - the feedback can come from an outside entity, the environment, or the agent itself
    - the feedback may be delayed, and not follow the respective action immediately

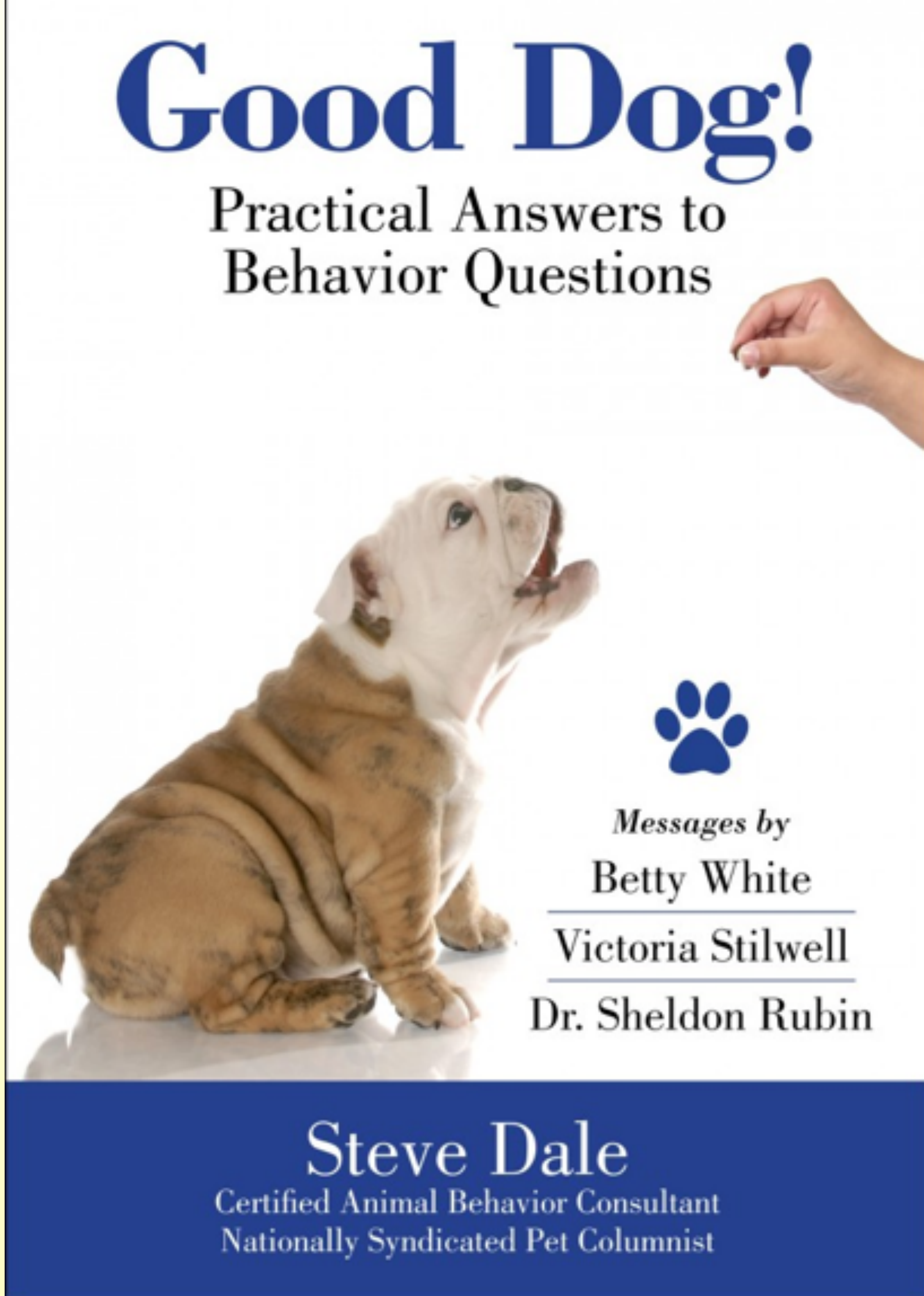
Feedback

- **provides information about the actual outcome of actions**
- **supervised learning**
  - both the input and the output of a component can be perceived by the agent directly
  - the output may be provided by a teacher
- **reinforcement learning**
  - feedback concerning the desirability of the agent's behavior is available
    - not in the form of the correct output, or
    - not immediately
  - may not be directly attributable to a particular action
    - feedback may occur only after a sequence of actions
  - the agent or component knows that it did something right (or wrong), but not what specific action caused it

Feedback: Good Dog!

- Note: This does not constitute an endorsement of the book - I have no idea if it's any good ...

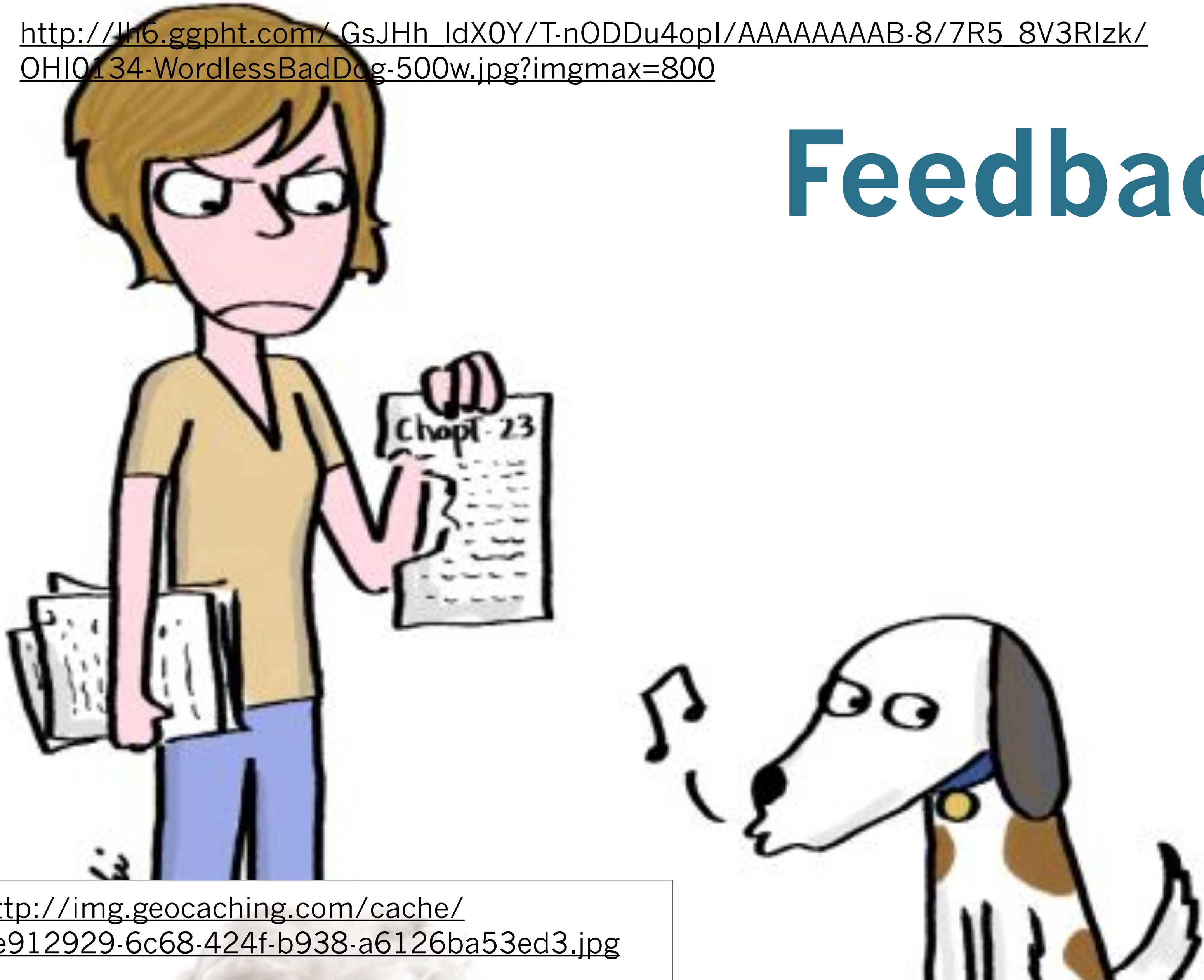
http://www.chicagonow.com/steve-dales-pet-world/files/2011/11/Good-Dog-624x936.jpg



http://blog.timesunion.com/bark/files/2013/09/Dog-training.jpg

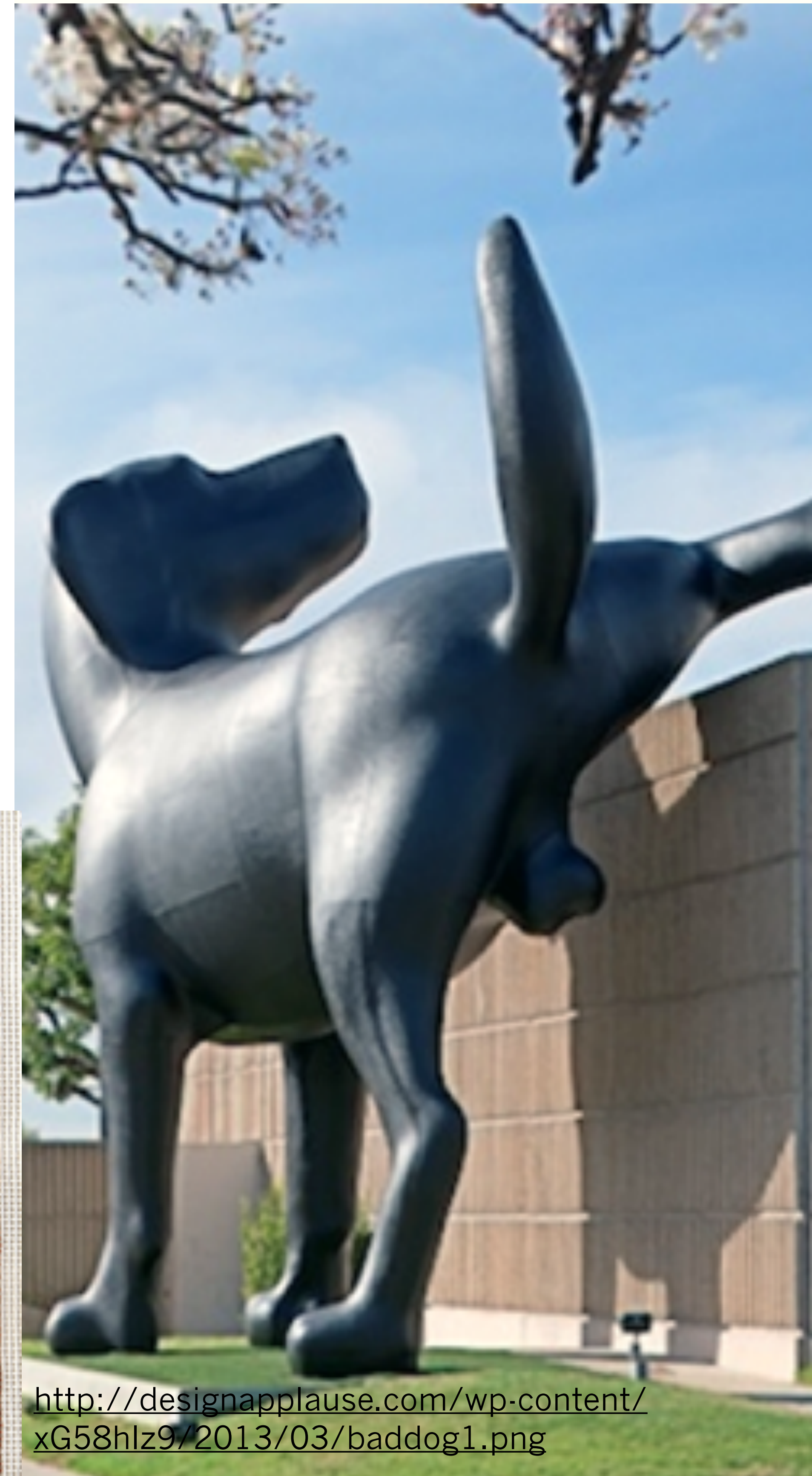
Feedback: Bad Dog!

http://lh6.ggpht.com/-GsJHh_IdX0Y/T-nODDu4opI/AAAAAAAAB-8/7R5_8V3RIzk/OHI0134-WordlessBadDog-500w.jpg?imgmax=800



http://img.geocaching.com/cache/ee912929-6c68-424f-b938-a6126ba53ed3.jpg

http://www.barbarabergstendesigns.com/S20%20No%20No%20Bad%20Dog%20Needlepoint%20Saying.jpg



http://designapplause.com/wp-content/xG58hlz9/2013/03/baddog1.png

Prior Knowledge

- **background knowledge available before a task is tackled**
- **can increase performance or decrease learning time considerably**
- **many learning schemes assume that no prior knowledge is available**
- **in reality, some prior knowledge is almost always available**
  - but often in a form that is not immediately usable by the agent

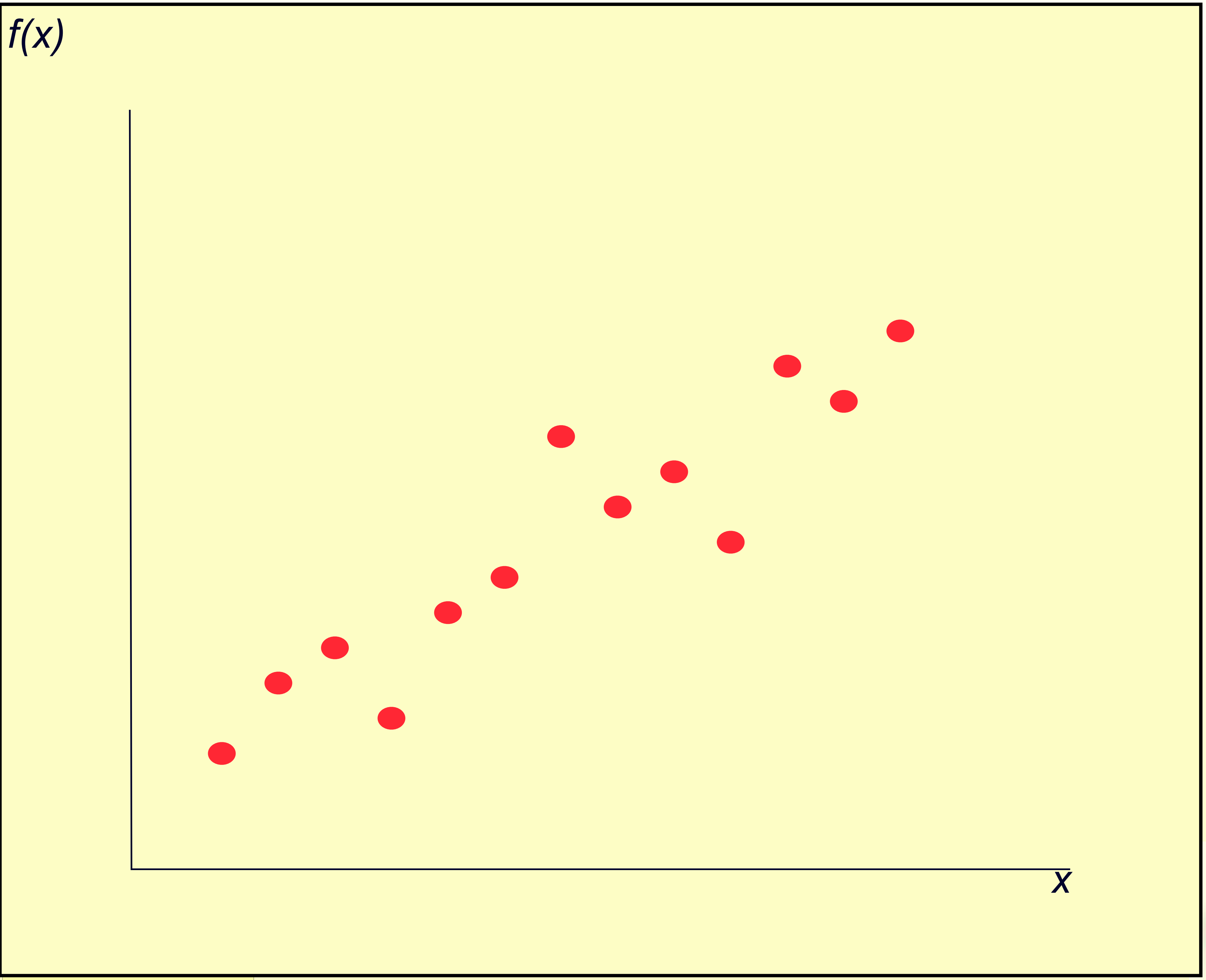
Inductive Learning

- **tries to find a function *h* (the *hypothesis*) that approximates a set of samples defining a function *f***
  - the samples are usually provided as input-output pairs *(x, f(x))*
- **supervised learning method**
- **relies on inductive inference, or induction**
  - conclusions are drawn from specific instances to more general statements

Hypotheses

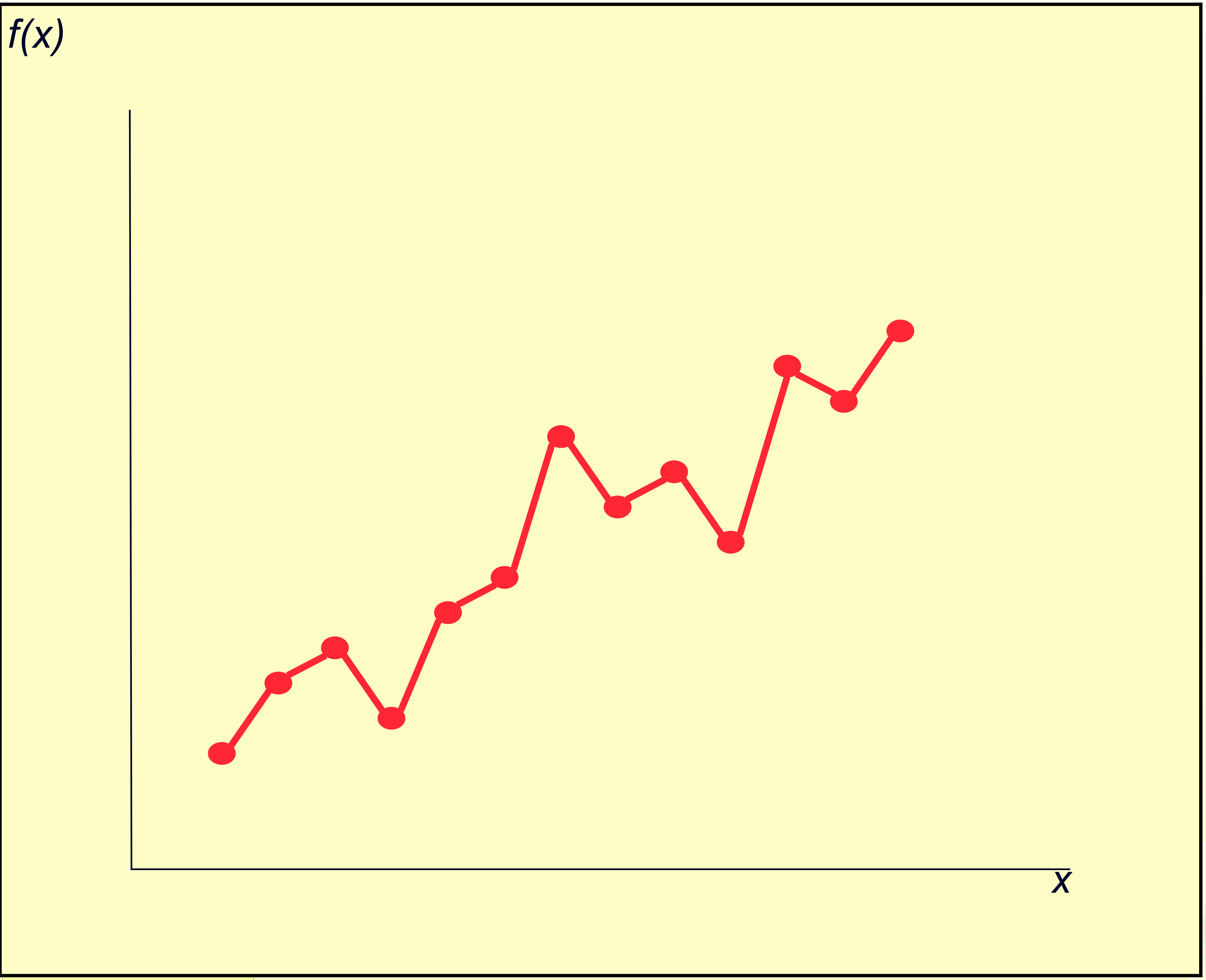
- **finding a suitable hypothesis can be difficult**
  - since the function *f* is unknown, it is hard to tell if the hypothesis *h* is a good approximation
- **the *hypothesis space* describes the set of hypotheses under consideration**
  - e.g. polynomials, sinusoidal functions, propositional logic, predicate logic, ...
  - the choice of the hypothesis space can strongly influence the task of finding a suitable function
  - while a very general hypothesis space (e.g. Turing machines) may be guaranteed to contain a suitable function, it can be difficult to find it
- **Ockham's razor: if multiple hypotheses are consistent with the data, choose the simplest one**

Example Inductive Learning 1



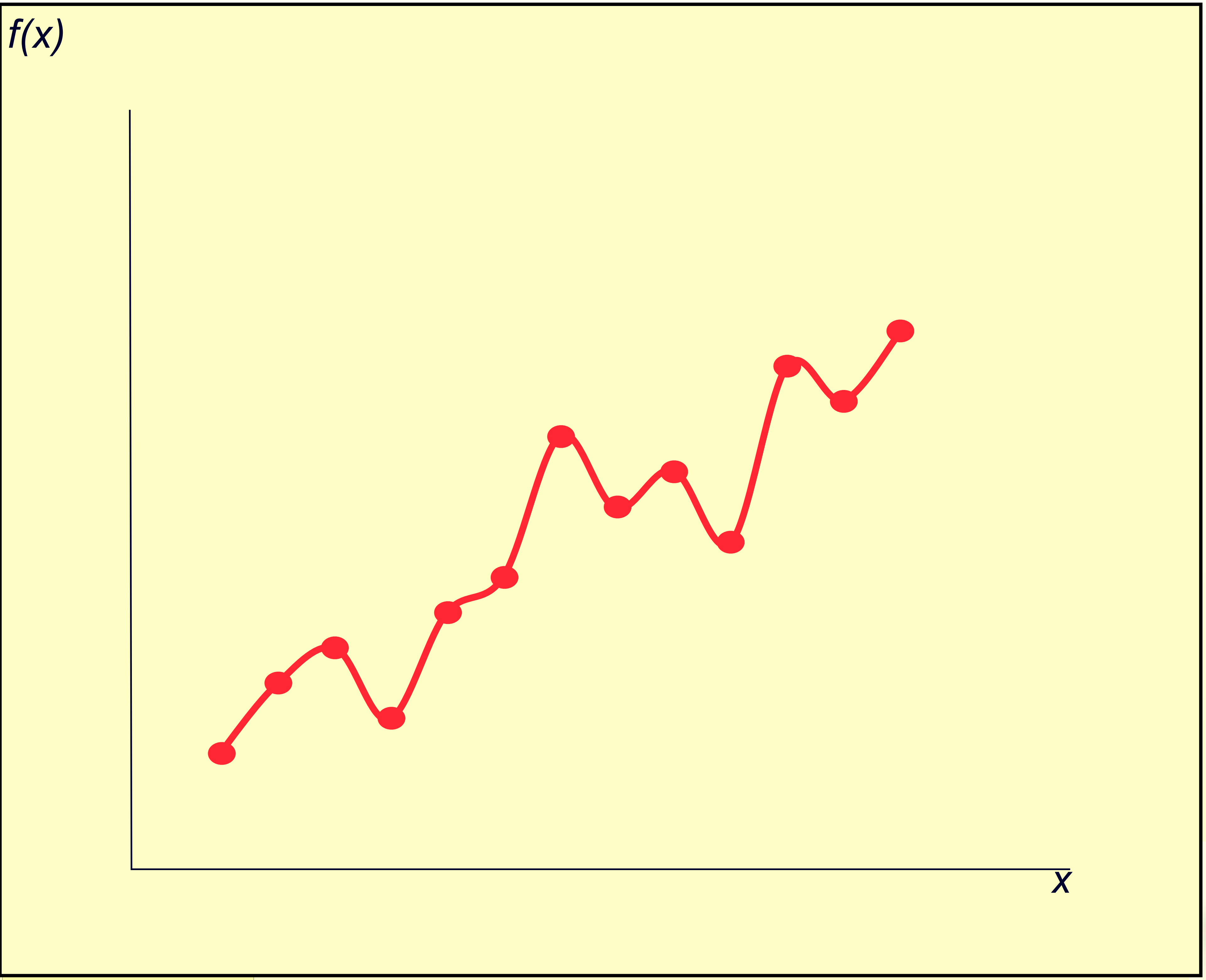
- **input-output pairs displayed as points in a plane**
- **the task is to find a hypothesis (function) that connects the points**
  - either all of them, or most of them
  - "close enough" often is sufficient
- **various performance measures**
  - number of points connected
  - minimal surface
  - lowest tension

Example Inductive Learning 2



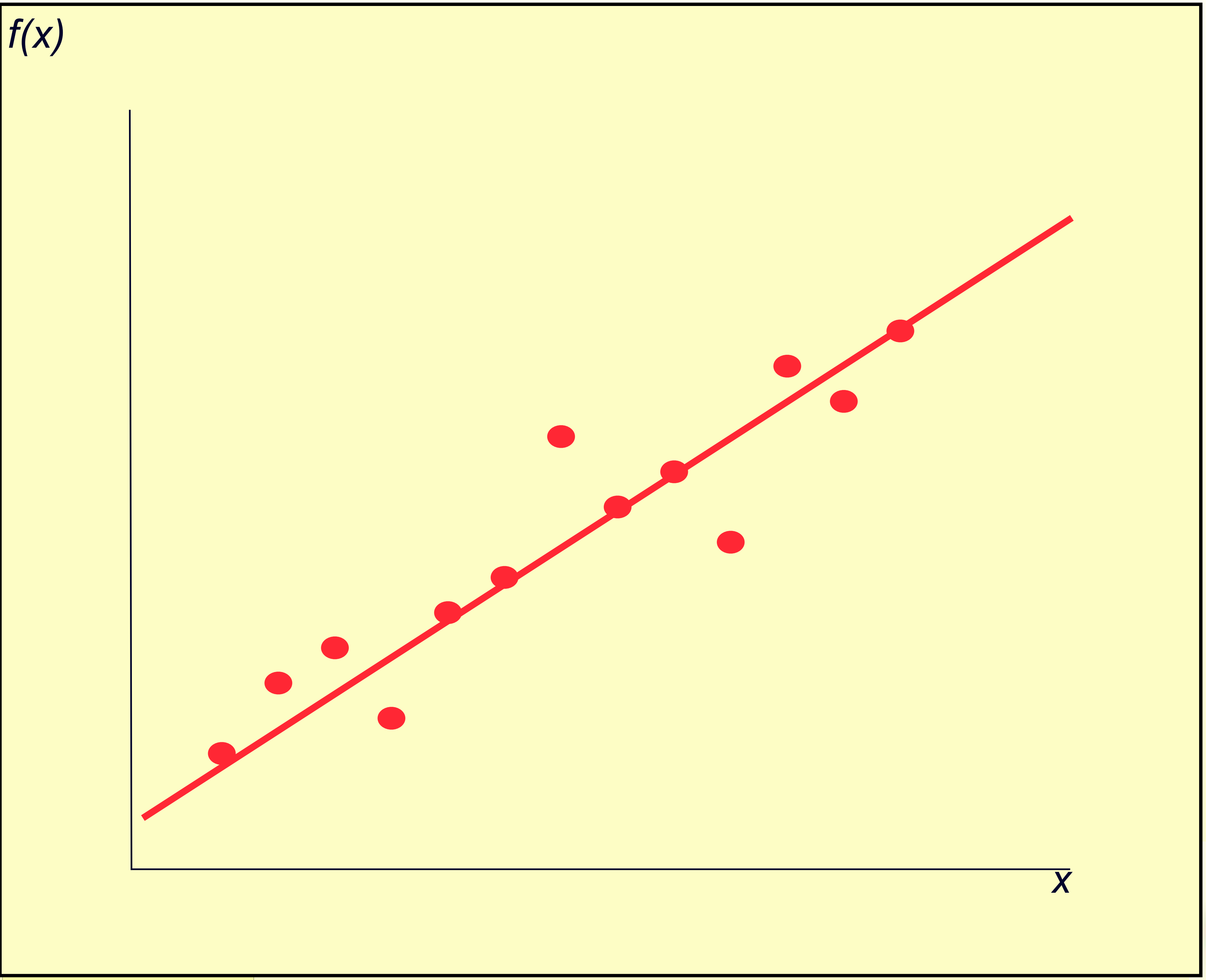
- hypothesis is a function consisting of linear segments
- fully incorporates all sample pairs
  - goes through all points
- very easy to calculate
- has discontinuities at the joints of the segments
- moderate predictive performance

Example Inductive Learning 3



- hypothesis expressed as a polynomial function
- incorporates all samples
- more complicated to calculate than linear segments
- no discontinuities
- better predictive power

Example Inductive Learning 4



- hypothesis is a linear function
- does not incorporate all samples
- extremely easy to compute
- low predictive power
  - unless the hidden function is linear

(Non-) monotonic

- **Lab 10 Submission: AI and Humor -> Marvin Minksy's Sense of Humor & Toons**
  - by Christina Taggart - Tuesday, November 20, 2012, 10:33 PM

http://aitopics.net/Nonmonotonicity

Learning and Decision Trees

- **based on a set of attributes as input, predicted output value, the *decision* is learned**
  - it is called *classification* learning for discrete values
  - *regression* for continuous values
- **Boolean or binary classification**
  - output values are true or false
  - conceptually the simplest case, but still quite powerful
- **making decisions**
  - a sequence of test is performed, testing the value of one of the attributes in each step
  - when a leaf node is reached, its value is returned
  - good correspondence to human decision-making

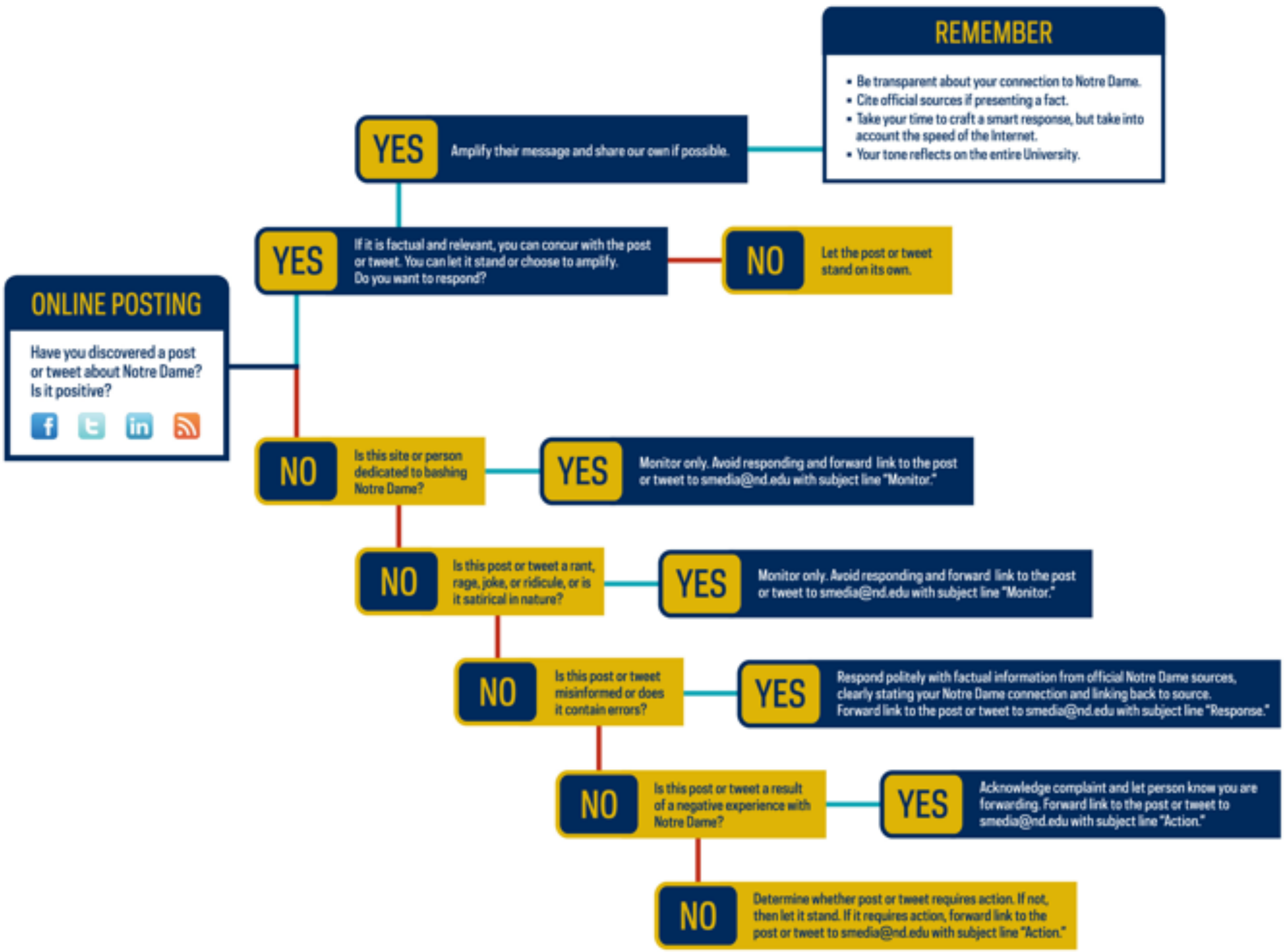
Boolean Decision Trees

- **compute yes/no decisions based on sets of desirable or undesirable properties of an object or a situation**
  - each node in the tree reflects one yes/no decision based on a test of the value of one property of the object
    - the root node is the starting point
    - leaf nodes represent the possible final decisions
  - branches are labeled with possible values
- **the learning aspect is to predict the value of a *goal predicate* (also called goal concept)**
  - a hypothesis is formulated as a function that defines the goal predicate

Decision Tree Examples

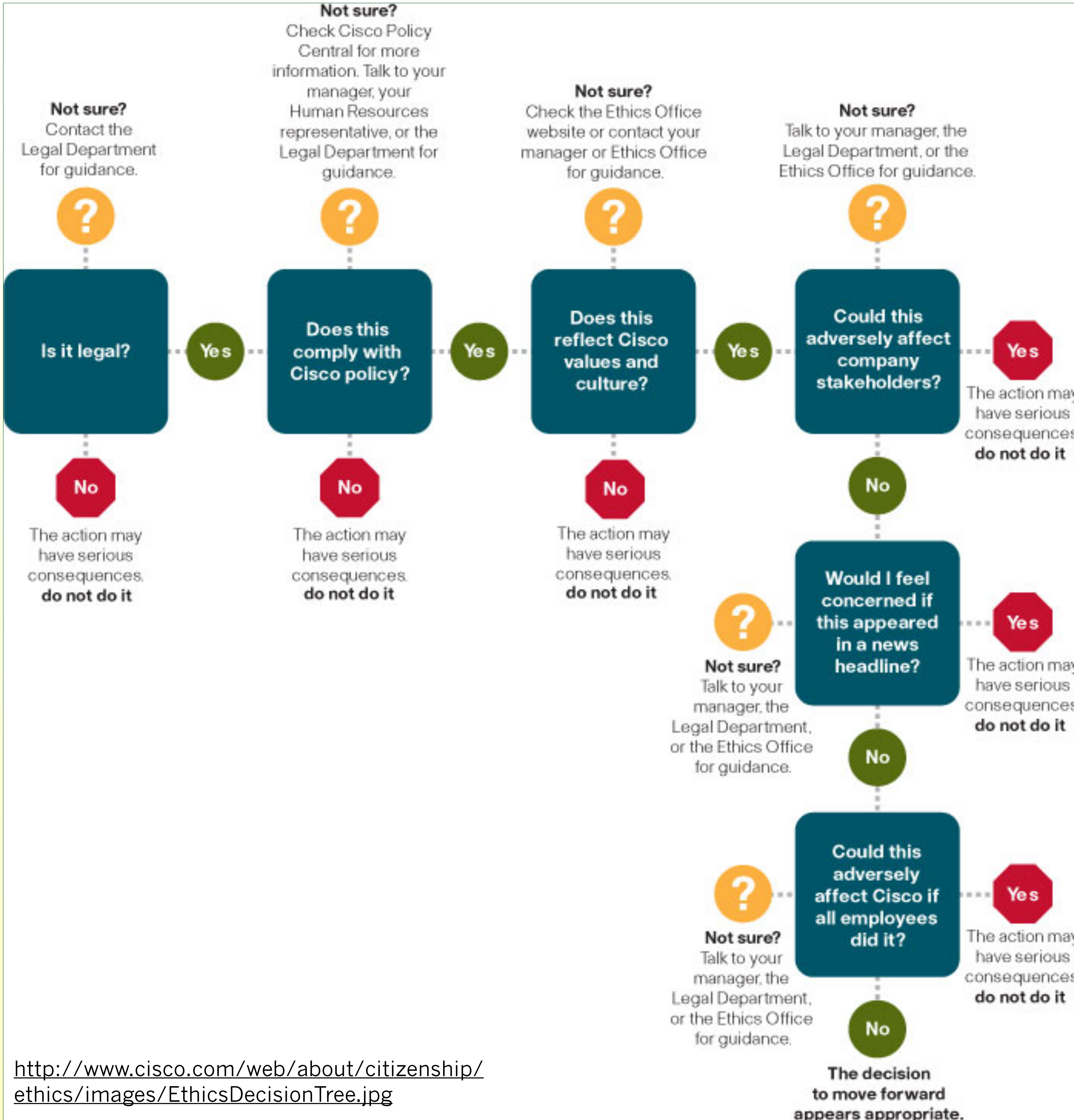
- U of Notre Dame "Online Postings" decision tree
- Cisco Ethics Decision Tree
- TreePlan Decision Tree Learning for MS Excel

U of Notre Dame Online Postings Decision Tree



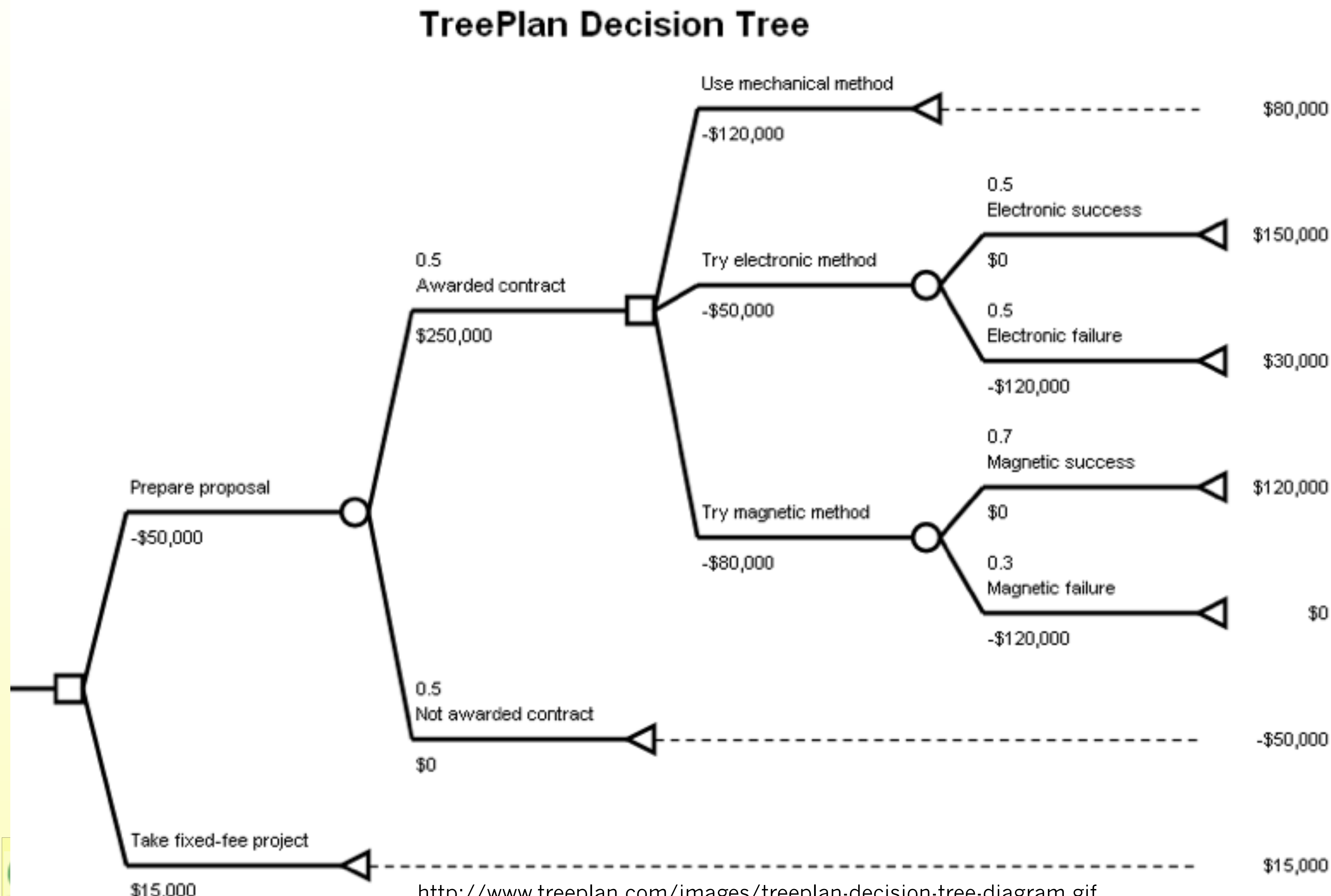
http://uc.nd.edu/assets/103716/original/nd_social_media_decision_tree_lg.jpg

Cisco Ethics Decision Tree



http://www.cisco.com/web/about/citizenship/ethics/images/EthicsDecisionTree.jpg

TreePlan Decision Tree Learning for MS Excel



http://www.treeplan.com/images/treeplan-decision-tree-diagram.gif

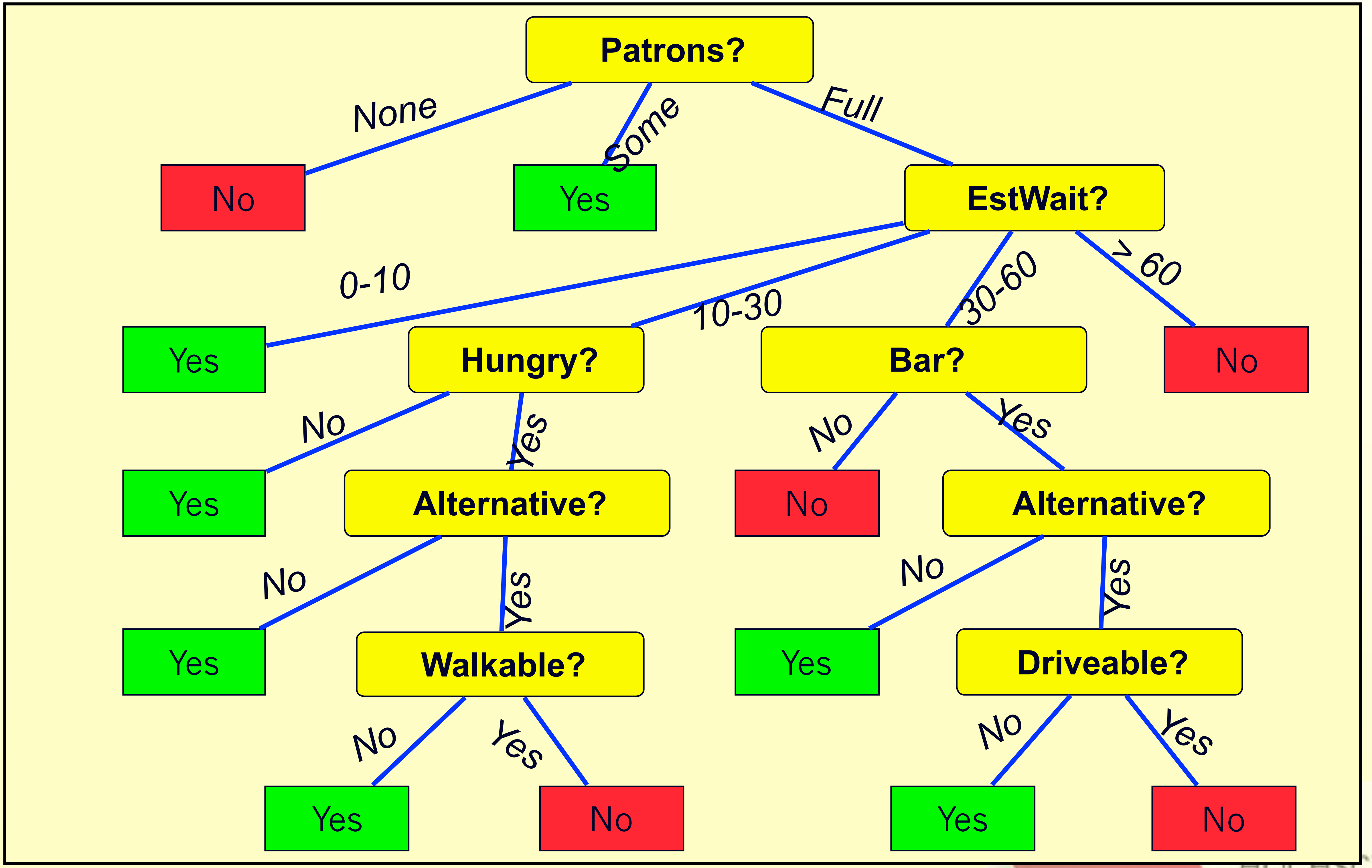
Terminology

- **example or sample**
  - describes the values of the attributes and the goal
    - a positive sample has the value true for the goal predicate, a negative sample false
- **sample set**
  - collection of samples used for training and validation
- **training**
  - the training set consists of samples used for constructing the decision tree
- **validation**
  - the test set is used to determine if the decision tree performs correctly
    - ideally, the test set is different from the training set

Restaurant Sample Set

Example					Attributes						Goal	Exar
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*	
X1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes	X1
X2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No	X2
X3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes	X3
X4	Yes	No	Yes	Yes	Full	$	No	No	Thai	10-30	Yes	X4
X5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No	X5
X6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes	X6
X7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No	X7
X8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes	X8
X9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No	X9
X10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No	X10
X11	No	No	No	No	None	$	No	No	Thai	0-10	No	X11
X12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes	X12

Decision Tree Example



To wait, or not to wait?

Expressiveness of Decision Trees

- **decision trees can also be expressed in logic as implication sentences**
- **in principle, they can express propositional logic sentences**
  - each row in the truth table of a sentence can be represented as a path in the tree
  - often there are more efficient trees
- **some functions require exponentially large decision trees**
  - parity function, majority function

Learning Decision Trees

- **problem: find a decision tree that agrees with the training set**
- **trivial solution: construct a tree with one branch for each sample of the training set**
  - works perfectly for the samples in the training set
  - may not work well for new samples (generalization)
  - results in relatively large trees
- **better solution: find a concise tree that still agrees with all samples**
  - corresponds to the simplest hypothesis that is consistent with the training set

Ockham's Razor

The most likely **hypothesis** is the **simplest** one that is **consistent** with all **observations**.

- general principle for inductive learning
- a simple hypothesis that is consistent with all observations is more likely to be correct than a complex one
- question: How does one measure the simplicity of a hypothesis?

William of Ockham (Occam)

http://upload.wikimedia.org/wikipedia/commons/7/70/William_of_Ockham.png

CAL POLY

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

Constructing Decision Trees

- **in general, constructing the smallest possible decision tree is an intractable problem**
- **algorithms exist for constructing reasonably small trees**
- **basic idea: test the most important attribute first**
  - attribute that makes the most difference for the classification of an example
    - can be determined through information theory
  - hopefully will yield the correct classification with few tests

Decision Tree Implementations

- **Weka tool set**
  - Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.
- **R programming language**
- **Orange**
  - component-based data mining and machine learning software suite
  - visual programming front-end for explorative data analysis and visualization
  - Python bindings and libraries for scripting

WEKA
The University of Waikato

http://www.cs.waikato.ac.nz/ml/weka/citing.html

Decision Tree Algorithm

- **recursive formulation**
  - select the best attribute to split positive and negative examples
  - if only positive or only negative examples are left, we are done
  - if no examples are left, no such examples were observed
    - return a default value calculated from the majority classification at the node's parent
  - if we have positive and negative examples left, but no attributes to split them, we are in trouble
    - samples have the same description, but different classifications
    - may be caused by incorrect data (noise), or by a lack of information, or by a truly non-deterministic domain

Restaurant Example
Decision Tree Learning

data set

human approach

human-generated decision tree

learning approach

algorithmically generated tree

Restaurant Sample Set

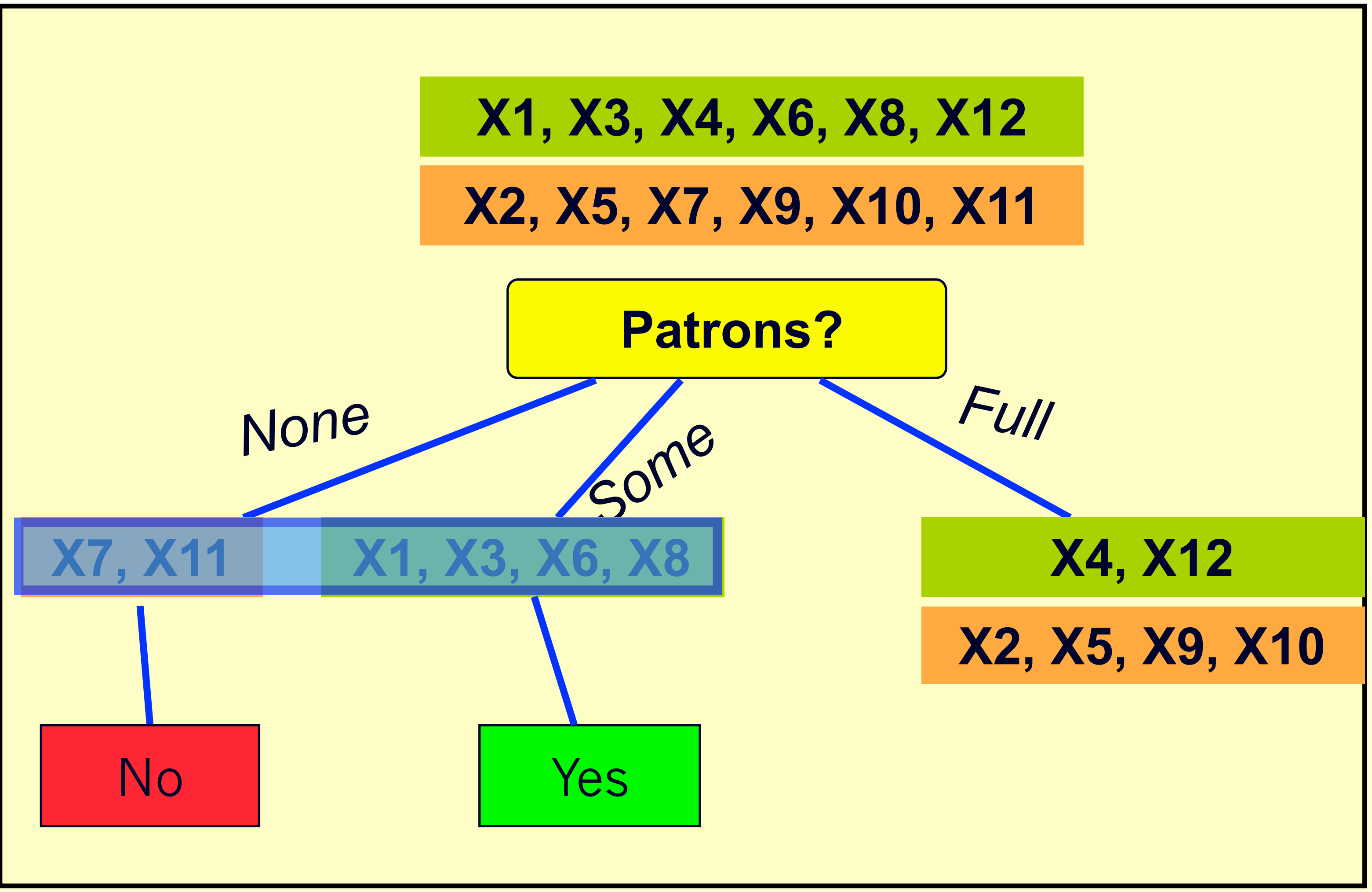
Example				Attributes						Goal	Exa
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*	
X1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes	X1
X2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No	X2
X3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes	X3
X4	Yes	No	Yes	Yes	Full	$	No	No	Thai	10-30	Yes	X4
X5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No	X5
X6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes	X6
X7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No	X7
X8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes	X8
X9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No	X9
X10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No	X10
X11	No	No	No	No	None	$	No	No	Thai	0-10	No	X11
X12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes	X12

Restaurant Sample Set

Example				Attributes						Goal	Exa
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*	
X1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes	X1
X2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No	X2
X3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes	X3
X4	Yes	No	Yes	Yes	Full	$	No	No	Thai	10-30	Yes	X4
X5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No	X5
X6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes	X6
X7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No	X7
X8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes	X8
X9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No	X9
X10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No	X10
X11	No	No	No	No	None	$	No	No	Thai	0-10	No	X11
X12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes	X12

- **select best attribute**
  - candidate 1: *Pat* *Some* and *None* in agreement with goal
  - candidate 2: *Type* No values in agreement with goal

Partial Decision Tree



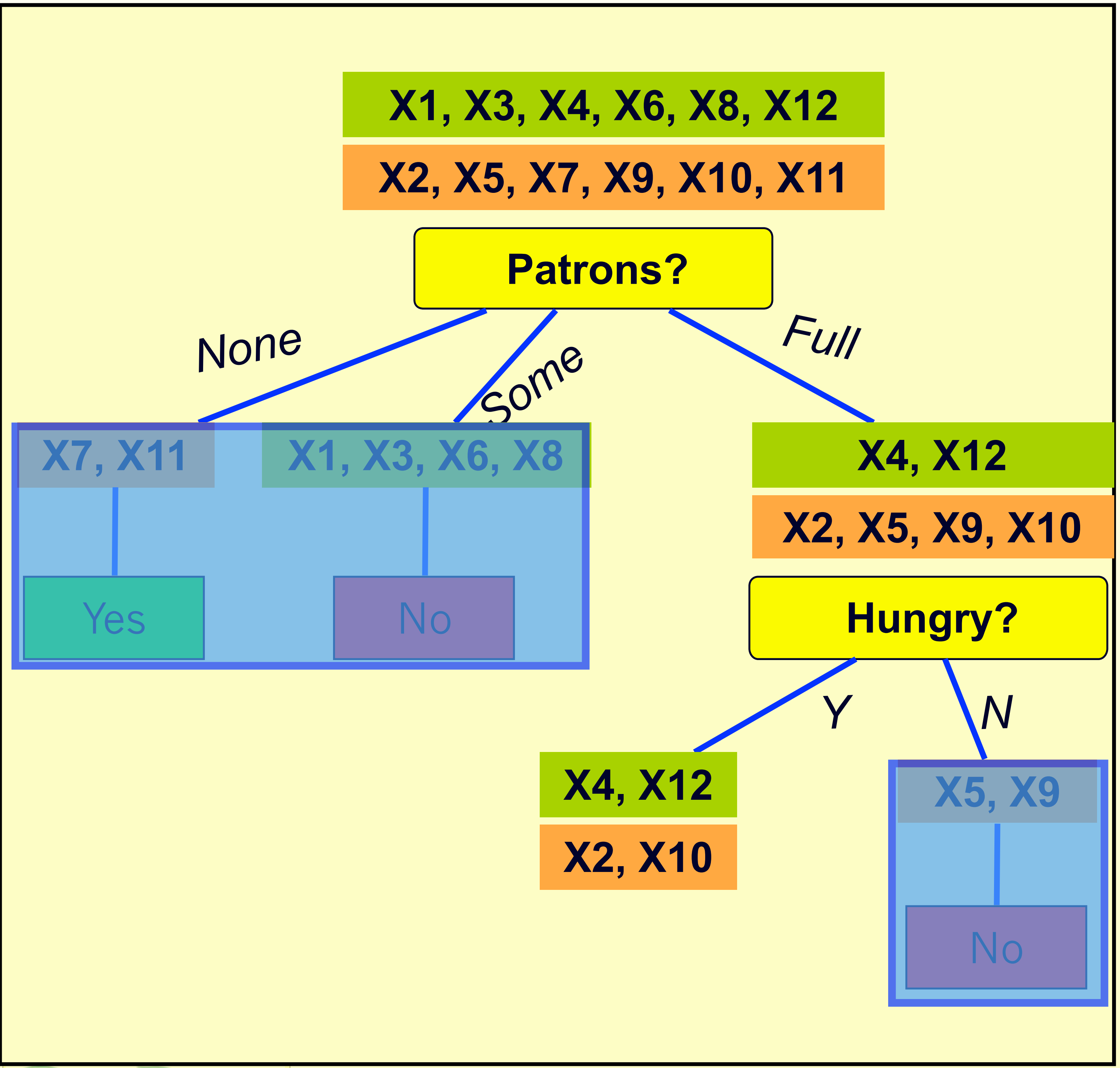
- *Patrons* **needs further discrimination only for the** ***Full*** **value**
- *None* **and** *Some* **agree with the** *WillWait* **goal predicate**
- **the next step will be performed on the remaining samples for the** ***Full*** **value of** ***Patrons***

Restaurant Sample Set

Example					Attributes						Goal	Exa
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*	
X1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes	X1
X2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No	X2
X3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes	X3
X4	Yes	No	Yes	Yes	Full	$	No	No	Thai	10-30	Yes	X4
X5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No	X5
X6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes	X6
X7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No	X7
X8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes	X8
X9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No	X9
X10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No	X10
X11	No	No	No	No	None	$	No	No	Thai	0-10	No	X11
X12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes	X12

- **select next best attribute**
  - candidate 1: *Hungry* *No* in agreement with goal
  - candidate 2: *Type* No values in agreement with goal

Partial Decision Tree



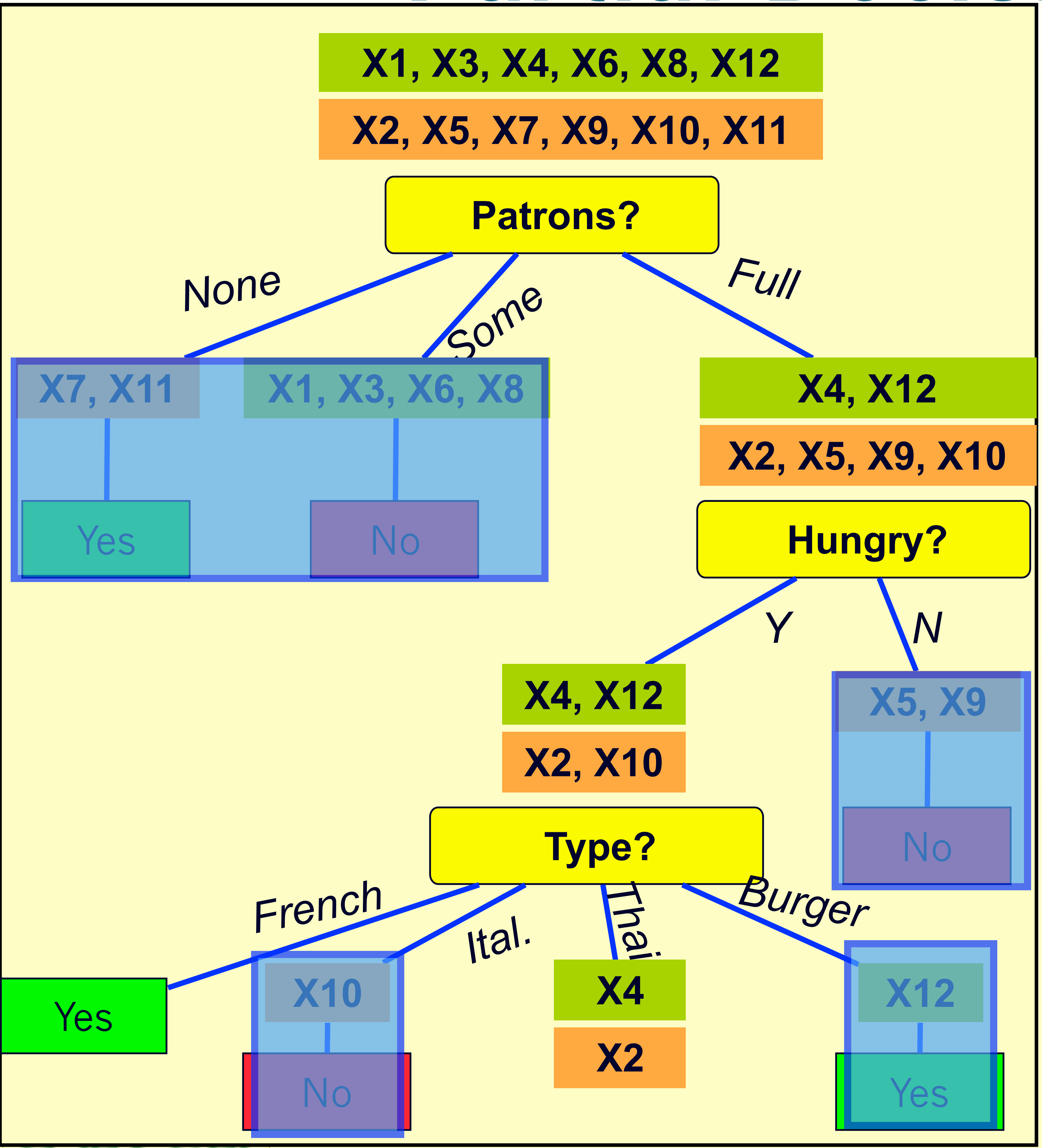
- *Hungry* **needs further discrimination only for the** *Yes* **value**
- *No* **agrees with the** *WillWait* **goal predicate**
- **the next step will be performed on the remaining samples for the** *Yes* **value of** *Hungry*

Restaurant Sample Set

Example					Attributes						Goal	Example
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*	
X1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes	X1
X2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No	X2
X3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes	X3
X4	Yes	No	Yes	Yes	Full	$	No	No	Thai	10-30	Yes	X4
X5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No	X5
X6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes	X6
X7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No	X7
X8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes	X8
X9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No	X9
X10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No	X10
X11	No	No	No	No	None	$	No	No	Thai	0-10	No	X11
X12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes	X12

- **select next best attribute**
  - candidate 1: *Type* *Italian, Burger* in agreement with goal
  - candidate 2: *Friday* *No* in agreement with goal

Partial Decision Tree



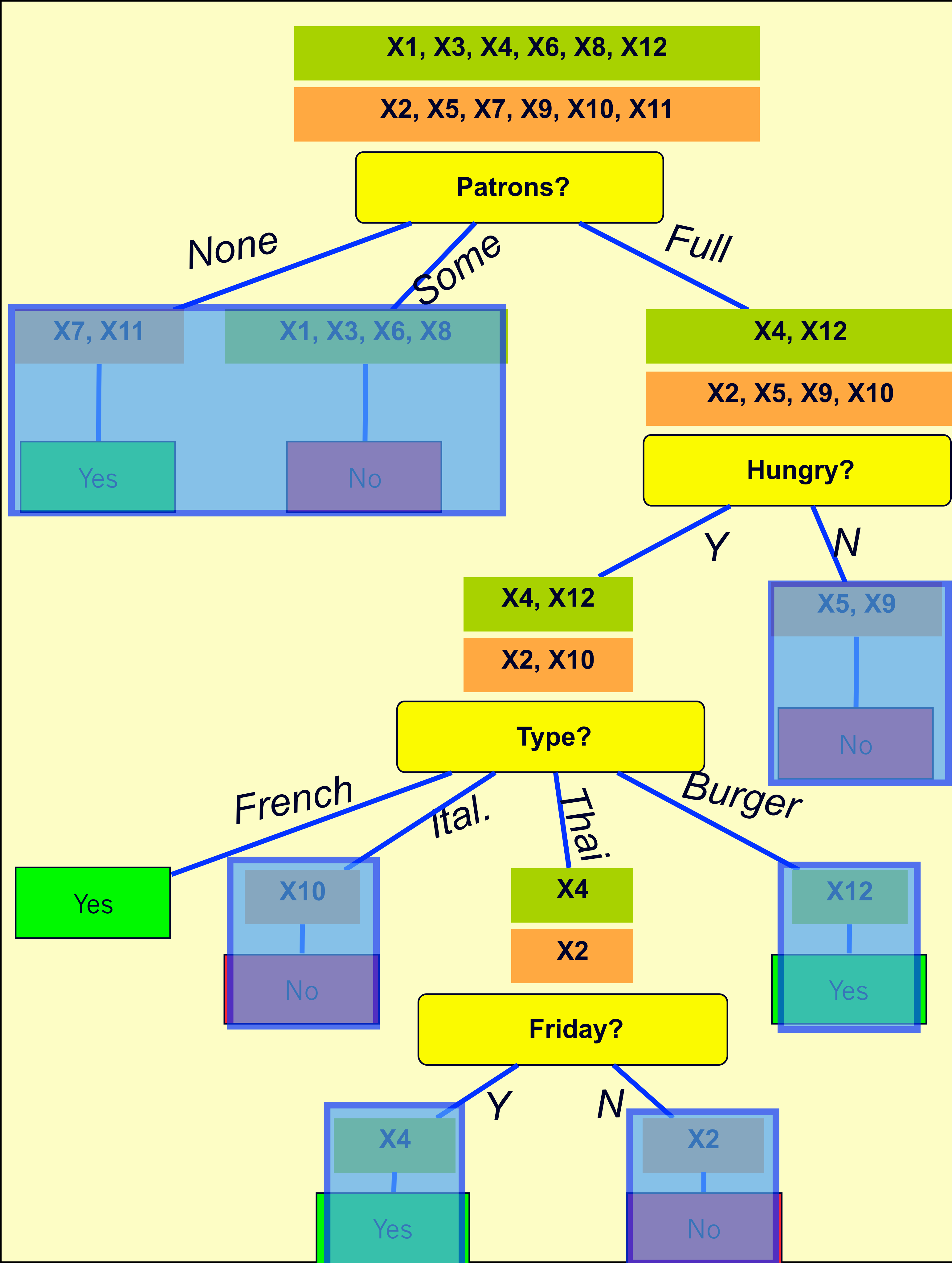
- *Hungry* needs further discrimination only for the *Yes* value
- *No* agrees with the *WillWait* goal predicate
- the next step will be performed on the remaining samples for the *Yes* value of *Hungry*

Restaurant Sample Set

Example					Attributes						Goal	Exa
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*	
X1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes	X1
X2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No	X2
X3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes	X3
X4	Yes	No	Yes	Yes	Full	$	No	No	Thai	10-30	Yes	X4
X5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No	X5
X6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes	X6
X7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No	X7
X8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes	X8
X9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No	X9
X10	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No	X10
X11	No	No	No	No	None	$	No	No	Thai	0-10	No	X11
X12	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes	X12

- **select next best attribute**
  - candidate 1: *Friday* *Yes* and *No* in agreement with goal

Tree



- the two remaining samples can be made consistent by selecting *Friday* as the next predicate
- no more samples left

Performance of Decision Tree Learning

- **quality of predictions**
  - predictions for the classification of unknown examples that agree with the correct result are obviously better
  - can be measured easily after the fact
  - it can be assessed in advance by splitting the available examples into a training set and a test set
    - learn the training set, and assess the performance via the test set
- **size of the tree**
  - a smaller tree (especially depth-wise) is a more concise representation

Noise and Over-fitting

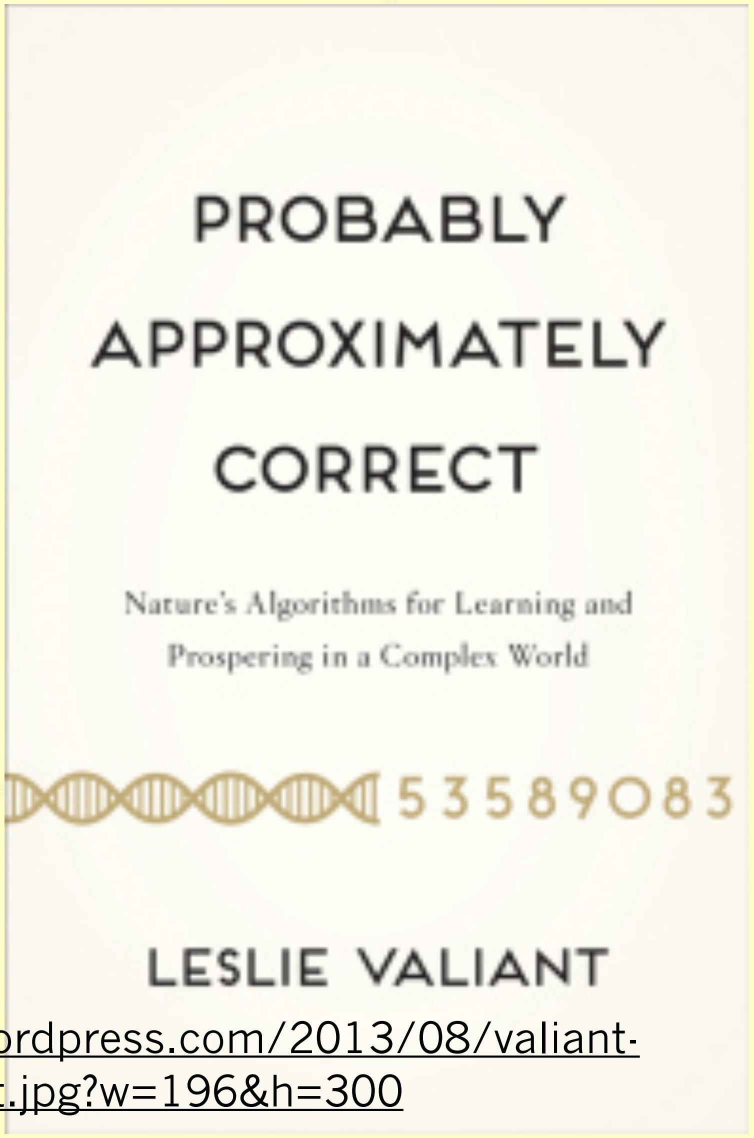
- **the presence of irrelevant attributes ("noise") may lead to more degrees of freedom in the decision tree**
  - the hypothesis space is unnecessarily large
- **overfitting makes use of irrelevant attributes to distinguish between samples that have no meaningful differences**
  - e.g. using the day of the week when rolling dice
  - over-fitting is a general problem for all learning algorithms
- **decision tree pruning identifies attributes that are likely to be irrelevant**
  - very low information gain
- **cross-validation splits the sample data in different training and test sets**
  - results are averaged

Ensemble Learning

- **multiple hypotheses (an ensemble) are generated, and their predictions combined**
  - by using multiple hypotheses, the likelihood for misclassification is hopefully lower
  - also enlarges the hypothesis space
- **boosting is a frequently used ensemble method**
  - each example in the training set has a weight associated
  - the weights of incorrectly classified examples are increased, and a new hypothesis is generated from this new weighted training set
  - the final hypothesis is a weighted-majority combination of all the generated hypotheses

Computational Learning Theory (COLT)

Background
PAC Learning



http://onionesquereality.files.wordpress.com/2013/08/valiant-probably_approximately_correct.jpg?w=196&h=300

Computational Learning Theory

- **relies on methods and techniques from theoretical computer science, statistics, and AI**
- **used for the formal analysis of learning algorithms**
- **basic principles**
  - a hypothesis is seriously wrong
    - it will most likely generate a false prediction even for small numbers of examples
  - hypothesis is consistent with a large number of examples
    - most likely it is quite good, or *probably approximately correct*

Probably Approximately Correct (PAC) Learning

- **approximately correct hypothesis**
  - its error lies within a small constant of the true result
  - by testing a sufficient number of examples, one can see if a hypothesis has a high probability of being approximately correct
- **stationary assumption**
  - training and test sets follow the same probability distribution
  - there is a connection between the past (known) and the future (unknown)
  - a selection of non-representative examples will not result in good learning

Learning in Neural Networks

Neurons and the Brain
Neural Networks
Perceptrons
Multi-layer Networks
Applications

Neural Networks

- **complex networks of simple computing elements**
- **capable of learning from examples**
  - with appropriate learning methods
- **collection of simple elements performs high-level operations**
  - thought
  - reasoning
  - consciousness

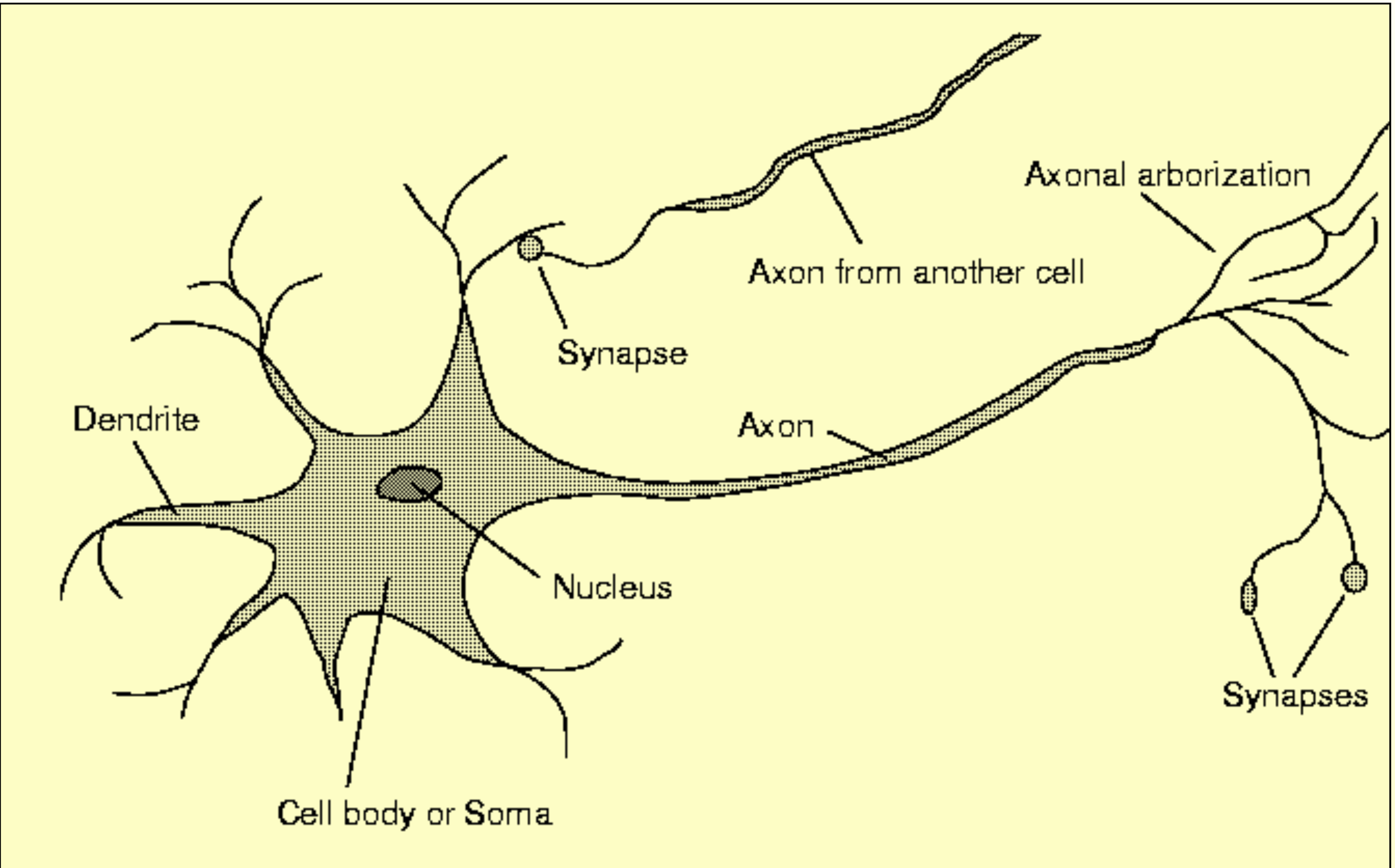
Neural Networks and the Brain

[Russell & Norvig, 1995]

- **brain**
  - set of interconnected modules
  - performs information processing operations at various levels
    - sensory input analysis
    - memory storage and retrieval
    - reasoning
    - feelings
    - consciousness

neurons
  - basic computational elements
  - heavily interconnected with other neurons

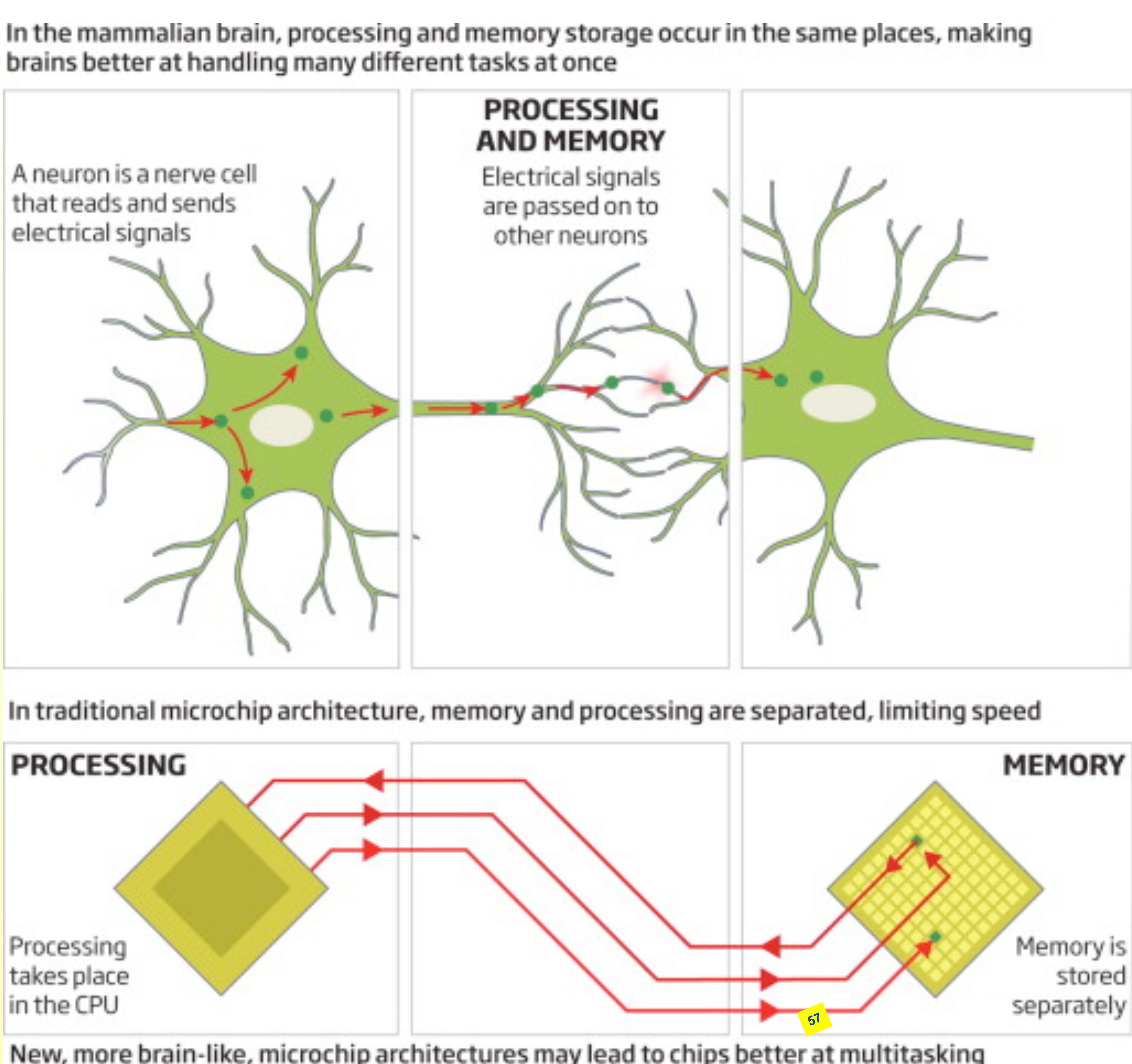
Neuron Diagram



[Russell & Norvig, 1995]

- **soma**
  - cell body
- **dendrites**
  - incoming branches
- **axon**
  - outgoing branch
- **synapse**
  - junction between a dendrite and an axon from another neuron

Brain vs. Computer



Peter Norvig, Artificial intelligence: A new future, New Scientist, Volume 216, Issue 2889, 3 November 2012, Pages vi-vii, ISSN 0262-4079, 10.1016/S0262-4079(12)62785-7.

(http://www.sciencedirect.com/science/article/pii/S0262407912627857)

http://ars.els-cdn.com/content/image/1-s2.0-S0262407912627857-gr1.jpg

Computer vs. Brain: Performance

	Computer	**Brain**
Computational units	1-1000 CPUs 10^7 gates/CPU	10^{11} neurons
Storage units	10^{10} bits RAM 10^{11} bits disk	10^{11} neurons 10^{14} synapses
Cycle time	10^{-9} sec (1GHz)	10^{-3} sec (1kHz)
Bandwidth	10^9 sec	10^{14} sec
Neuron updates/sec	10^5	10^{14}

Computer Brain vs. Cat Brain

- **in 2009 IBM makes a supercomputer “significantly smarter than a cat”**
  - “IBM has announced a software simulation of a mammalian cerebral cortex that's significantly more complex than the cortex of a cat. And, just like the actual brain that it simulates, they still have to figure out how it works.”

http://arstechnica.com/science/news/2009/11/ibm-makes-supercomputer-significantly-smarter-than-cat.ars?utm_source=rss&utm_medium=rss&utm_campaign=rss

http://static.arstechnica.com/cat_computer_ars.jpg

Google Neural Network learns about ???

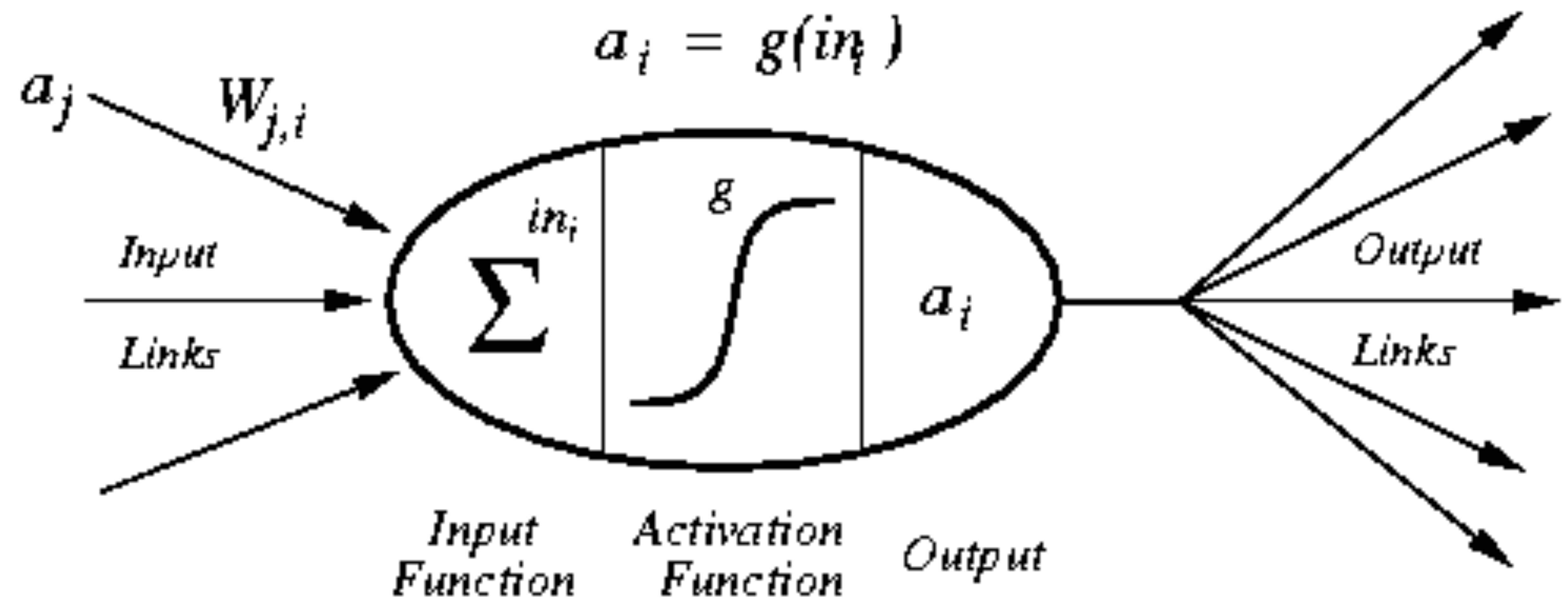
- **What does a really large NN learn from watching Youtube videos for one week?**
- **NN implementation**
  - computation spread across 16,000 CPU cores
  - more than 1 billion connections in the NN
- **http://googleblog.blogspot.com/2012/06/using-large-scale-brain-simulations-for.html**

Cat Discovery

- **"cat" discovery in NN**
  - learned to identify a category of images with cats
  - Google blog post
    - https://plus.google.com/u/0/+ResearchatGoogle/posts/EMyhnBetd2F
  - published paper
    - http://static.googleusercontent.com/external_content/untrusted_dlcp/research.google.com/en/us/archive/unsupervised_icml2012.pdf

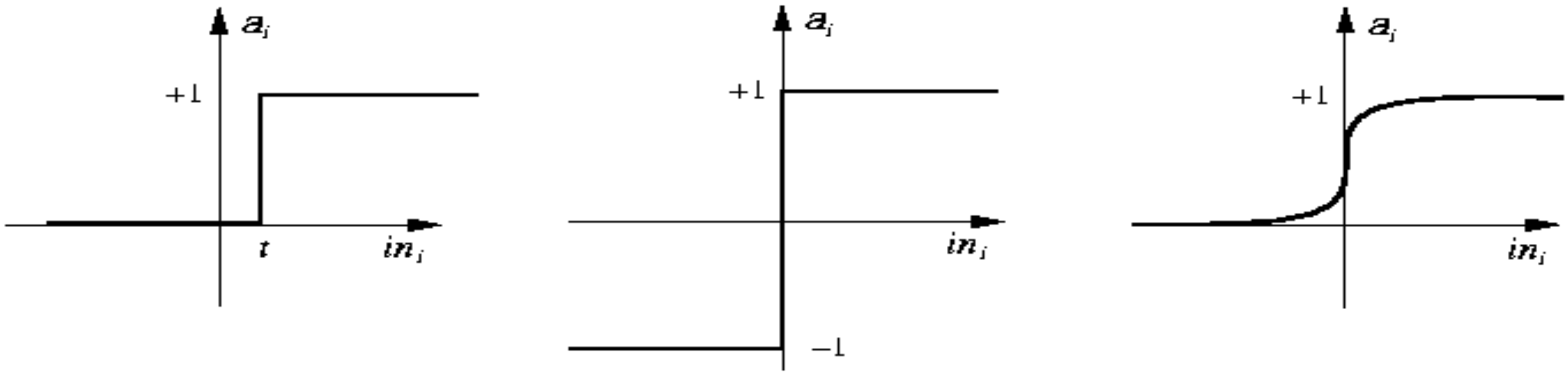
http://1.bp.blogspot.com/-VENOsYD1uJc/T-nkLAiANtI/AAAAAAAAJWc/2KCTl3OsI18/s1600/cat+detection.jpeg

Artificial Neuron Diagram



[Russell & Norvig, 1995]

Common Activation Functions



(a) Step function (b) Sign function (c) Sigmoid function

[Russell & Norvig, 1995]

$Step_t(x)$ = 1 if x >= t, else 0

Sign(x) = +1 if x >= 0, else –1

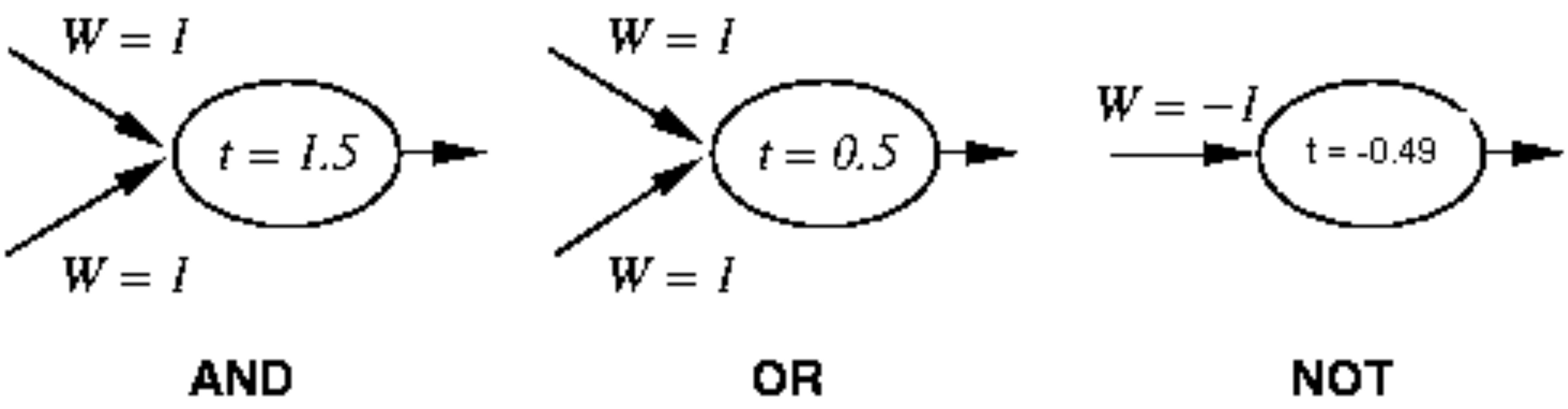
Sigmoid(x) = $1/(1+e^{-x})$

CAL POLY

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

Neural Networks and Logic Gates

- **simple neurons with can act as logic gates**
  - appropriate choice of activation function, threshold, and weights
    - step function as activation function

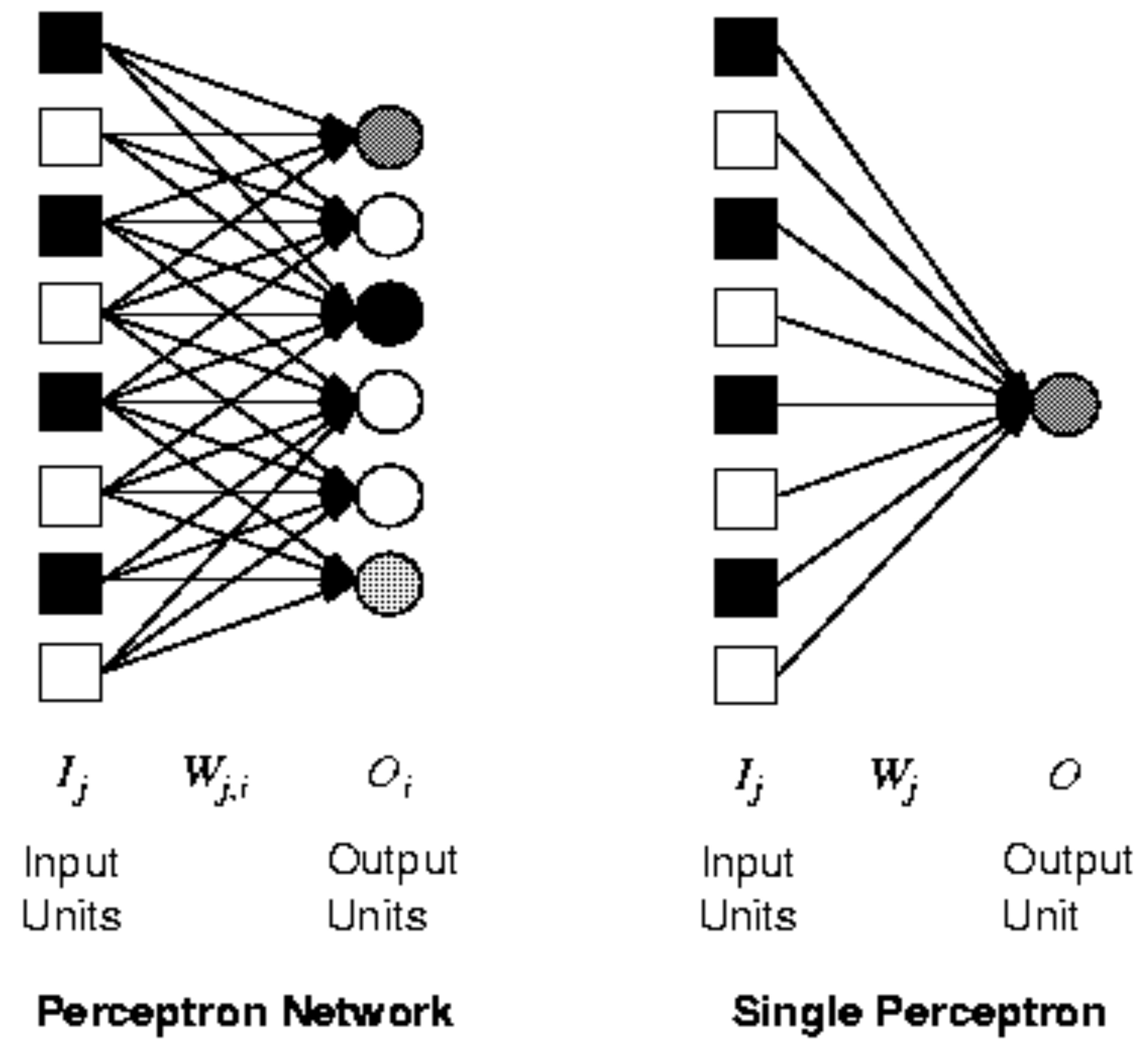


[Russell & Norvig, 1995]

Network Structures

- **in principle, networks can be arbitrarily connected**
  - occasionally done to represent specific structures
    - semantic networks
    - logical sentences
  - makes learning rather difficult
- **layered structures**
  - networks are arranged into layers
  - interconnections mostly between two layers
  - some networks have feedback connections

Perceptrons

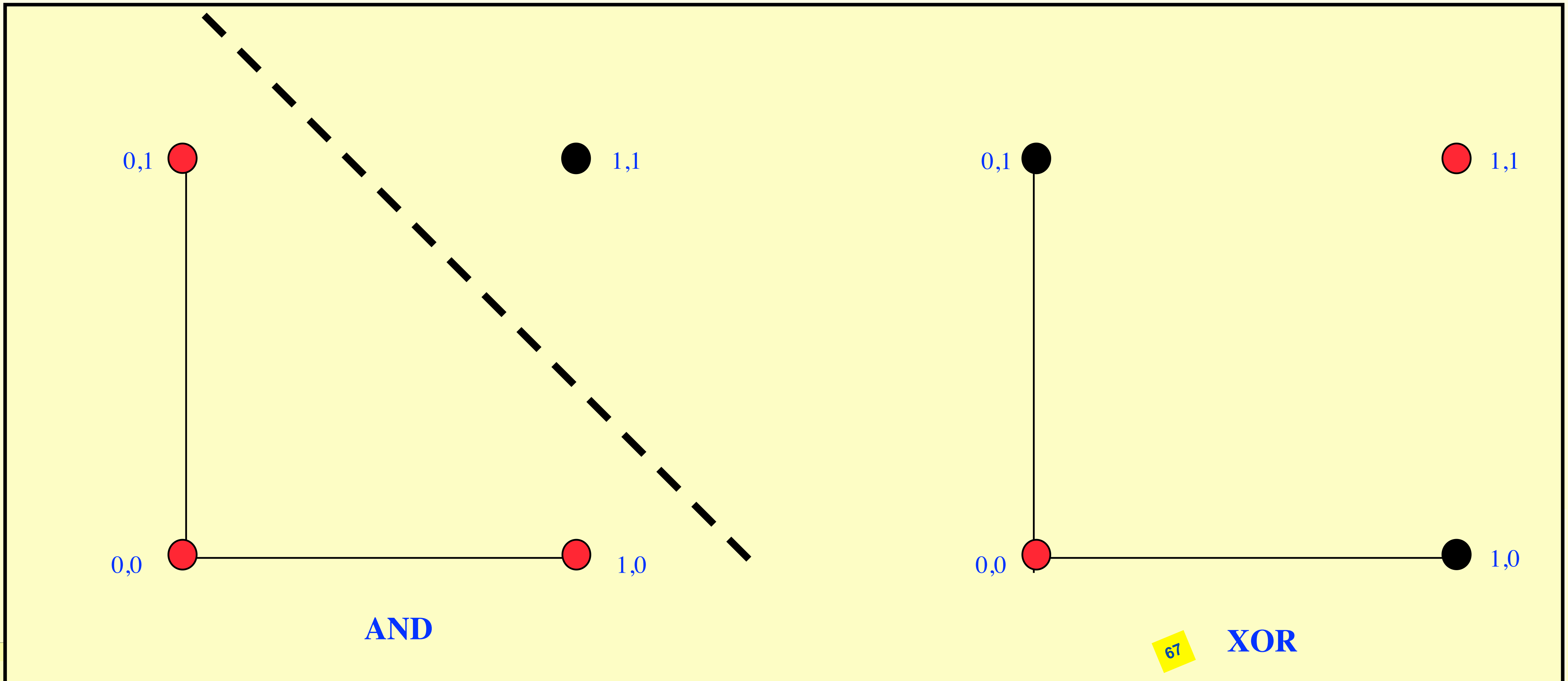


[Russell & Norvig, 1995]

- single layer, feed-forward network
- historically one of the first types of neural networks
  - late 1950s
- the output is calculated as a step function applied to the weighted sum of inputs
- capable of learning simple functions
  - linearly separable

Perceptrons and Linear Separability

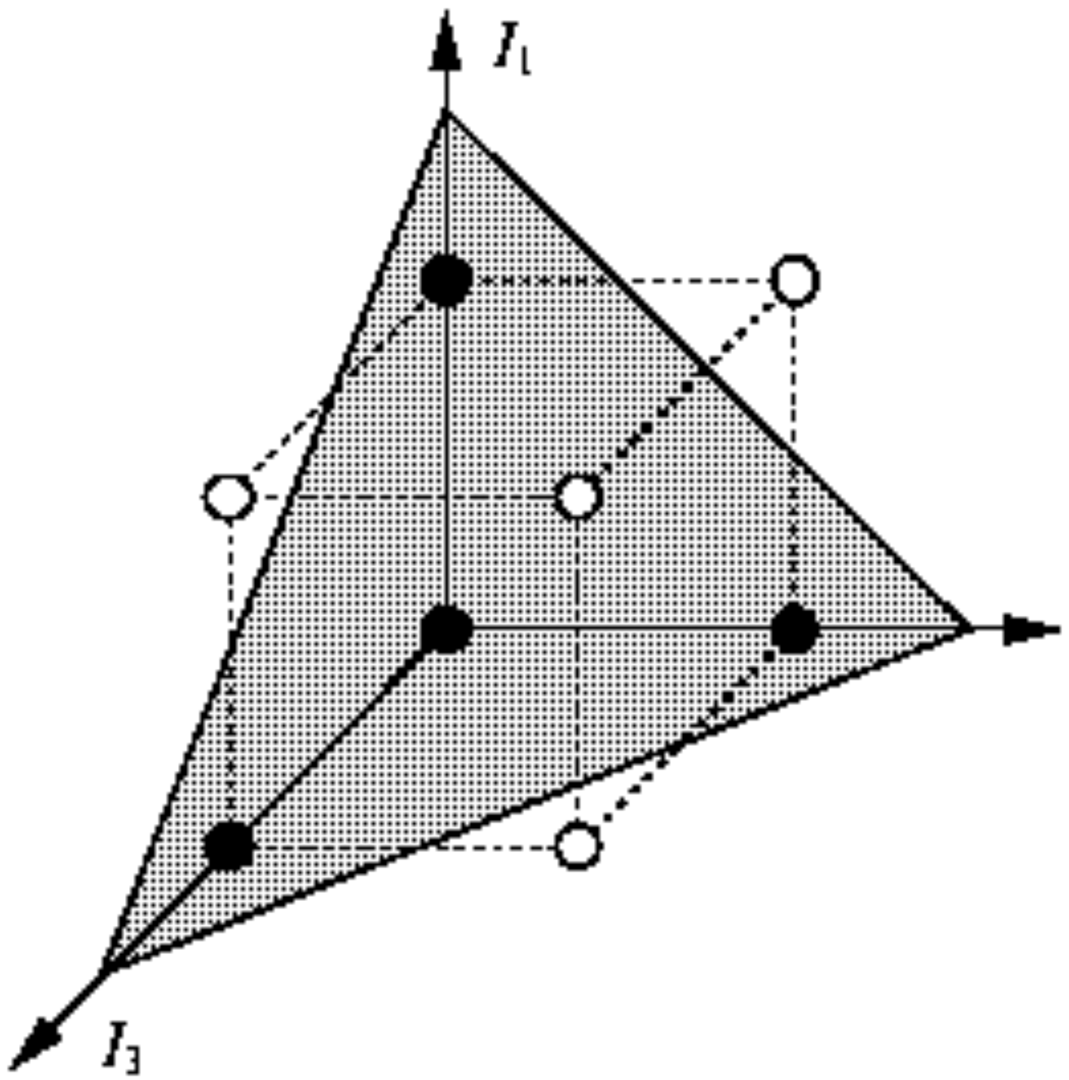
- **perceptrons can deal with linearly separable functions**
- **some simple functions are not linearly separable**
  - XOR function



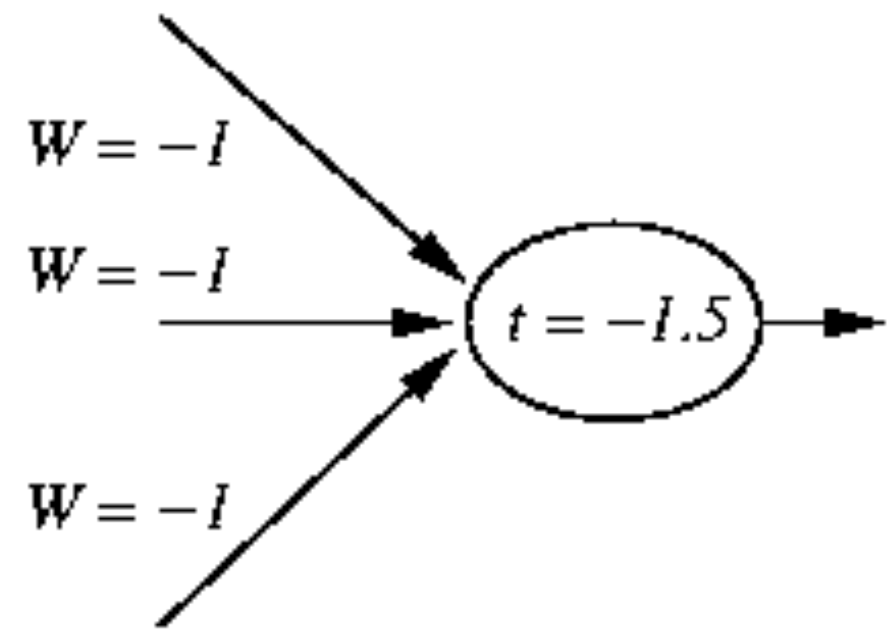
[Russell & Norvig, 1995]

Perceptrons and Linear Separability

- linear separability can be extended to more than two dimensions
- more difficult to visualize



(a) Separating plane



(b) Weights and threshold

Perceptrons and Learning

- **perceptrons can learn from examples through a simple learning rule**
  - calculate the error of a unit Err_i as the difference between the correct output T_i and the calculated output O_i

    $Err_i = T_i - O_i$
  - adjust the weight W_j of the input I_j such that the error decreases

    $W_{ij} := W_{ij} + \alpha * I_{ij} * Err_{ij}$
    - α is the learning rate
  - this is a gradient descent search through the weight space
  - lead to great enthusiasm in the late 50s and early 60s until Minsky & Papert in 69 analyzed the class of representable functions and found the linear separability problem

Generic Neural Network Learning

- **basic framework for learning in neural networks**

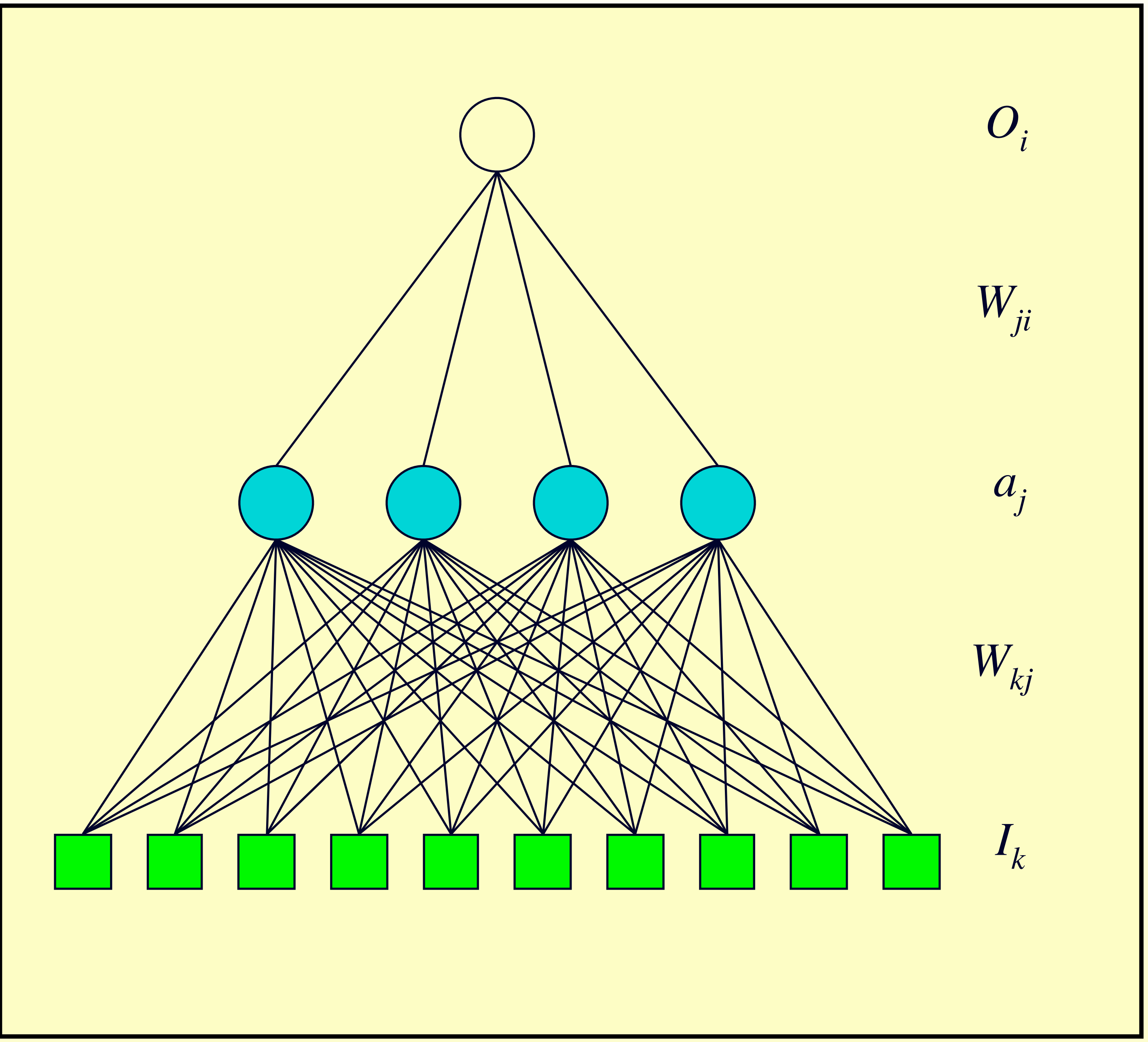
function NEURAL-NETWORK-LEARNING(*examples*) **returns** *network*
network := a network with randomly assigned weights
for each *e* **in** *examples* **do**
O := NEURAL-NETWORK-OUTPUT(*network,e*)
T := observed output values from *e*
update the weights in *network* based on *e, O,* and *T*
return *network*

adjust the weights until the predicted output values *O* and the observed values *T* agree

Multi-Layer Networks

- **research in the more complex networks with more than one layer was very limited until the 1980s**
  - learning in such networks is much more complicated
  - the problem is to assign the blame for an error to the respective units and their weights in a constructive way
- **the back-propagation learning algorithm can be used to facilitate learning in multi-layer networks**

Diagram Multi-Layer Network



- **two-layer network**
  - input units I_k
    - usually not counted as a separate layer
  - hidden units a_j
  - output units O_i
- **usually all nodes of one layer have weighted connections to all nodes of the next layer**

Back-Propagation Algorithm

- **assigns blame to individual units in the respective layers**
  - essentially based on the connection strength
  - proceeds from the output layer to the hidden layer(s)
  - updates the weights of the units leading to the layer
- **essentially performs gradient-descent search on the error surface**
  - relatively simple since it relies only on local information from directly connected units
  - has convergence and efficiency problems

Capabilities of Multi-Layer Neural Networks

- **expressiveness**
  - weaker than predicate logic
  - good for continuous inputs and outputs
- **computational efficiency**
  - training time can be exponential in the number of inputs
  - depends critically on parameters like the learning rate
  - local minima are problematic
    - can be overcome by simulated annealing, at additional cost
- **generalization**
  - works reasonably well for some functions (classes of problems)
    - no formal characterization of these functions

Capabilities of Multi-Layer Neural Networks (cont.)

- **sensitivity to noise**
  - very tolerant
  - they perform nonlinear regression
- **transparency**
  - neural networks are essentially black boxes
  - there is no explanation or trace for a particular answer
  - tools for the analysis of networks are very limited
  - some limited methods to extract rules from networks
- **prior knowledge**
  - very difficult to integrate since the internal representation of the networks is not easily accessible

Deep Learning

- **are part of the broader machine learning field of learning representations of data**
- **cascade of many layers of nonlinear processing units for feature extraction and transformation**
  - each successive layer uses the output from the previous layer as input
  - algorithms may be supervised or unsupervised
  - applications include pattern recognition and statistical classification
- **based on the (unsupervised) learning of multiple levels of features or representations of the data**
  - higher level features are derived from lower level features to form a hierarchical representation
- **learn multiple levels of representations**
  - correspond to different levels of abstraction
  - the levels form a hierarchy of concepts
- **can be computationally very expensive**
  - very large data sets
  - complex algorithms

https://en.wikipedia.org/wiki/Deep_learning
J. Schmidhuber, "Deep Learning in Neural Networks: An Overview" http://arxiv.org/abs/1404.7828, 2014

Why Deep Learning?

- **Machine learning**
  - just optimizing weights to make a final prediction
  - human-designed representations and input features
  - internal representation nevertheless not very transparent
- **Representation learning**
  - attempts to automatically learn good features or representations
- **Deep learning algorithms**
  - attempt to learn multiple levels of representation of increasing complexity/ abstraction

Deep Learning Architectures

- **deep belief networks**
- **Markov Random Fields with multiple layers**
- **multi-layer neural networks for supervised learning**
  - input layer
    - sensory inputs
  - hidden layers
    - more abstract representations for higher levels
  - output layer
    - prediction of a supervised target

Deep Learning Advantages

- **learning representations**
  - instead of hand-crafted features that are often domain-specific, incomplete, over-specified, time-consuming to develop, subjective
- **distributed vs. symbolic representations**
  - traditional systems represent one conceptual entity (object, word, number, etc.) as one “chunk” (instance of a data structure, record)
    - similar conceptual entity may have completely different representations
  - distributed representations “spread out” the storage of entities (“sparse” representations)
    - multiple storage “chunks” contain parts of multiple conceptual entities
  - similarity becomes an inherent property of the representation
  - multiple dimensions of similarity are possible
    - for words, similarity for both spelling and meaning can be incorporated
  - distributed representations facilitate multi-clustering
    - categorize entities according do multiple features often works better than local clustering (e.g. nearest-neighbor)
  - distributed representations reduce the “curse of dimensionality”
    - generalizing locally requires representative examples for all relevant variations, covering all dimensions in the feature space

Deep Learning Advantages (cont.)

- **unsupervised learning**
  - both for features and for weights
  - in many domains, many to most data sets are unlabeled
    - e.g., Natural Language Processing: text documents don't have annotations about sentence structure and meaning
  - good data models can help with learning
    - data models can still be a challenge to obtain or create
- **multiple levels of representation**
  - good intermediate representations
    - can be shared across tasks
    - capture some aspects of the domain
  - related to the "depth" of a domain or data model
    - higher model layers learn more abstract intermediate representations
  - may increase comprehensibility for humans
  - can help with compositionality

Deep Learning Success

- **related ideas have been explored for some time**
  - connectionism
  - parallel distributed processing
  - sparse encoding
  - associative memory
  - ...
- **practical learning methods have become available since around 2006**
  - unsupervised pre-training
  - better parameter estimation methods
  - understanding of model regularization
- **computing power**
  - Google's Deep Learning "Cat" Success in 2012
    - see e.g. John Markoff's NY Times article "How Many Computers to Identify a Cat? 16,000"

Deep Learning and Image Processing

- **one of the first areas to use deep learning methods**
- **substantial background knowledge**
  - features at different levels of abstraction
  - conventional computational methods
    - many carefully adapted to specific circumstances
- **large data sets available**

Deep Learning & NLP

- **Natural Language Processing is one of the areas where deep learning has been very successful**
  - Google's NLP efforts (probably)
    - Google Now
    - Youtube transcription service
    - Google phone service
  - Microsoft MAVIS speech recognition engine
  - Apple's Siri (probably)
- **background material**
  - a good overview of NLP and Deep Learning is
    - Deep Learning for NLP, a tutorial given at NAACL HLT 2013 by Richard Socher and Christopher Manning
  - Stanford course Spring 2015 by Richard Socher
    - "CS224d: Deep Learning for Natural Language Processing"

Applications

- **domains and tasks where neural networks are successfully used**
  - handwriting recognition
  - control problems
    - juggling, truck backup problem
  - series prediction
    - weather, financial forecasting
  - categorization
    - sorting of items (fruit, characters, phonemes, ...)

Important Concepts and Terms

- axon
- back-propagation learning algorithm
- bias
- decision tree
- dendrite
- feedback
- function approximation
- generalization
- gradient descent
- hypothesis
- inductive learning
- learning element
- linear separability
- machine learning
- multi-layer neural network
- neural network
- neuron
- noise
- Ockham's razor
- perceptron
- performance element
- prior knowledge
- sample
- synapse
- test set
- training set
- transparency

Chapter Summary

- **learning is very important for agents to improve their decision-making process**
  - unknown environments, changes, time constraints
- **most methods rely on inductive learning**
  - a function is approximated from sample input-output pairs
- **decision trees are useful for learning deterministic Boolean functions**
- **neural networks consist of simple interconnected computational elements**
- **multi-layer feed-forward networks can learn any function**
  - provided they have enough units and time to learn