CS 480 ARTIFICIAL INTELLIGENCE

Bala Ravikumar
Computer Science Department
116 I Darwin Hall

Class meets:
T Th 5:30 to 6:45 PM, Salazar 1051

Office hours:
T 8:30 to 9:30 AM Th 3 to 4:30 PM
F 9 AM to 10 AM (zoom)

Course details

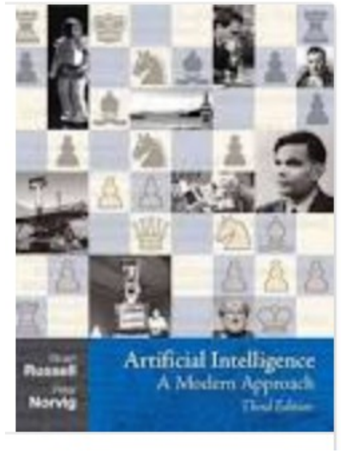
Catalog Description:

CS/ES 480 A survey of techniques that simulate human intelligence. Topics may include: pattern recognition, general problem solving, adversarial game-tree search, decision-making, expert systems, neural networks, fuzzy logic, and genetic algorithms. Prerequisite: CS 315 or consent of instructor.

Background Expected:

- Programming and data structures (CS 315)
- Linear algebra
- Probability (basics)
- Some background in logic (covered in CS 242) will be helpful.

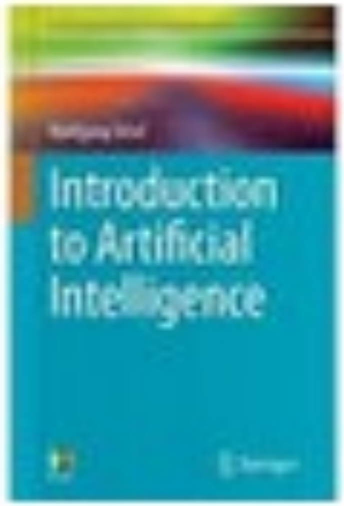
Text book(s)

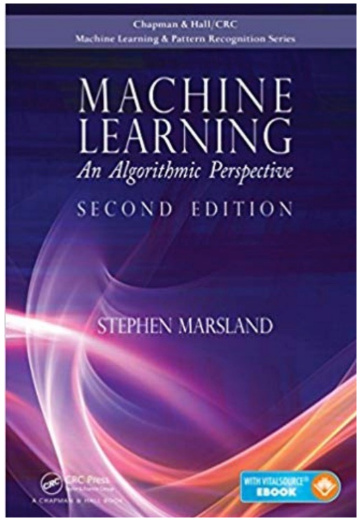


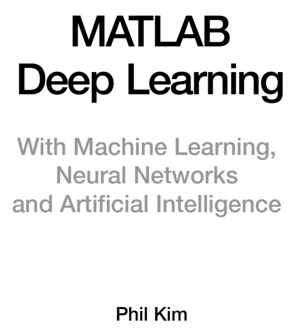
Main text:

Artificial intelligence – a modern approach
Third edition

Additional references:




Chapman & Hall/CRC
Machine Learning & Pattern Recognition Series
MACHINE LEARNING
An Algorithmic Perspective
SECOND EDITION
STEPHEN MARSLAND



MATLAB
Deep Learning

With Machine Learning,
Neural Networks
and Artificial Intelligence

Phil Kim

AI includes a very wide range of topics:

- problem solving, planning (searching and constraint programming)
- Reasoning and deduction
- Understanding natural language
- vision and speech processing
- symbolic programming, theorem proving
- Medical diagnosis
- Financial decision making
- game playing

Related fields:

- Robotics
- Cognitive science, neuroscience

AI techniques

- combinatorial (tree searching, A*, and/or graphs, mix-max, alpha-beta pruning etc.)
- logical (model world as predicate and evaluate)
- statistical (predict probabilities of outcomes and maximize expected gain)
- learning from data (decision tree, Bayesian network, naïve Bayes, support vector machine, boosting etc.)
- algorithms modeled by brain, genetics, insect swarms (neural network, genetic algorithms, swarm intelligence etc.)
- optimization (quantify the goal and solve max or min finding problem)

Topics covered

Search Algorithms (5 weeks)

- State space representation
- Uninformed search techniques:
  - BFS, DFS, iterative-deepening etc.
- Informed search techniques
  - Greedy search
  - A* search
  - Graph search vs. tree search
  - Heuristics – monotonicity,
- Adversarial search
  - Two-player games and game trees
  - Min-max search
  - Alpha-beta pruning
- Markov Decision Processes
  - Maximizing expected gain
  - Application to reinforcement learning
  - Hidden Markov model*

- Neural Networks (3 weeks)
  - Perceptron model
    - Perceptron training algorithm
    - Perceptron convergence and linear separability

  - Neural networks
    - Back-propagation algorithm
    - Application to hand-writing recognition

  - Deep-learning
    - Convolutional neural networks
    - Advanced topics

- Machine learning algorithms (4 weeks)
  - Supervised learning
    - Naïve-bayes algorithm
    - Decision-tree algorithm
    - Support-vector machines

  - Markov Decision Processes
    - Application to reinforcement learning

  - Unsupervised learning
    - K-means clustering
    - Spectral clustering*

- Applications (3 weeks)
  - Computer vision
  - Hand-writing recognition
  - Speech recognition
  - Document classification, spam filtering
  - Natural language modeling, language translation etc.

Course work and grading (tentative and approximate)

- *Short quizzes* (10 to 15%) There will be about 10 quizzes – almost in every class (except the first class and when there is a mid-term). The quizzes will be done in class, taking approximately 10 to 15 minutes. You can use the notes and openly discuss the solution with other students. Every fourth quiz will be longer and weighted 3 points and will be done individually.

- *Two Mid-Term tests* (20 to 25%) – Both tests will be in class and will be about 90 minutes long. The tests will be open book/open notes.

- *Programming projects* (30 + 15%) – There will be three or four projects that will be done in pairs. Each project involves application of an AI technique to solve a problem in areas such as document classification, game playing, searching for optimal path, hand-writing recognition, speech analysis etc. A final project involves investigation and implementation of a solution to one problem from a list of assigned problems. You are to write a paper based on the work, and also present it during the last week of the semester. The final project will be done in teams of size three.

- *Final Examination* (20%) – The final examination will be comprehensive although greater emphasis will be on the latter topics. It will be in-class and will follow the scheduled time provided in the academic calendar.

Important Dates:

- First day of class: Tuesday, August 23
- Last day of class: Thursday, Dec 8
- Mid-term 1: Thursday, Sept 22
- Mid-term 2: Thursday, Nov 3
- Final Examination: Thursday, Dec 15, 6 PM to 8 PM
- Final Project due: Monday, Dec 12

What is AI?

Some major current applications of AI:

- business: advertising, financial decision making (mortgage qualification, bank loans etc.), economic modeling, forecasting, user profiling through social networks (recall presidential election of 2012)
- web: identifying objects in images, automatic image classification, classifying web content, market analysis, data analytics etc.
- utilities: spam filter, search engine, airline reservations etc.
- medical: image classification (benign vs. malignant tumor), micro-robotics, drug design, microarray analysis, RNA and protein folding models, systems biology, brain-computer interface, …
- industrial: vision, robotics
- others: language translation, semantic analysis, speech synthesis, speech to text conversion

Some concrete examples

- Alexa, siri
- IBM’s Watson
- Self-driving car
- Search engines
- Google maps
- Automatic grading tools (used in SAT for grading essays)
- Spam filter, plagiarism detection
- Flight crew assignment
- Google translator
- Netflix movie recommendation system
- Review of consumer products

and thousands more.

Acting humanly: cognitive modeling

- Requires scientific theories of internal activities of the brain.
- How to validate? Requires
  - Predicting and testing behavior of human subjects (top-down) or
  - Direct identification from neurological data (bottom-up)
- Both approaches (roughly, Psychology vs. Neuroscience)
- now distinct from AI

Acting rationally: rational agent

- Rational behavior: doing the right thing.

- The right thing: that which is expected to maximize goal achievement, given the available information.

  (goal of machine learning: maximize the expected utility based on data.)

- Some times terms "strong AI" (directly implement brain model) and "weak AI" (simulate the i/o but with different mechanism) are used to contrast the two approaches.

Abridged history of AI

- 1943 McCulloch & Pitts: Boolean circuit model of brain
- 1950 Turing's "Computing Machinery and Intelligence“
- 1956 Dartmouth meeting: "Artificial Intelligence" adopted
- 1950s Early AI programs, including Samuel's checkers program, Newell & Simon's Logic Theorist
- 1965 Robinson's resolution algorithm for logical reasoning

1966 - 73 AI discovers computational complexity
Neural network research almost disappears (after the book by *Minsky and Papert*)

1969 - 79 Early development of knowledge-based systems

1986 - Neural networks return to popularity (Rumelhart, Hinton and Sejnowski)

1987 - AI becomes a science, probabilistic techniques dominate

1995 - The emergence of intelligent agents

2000's: Embedded mobile applications, web services, web analytics, voice recognition, natural language understanding and user interfaces, natural language translation, mobile robots, deep learning etc.

Some landmark AI programs/systems

- Checkers program by Arthur Samuel
- Babrow, Slagle, Moses, Evans etc.
- Winograd’s block world
- Dendral (medical diagnosis)
- Huffman et al. on 3d model from line drawing
- Backgammon program by Tesauro
- Deepblue and its successors
- NETtalk by Sejnowiski
- DARPA challenge on self-driving car
- Dragon naturally speaking (evolved from CMU research on speech processing)
- Expert systems
- AlphaGO

AI techniques

Combinatorial search problems

- state space (over which search is performed)
- finite state space (discrete)
- how to move from one state to another (transition rules)

- Applications
  - Games (one player or two players)
  - Navigation (robotics)
  - Planning, constraint satisfaction problems etc.

- Solution
  - Search tree exploration

AI techniques

- Symbolic (logical) approach to AI

intelligent problem solving requires reasoning and deduction.

Knowledge is represented as a set of logical assertions $A_1, \ldots, A_n$, and a conclusion to be drawn is also expressed as an assertion.

Can we deduce F from $A_1, \ldots, A_n$?

Deductive reasoning system - Example

"All hummingbirds are richly colored."
"No large birds live on honey."
"Birds that do not live on honey are dull in color."
"Hummingbirds are small."

Let $P(x)$, $Q(x)$, $R(x)$, and $S(x)$ be the statements "x is a hummingbird," "x is large," "x lives on honey," and "x is richly colored," respectively. Assuming that the domain consists of all birds, express the statements in the argument using quantifiers and $P(x)$, $Q(x)$, $R(x)$, and $S(x)$.

Solution: We can express the statements in the argument as

$$\forall x(P(x) \to S(x)).$$
$$\neg\exists x(Q(x) \land R(x)).$$
$$\forall x(\neg R(x) \to \neg S(x)).$$
$$\forall x(P(x) \to \neg Q(x)).$$

Logic Programming

Logical approach to problem solving led to development of Prolog, a general purpose programming language

- Computation is expressed as assertion
- Prolog system will try to find a proof of the assertion
- Often, solution can be extracted from the proof.

Example: can a node x be reached from a node y?
Predicate reach defines the computation:

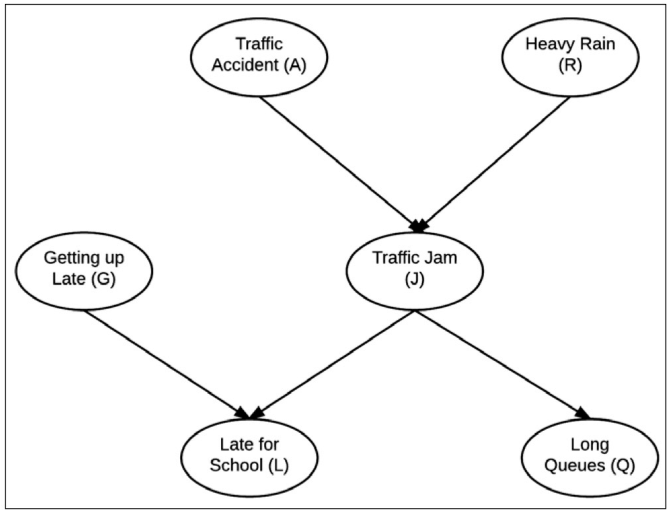
```
   reach(x, x).  // every node is reachable from itself
   reach(x, y) if edge(x,y).  // if there is an edge from x to y
   reach(x, z) if reach(x, w) and edge(x, y).
```

Statistical techniques

Modeling unknown quantities using probabilities

Use Bayesian inference

- Naïve Bayes method
- Bayesian networks
- Hidden Markov models



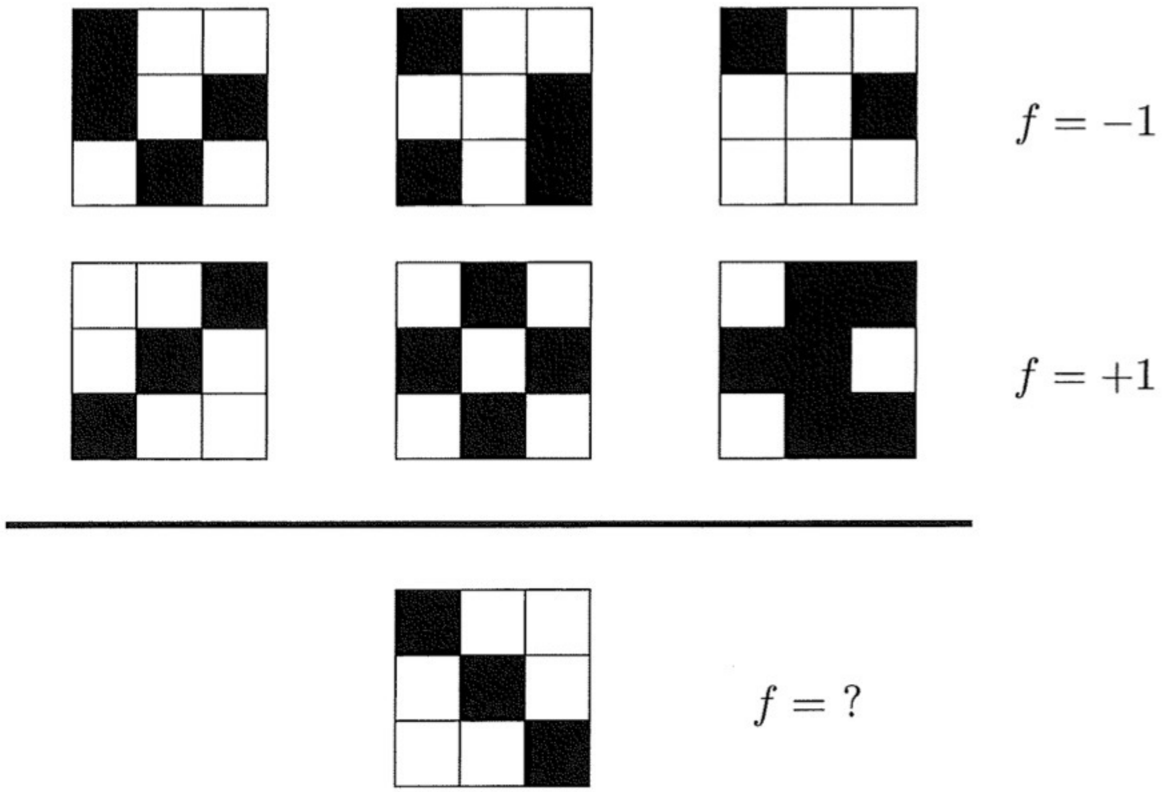


Figure 1.7: A visual learning problem. The first two rows show the training examples (each input $\mathbf{x}$ is a 9-bit vector represented visually as a 3×3 black-and-white array). The inputs in the first row have $f(\mathbf{x}) = -1$, and the inputs in the second row have $f(\mathbf{x}) = +1$. Your task is to learn from this data set what f is, then apply f to the test input at the bottom. Do you get -1 or $+1$?

AI technique – learning from data

Machine learning approach to AI:

- self-improving algorithms
- solution obtained without explicit programming
- Closer to modeling human intelligence or natural intelligence (we learn many things by observing even if step by step procedure absent)

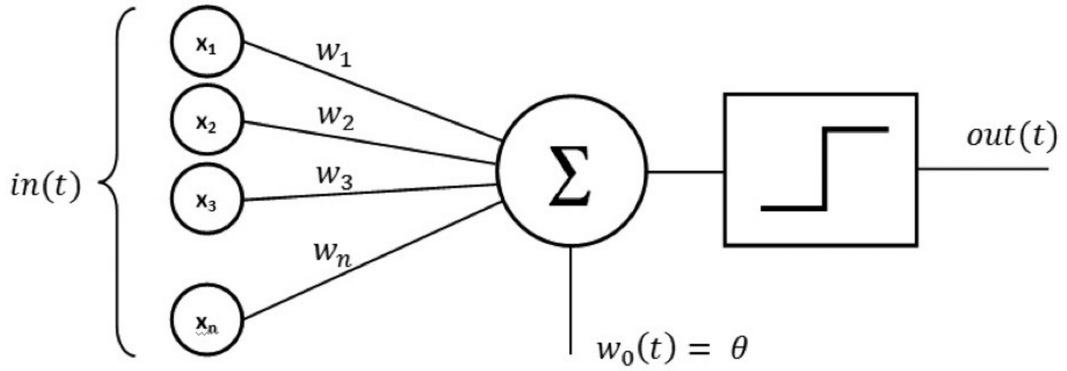
Prominent examples:

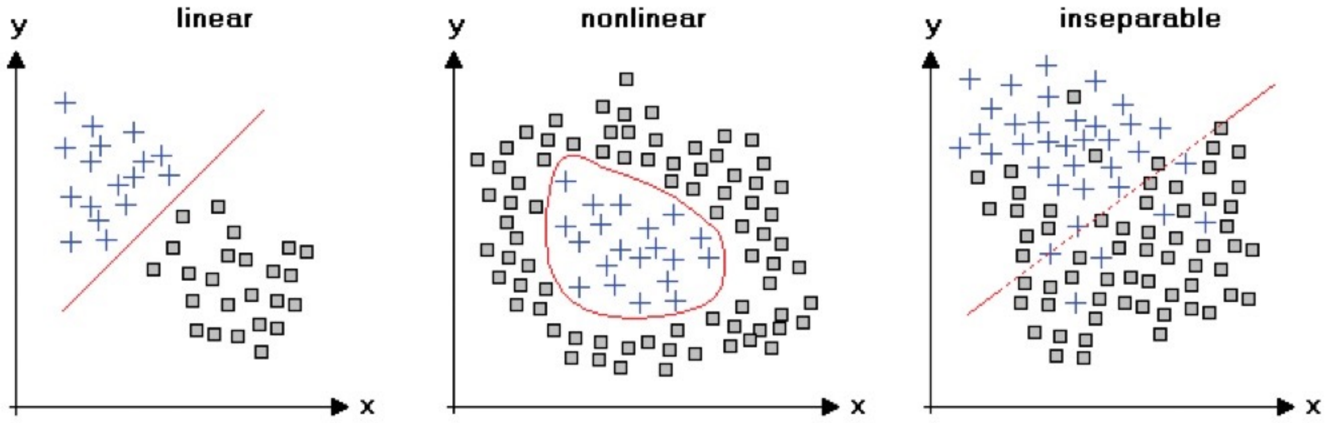
- Neural networks
- Support vector machine, decision trees, Bayesian inference, nearest neighbor etc.

AI technique – model from brain

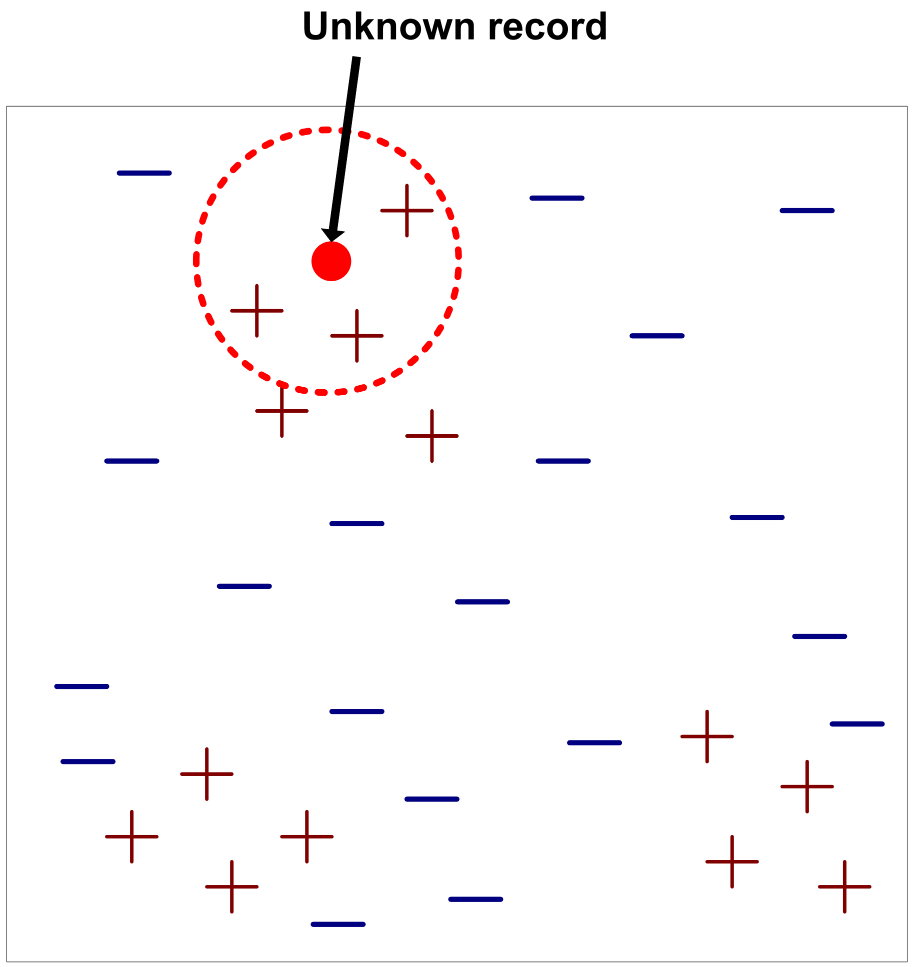
Perceptron model (very roughly modeled by neurons in human brains).

The Perceptron





Nearest-Neighbor Classifiers



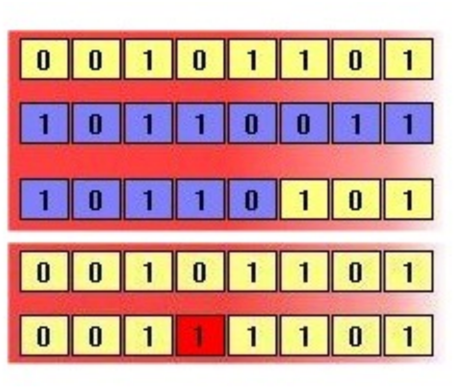
- Requires three things
  - The set of labeled records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

AI technique – model from genetics

Evolutionary algorithms:

- encoding of the collection of solutions as strings.
- goal is to evolve the "best" solution.
- use cross-over and mutation and iterate.



Example of cross-over and mutation

Machine learning

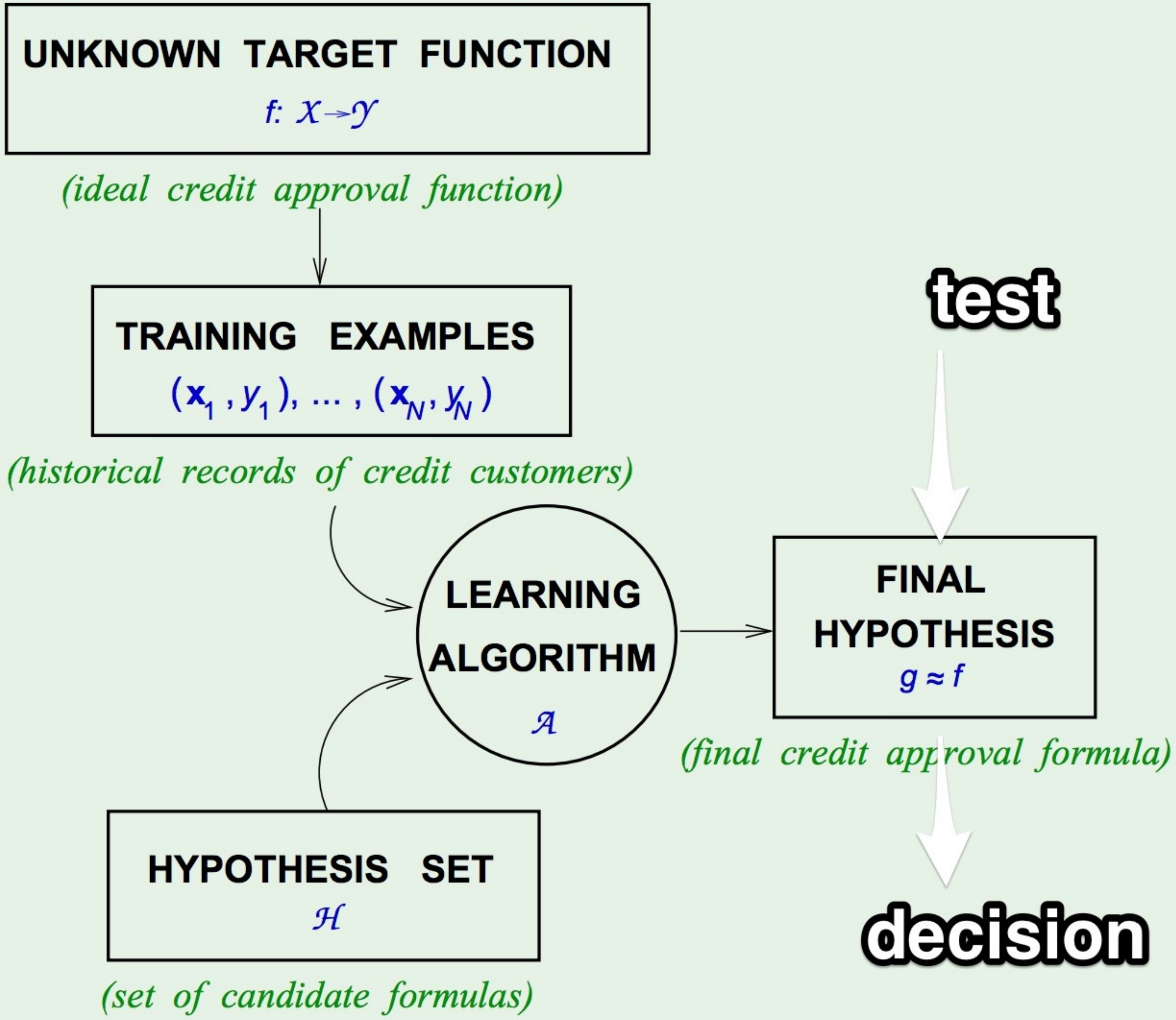
- Was considered a subfield of AI
- Now central to AI – most current applications of AI are based on machine learning
- Theoretical foundations of AI:
  - PAC learning, learnable concepts, connection to information theory and complexity
- Three major classes:
  - Supervised learning
  - Unsupervised learning
  - Reinforcement learning

Machine learning general framework

- Input: $\mathbf{x}$ *(customer application)*
- Output: y *(good/bad customer?)*
- Target function: $f : \mathcal{X} \to \mathcal{Y}$ *(ideal credit approval formula)*
- Data: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$ *(historical records)*

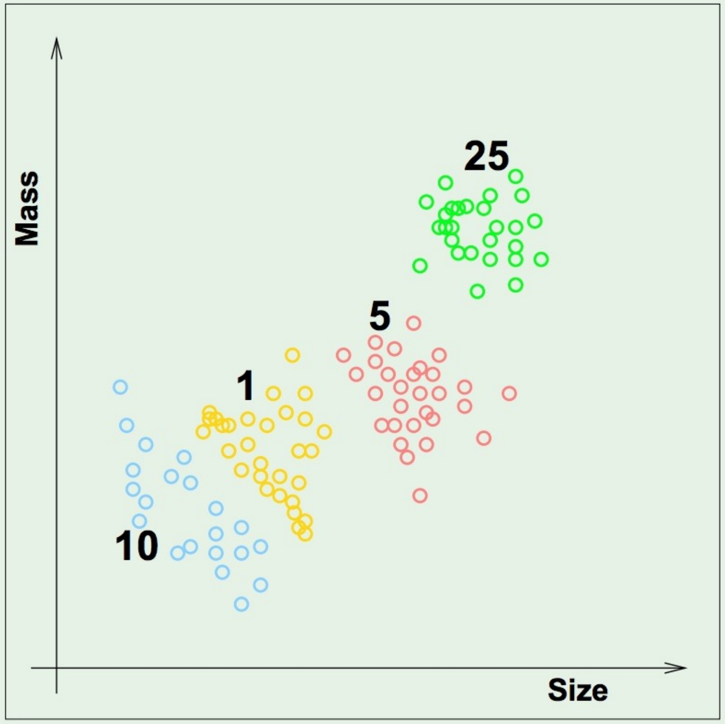
$$\downarrow \quad \downarrow \quad \downarrow$$

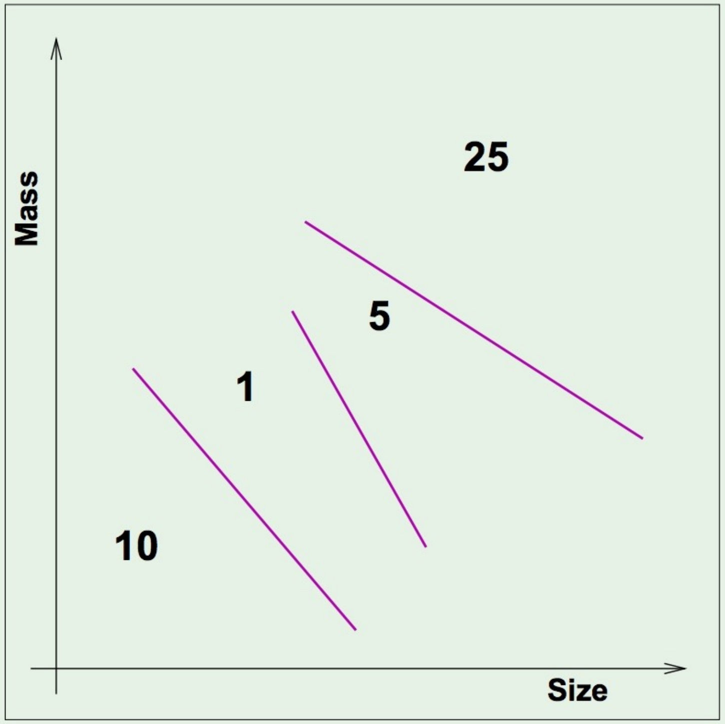
- Hypothesis: $g : \mathcal{X} \to \mathcal{Y}$ *(formula to be used)*


UNKNOWN TARGET FUNCTION
$f: \mathcal{X} \to \mathcal{Y}$
(ideal credit approval function)
TRAINING EXAMPLES
$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
(historical records of credit customers)
LEARNING ALGORITHM
$\mathcal{A}$
FINAL HYPOTHESIS
$g \approx f$
(final credit approval formula)
HYPOTHESIS SET
$\mathcal{H}$
(set of candidate formulas)
test
decision

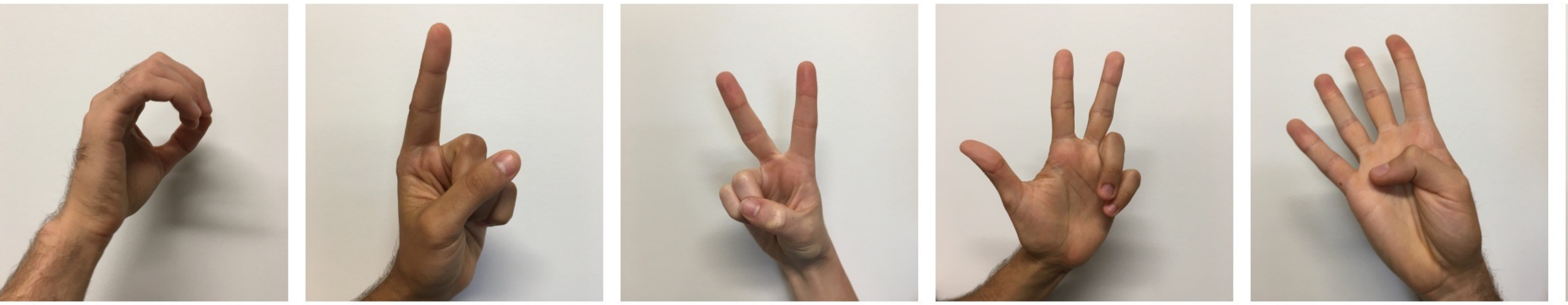
Classification with four class labels

Example from vending machines – **coin recognition**





Recognizing sign language from image



Combinatorial search

Example:

Sliding piece puzzle:

Start:

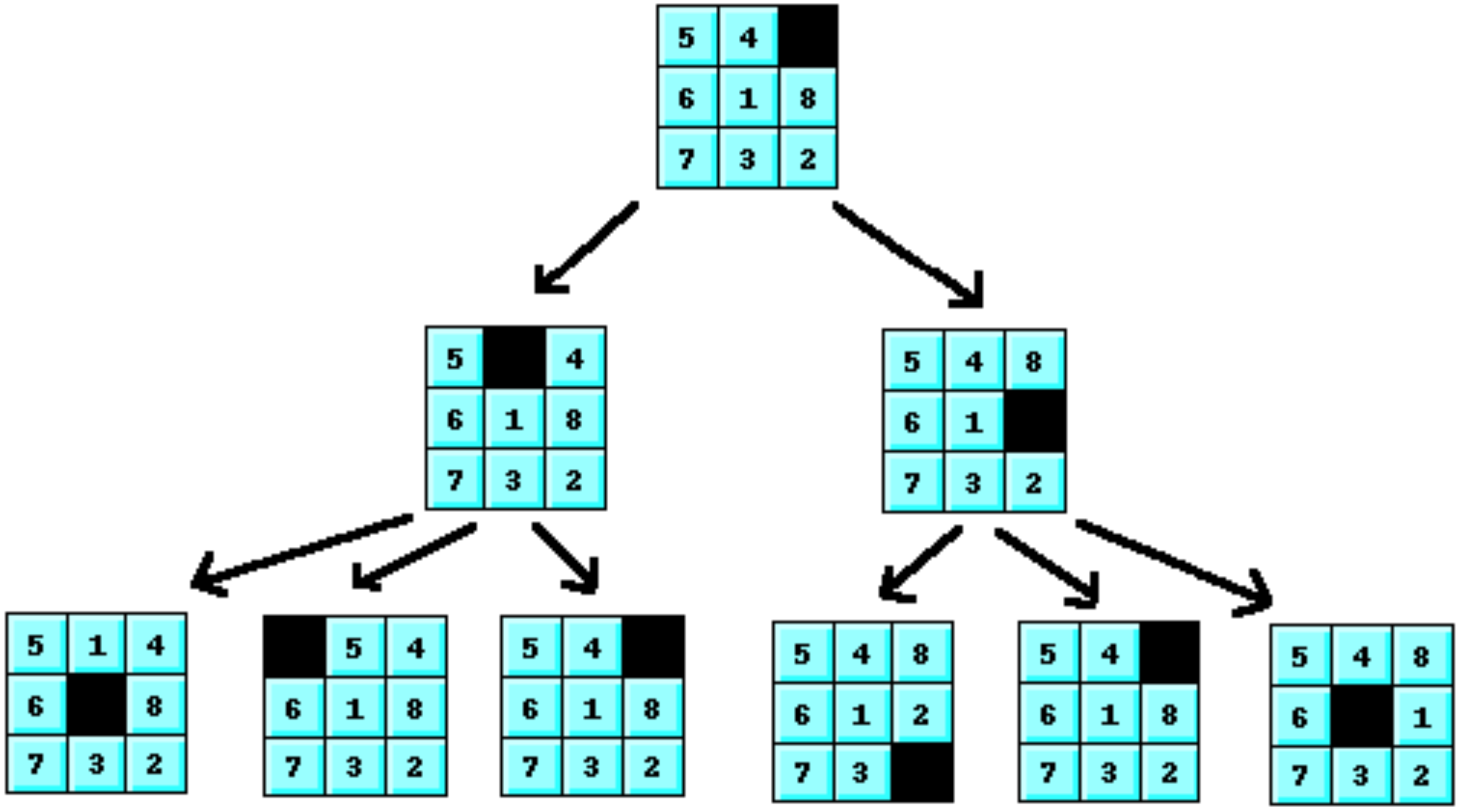
1	2	3
4	■	6
7	5	8

goal:

1	2	3
4	5	6
7	8	■

Legal moves: slide a piece next to empty slot.

Many AI problems (more basically computational problems) can be modeled as search problems.



A portion of a search tree for the 8-puzzle.

Combinatorial search

- Uninformed search
    - depth-first
    - breadth-first
    - iterative deepening
    - breadth-depth
- informed search
    - best-first
    - A*
    - IDA* etc.

Combinatorial search

Depth-first search

```
State DepthFirstSearch(node){
  if (goalTest(node)) return node;
  foreach (n in successors(node, operators)){
    result = DepthFirstSearch(n);
    if (result != FAIL) return result;
  }
```

- DFS may not even terminate.
- What are the ways to speed-up DFS?

Combinatorial search

- Breadth-first search

```
Queue q;
q.insert(initialState);
while (!q.isEmpty()){
  node = q.remove();
  if (goalTest(node)) return node;
  foreach (n in successors(node, operators))
    q.insert(n);

 }
 return FAIL;
```

Combinatorial search

heuristic search

- for each node, a heuristic provides an estimate of its distance from the goal.

- for sliding-piece puzzle, Manhattan distance is one such estimate.

- finding such a (good) estimate is difficult if we place additional condition that we should never overestimate.