Machine Learning

Instructor: Doina Precup

Email: dprecup@cs.mcgill.ca

Class web page:

http://www.cs.mcgill.ca/d̃precup/courses/Fall2001/ml.html

Outline

- Administrative issues
- What is machine learning?
- Why study machine learning?
- Formulating machine learning problems
- Machine learning questions

Administrative issues

- Class materials:
  - Tom Mitchell, Machine Learning (main text)
  - Additional readings: distributed in class and/or posted on the web page
  - Class notes: posted on the web page
- Prerequisites:
  - Knowledge of a programming language (e.g. C, C++, Java, LISP, Matlab)
  - Some AI background is recommended
  - Some probability theory and statistics background highly recommended

Evaluation

- 5 homework assignments (35%)
- Project (50%)
  - reading research papers on a chosen topic
  - implementing and/or experimenting with algorithms related to the topic
  - a written report on your findings
  - a class presentation (evaluated by everyone else)
- Reading assignments (20%)
- Participation to class discussions (up to 5% extra credit)

What is learning?

- H.Simon: Any process by which a system improves its performance
- M.Minsky: Learning is making useful changes in our minds
- Michalsky: Learning is constructing or modifying representations of what is being experienced
- Valiant: Learning is the process of knowledge acquisition in the absence of explicit programming

Why study machine learning?

- Easier to build a learning system than to hand-code a working program! E.g.:
  - Robot that learns a map of the environment by wandering around it
  - Programs that learn to play games by playing against themselves
- Improving on existing programs, e.g
  - Instruction scheduling and register allocation in compilers
  - Combinatorial optimization problems
- Discover knowledge and patterns in databases (data mining)
- Solving tasks that require a system to be adaptive, e.g.
  - Speech and handwriting recognition

  - “Intelligent” user interfaces
- Understanding animal and human learning
  - How do we learn language?
  - How do we recognize faces?

Very brief history

- Studied ever since computers were invented (e.g. Samuel's checkers player)
- Coined as "machine learning" in late 70s - early 80s
- Very active research field, several yearly conferences (e.g. ICML, NIPS), major journals (e.g. Machine Learning, Journal of Machine Learning Research)
- The time is right to start studying in the field!
  - Recent progress in algorithms and theory
  - Growing flood of on-line data to be analyzed
  - Computational power is available
  - Growing demand for industrial applications

Related disciplines

- Artificial intelligence
- Probability theory and statistics
- Computational complexity theory
- Control theory
- Information theory
- Philosophy
- Psychology and neurobiology

Three niches for machine learning

- Data mining : using historical data to improve decisions

  E.g. medical records $\rightarrow$ medical knowledge
- Software applications we cannot program by hand

  E.g. autonomous driving, speech recognition
- Self customizing programs

  E.g. Newsreader that learns user interests

Typical datamining task

Data:

Patient103 time=1	→	*Patient103* time=2	... →	*Patient103* time=n
Age: 23		Age: 23		Age: 23
FirstPregnancy: no		FirstPregnancy: no		FirstPregnancy: no
Anemia: no		Anemia: no		Anemia: no
Diabetes: no		Diabetes: YES		Diabetes: no
PreviousPrematureBirth: no		PreviousPrematureBirth: no		PreviousPrematureBirth: no
Ultrasound: ?		Ultrasound: abnormal		Ultrasound: ?
Elective C−Section: ?		Elective C−Section: no		Elective C−Section: no
Emergency C−Section: ?		Emergency C−Section: ?		**Emergency C−Section: Yes**
...		...		...

Given:

- 9714 patient records, each describing a pregnancy and birth
- Each patient record contains 215 features

Learn to predict: classes of future patients at high risk for

Emergency Cesarean Section

Datamining result

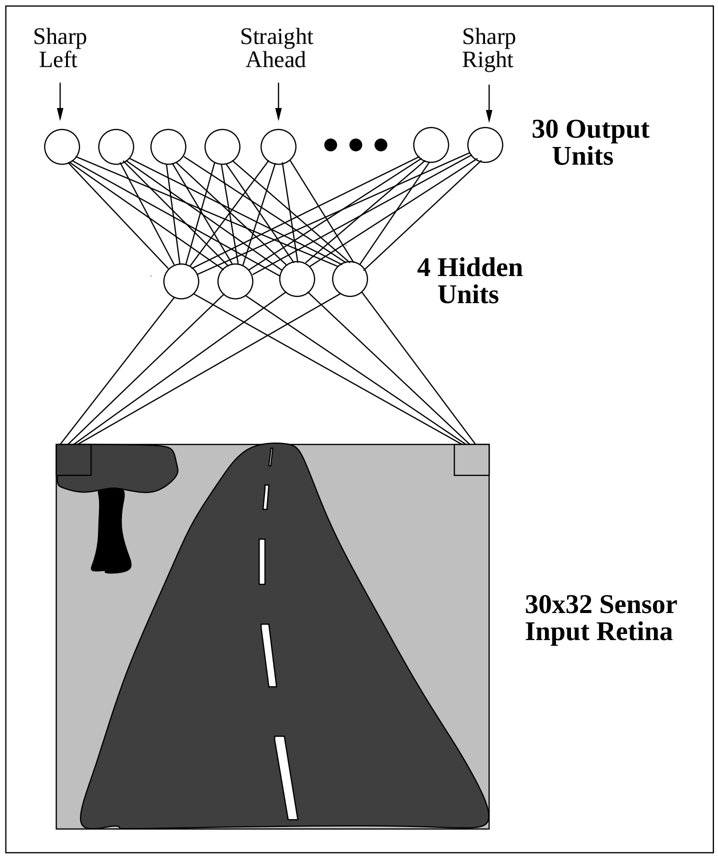
Patient103 time=1 →	*Patient103* time=2 ... →	*Patient103* time=n
Age: 23	Age: 23	Age: 23
FirstPregnancy: no	FirstPregnancy: no	FirstPregnancy: no
Anemia: no	Anemia: no	Anemia: no
Diabetes: no	Diabetes: YES	Diabetes: no
PreviousPrematureBirth: no	PreviousPrematureBirth: no	PreviousPrematureBirth: no
Ultrasound: ?	Ultrasound: abnormal	Ultrasound: ?
Elective C−Section: ?	Elective C−Section: no	Elective C−Section: no
Emergency C−Section: ?	Emergency C−Section: ?	**Emergency C−Section: Yes**
...	...	...

One of 18 learned rules:

```
If   No previous vaginal delivery, and
     Abnormal 2nd Trimester Ultrasound, and
     Malpresentation at admission
Then Probability of Emergency C-Section is 0.6
 Over training data: 26/41 = .63,
 Over test data: 12/20 = .60
```

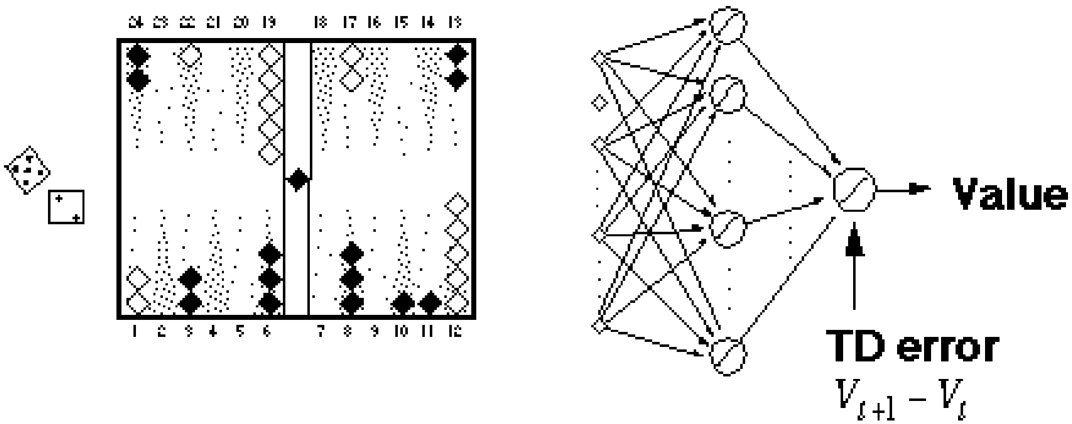
Problems too difficult to program by hand

ALVINN [Pomerleau] drives 70 mph on highways



Tesauro, 1992–1995

TD-Gammon



Action selection
by 2–3 ply search

Start with a random network

Play millions of games against self

Learn a value function from this simulated experience

This produces arguably the best player in the world

Software that Customizes to User

http://www.wisewire.com

What is the future?

Today: tip of the iceberg

- First-generation algorithms: neural nets, decision trees, regression ...
- Applied to well-formated database
- Budding industry

Opportunity for tomorrow: enormous impact

- Learn across multiple databases, plus the web and newsfeeds
- Learn by active experimentation
- Learn decisions rather than predictions
- Cumulative, lifelong learning
- Programming languages with learning embedded?

What is a learning problem?

Learning = Improving with experience at some task

More precisely:

- Improve over task T,
- with respect to performance measure P,
- based on experience E.

E.g. Learn to play checkers

- T: Play checkers
- P: % of games won in world tournament
- E: opportunity to play against self

Posing learning problems

	Task Definition	Performance Measure	Training Experience
Speech recognition			
Robot driving			
Language learning			

Type of Training Experience

- Direct or indirect?
- Teacher or not?

A problem: is training experience representative of performance goal?

Choose the target function

- $ChooseMove : Board \rightarrow Move$??
- $V : Board \rightarrow \Re$??
- ...

Possible Definition for Target Function V

- 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 a not a final 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.

This gives correct values, but is not operational

Choose Representation for Target Function

- Collection of rules?
- Neural network ?
- Polynomial function of board features?
- ...

A Representation for Learned Function

$$w_0 + w_1 \cdot bp(b) + w_2 \cdot rp(b) + w_3 \cdot bk(b) + w_4 \cdot rk(b) + w_5 \cdot bt(b) + w_6 \cdot rt(b)$$

- $bp(b)$: number of black pieces on board b
- $rp(b)$: number of red pieces on b
- $bk(b)$: number of black kings on b
- $rk(b)$: number of red kings on b
- $bt(b)$: number of red pieces threatened by black (i.e., which can be taken on black's next turn)
- $rt(b)$: number of black pieces threatened by red

Obtaining Training Examples

One rule for estimating training values:

$$V_{train}(b) \leftarrow \hat{V}(Successor(b))$$

where:

- $V(b)$: the true target function
- $\hat{V}(b)$: the learned function
- $V_{train}(b)$: the training value

Choose weight tuning rule

LMS Weight update rule:

Do repeatedly:

1. Select a training example b at random
2. Compute $error(b)$:

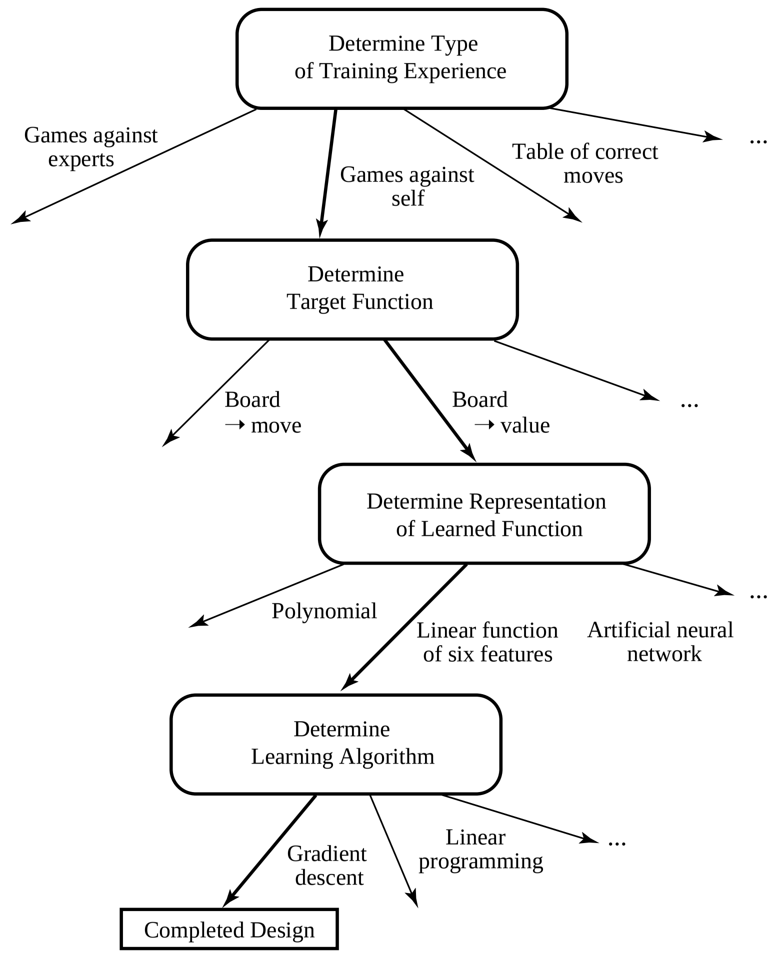
$$error(b) = V_{train}(b) - \hat{V}(b)$$

3. For each board feature f_i, update weight w_i:

$$w_i \leftarrow w_i + c \cdot f_i \cdot error(b)$$

c is some small constant, say 0.1, to moderate the rate of learning

Design Choices



Some Issues in Machine Learning

- What algorithms can approximate functions well (and when)?
- How does number of training examples influence accuracy?
- How does complexity of hypothesis representation impact it?
- How does noisy data influence accuracy?
- What are the theoretical limits of learnability?
- How can prior knowledge of learner help?
- What clues can we get from biological learning systems?
- How can systems alter their own representations?

Kinds of learning

- Supervised learning
  - Training experience: a set of **labeled examples** of the form $v_1\, v_2\, \ldots v_n, o$, where v_i are values for *input variables* and o is the *output*
  - What to learn: A **function** $f : V_1 \times V_2 \times \cdots \times V_n \rightarrow O$, which maps the input variables into the output domain
  - Performance measure: minimize the error on the training examples
- Reinforcement learning
  - Training experience: interaction with an environment; the agent receives a numerical reward signal
  - What to learn: a way of behaving that leads to a lot of reward

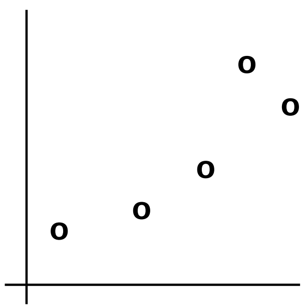
in the long run

- Performance measure: maximize the long-term reward

- Unsupervised learning
  - Training experience: unlabeled examples
  - What to learn: Interesting associations in the data
  - Performance measure: ?

Supervised (inductive) learning

Assume somebody gives us examples of what we are trying to learn



(a)

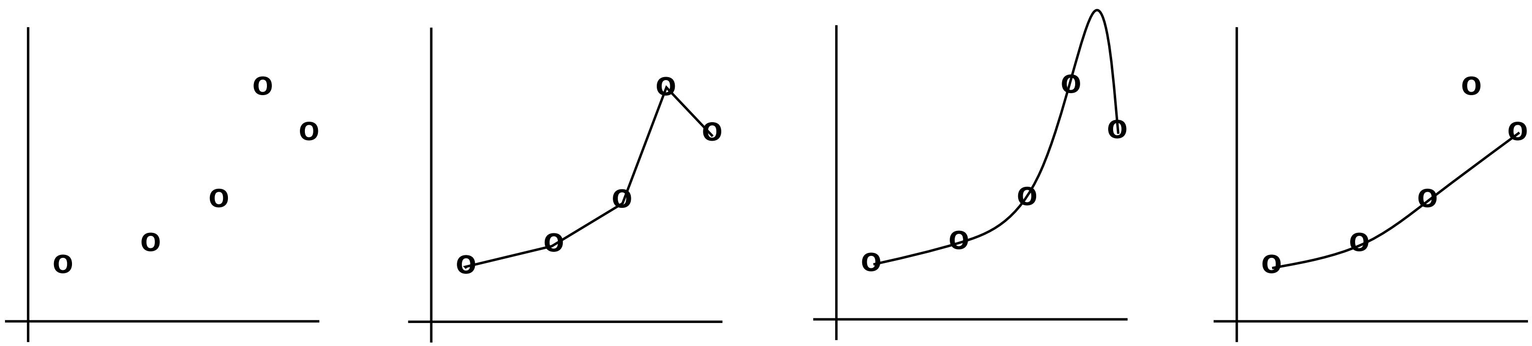
Try to find a *target function* that fits the examples

If the output domain is discrete, this problem is called *classification*

If the output domain is Boolean, this problem is called *concept learning*

If the output is continuous, the problem is known as *regression* of *function approximation*.

The big question



(a) (b) (c) (d)

There are an infinite number of functions that can fit the training points!

How do we find a *good* one?

I.e. one that will give approximately correct values for *unseen* data.

Choosing a target function representation

That's a bit of black magic... depends on intuition about the task.

We usually consider a *class of hypothesis*, and choose the one that fits the data best.

E.g. decision trees, artificial neural networks, linear combinations of 3rd-degree polynomials, linear combinations of the input etc. etc

If there are several hypothesis that fit the data about the same, *choose the simplest one* - **Occam's razor**

Learning methods usually *search* through the class of hypothesis to find a good one.