
Chapter 1: Introduction

CS 536: Machine Learning
Littman (Wu, TA)

Outline

- Why Machine Learning?
- What is a well-defined learning problem?
- An example: Learning to play checkers
- What questions should we ask about Machine Learning?

Why Machine Learning

- Recent progress in algorithms and theory
- Growing flood of online data
- Computational power is available
- Budding industry

Three Niches for ML

- Data mining: using historical data to improve decisions
 - medical records □ medical knowledge
- Software applications we can't program by hand
 - autonomous driving
 - speech recognition
- Self customizing programs
 - Newsreader that learns user interests

Typical Datamining Task

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

Patient 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
...

Datamining Result

Rules learned from synthesized data:

```
If    Other-Delinquent-Accounts > 2, and
      Number-Delinquent-Billing-Cycles > 1
Then  Profitable-Customer? = No
      [Deny Credit Card application]

If    Other-Delinquent-Accounts = 0, and
      (Income > $30k) OR (Years-of-Credit > 3)
Then  Profitable-Customer? = Yes
      [Accept Credit Card application]
```

Customer Purchase Behavior

$Cust103_{time=1}$	\rightarrow	$Cust103_{time=2}$	\rightarrow	$Cust103_{time=n}$
Sex: M		Sex: M	...	Sex: M
Age: 53		Age: 53		Age: 53
Income: \$50k		Income: \$50k		Income: \$50k
Own House: Yes		Own House: Yes		Own House: Yes
MS Products: Word		MS Products: Word		MS Products: Word
Computer: 386 PC		Computer: Pentium		Computer: Pentium
Purchase Excel?: ?		Purchase Excel?: ?		Purchase Excel?: Yes
...	

Customer Retention

*Cust103*_{time=1} → *Cust103*_{time=2} → *Cust103*_{time=n}

Sex: M	Sex: M	...	Sex: M
Age: 53	Age: 53		Age: 53
Income: \$50k	Income: \$50k		Income: \$50k
Own House: Yes	Own House: Yes		Own House: Yes
Checking: \$5k	Checking: \$20k		Checking: \$0k
Savings: \$15k	Savings: \$0k		Savings: \$0k
Current-customer?: yes	Current-customer?: yes		Current-customer?: no

Process Optimization

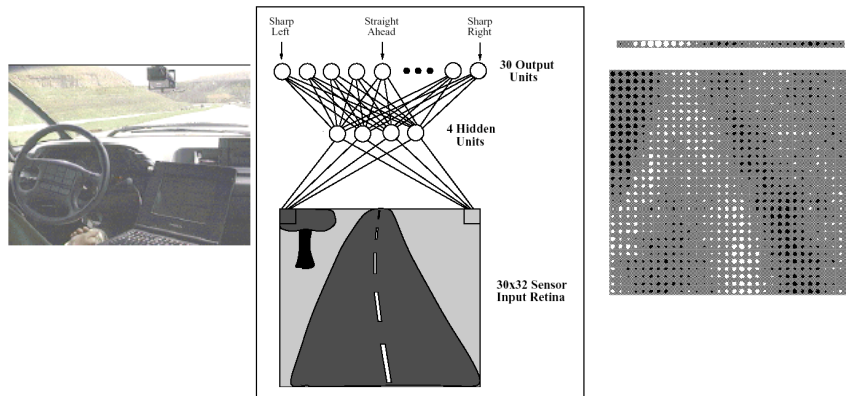
*Product72*_{time=1} → *Product72*_{time=2} → *Product72*_{time=n}

Stage: mix	Stage: cook	...	Stage: cool
Mixing-speed: 60rpm	Temperature: 325		Fan-speed: medium
Viscosity: 1.3	Viscosity: 3.2		Viscosity: 1.3
Fat content: 15%	Fat content: 12%		Fat content: 12%
Density: 2.8	Density: 1.1		Density: 1.2
Spectral peak: 2800	Spectral peak: 3200		Spectral peak: 3100
Product underweight?: ?	Product underweight?: ?		Product underweight?: Yes

Too Difficult to Program

Problems too difficult to program by hand

ALVINN [Pomerleau] drives 70mph on highways



Software Customizes to User

<http://www.wisewire.com>

April 30, 1998

Lycos Acquires WiseWire

By internetnews.com Staff

Lycos, Inc. today announced the acquisition of targeted-content provider WiseWire Corp. for around \$39.75 million in Lycos shares.

Under the acquisition, the three-year old, Pittsburgh-based WiseWire's automated directory will be incorporated into Lycos' search page. Lycos said it will be the first online service using the technology which is based on user input and an intelligent agent product.



Where Is This Headed?

Today: tip of the iceberg

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

Looking to Tomorrow

Opportunity for tomorrow: enormous impact

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

Relevant Disciplines

- Artificial intelligence
- Bayesian methods
- Computational complexity theory
- Control theory
- Information theory
- Philosophy
- Psychology and neurobiology
- Statistics
- ...

What is the Learning Problem?

Learning = Improving with experience at some task

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

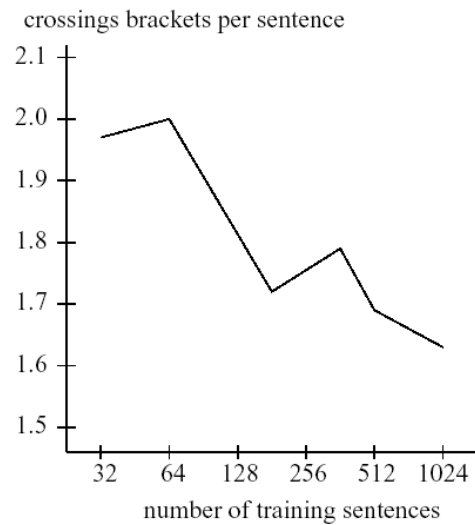
E.g., Learn to play checkers

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

Learning Curves

- T : parsing Korean
- P : “crossing brackets”
- E : training sentences

From “Rapid Parser Development: A Machine Learning Approach for Korea” by Ulf Hermjakob



Other Kinds of “Learning”?

"Tamagotchi is a tiny pet from cyberspace who needs your love to survive and grow. If you take good care of your Tamagotchi pet, it will slowly grow bigger, healthier, and more beautiful every day. But if you neglect your little cyber creature, your Tamagotchi may grow up to be mean or ugly. How old will your Tamagotchi be when it returns to its home planet? What kind of virtual caretaker will you be?" - Bandai

<http://www.mimitchi.com/html/q1.htm>



Learning to Play Checkers

- *T*: Play checkers
- *P*: Percent of games won in world tournament
- What experience?
- What exactly should be learned?
- How shall it be represented?
- What specific algorithm to learn it?

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 \square Move ??
- *V*: Board \square ??
- ...

Possible Definition for *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.

Target Function Representation

- collection of rules?
- neural network?
- polynomial function of board features?
- ...

A Linear Representation

$$\begin{aligned}V(b) &= w_0 + w_1 bp(b) + w_2 rp(b) + w_3 bk(b) \\ &\quad + w_4 rk(b) + w_5 bt(b) + w_6 rt(b) \\ &= w \cdot f(b)\end{aligned}$$

- $bp(b)$: 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

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

One rule for estimating training values:

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

What's a Successor?

From the textbook:

- $Successor(b)$

Alternative:

- $Successor(b)$

Choose Weight Tuning Rule

LMS Weight update rule:

Do repeatedly:

- Select a training example b at random

1. Compute $error(b)$:

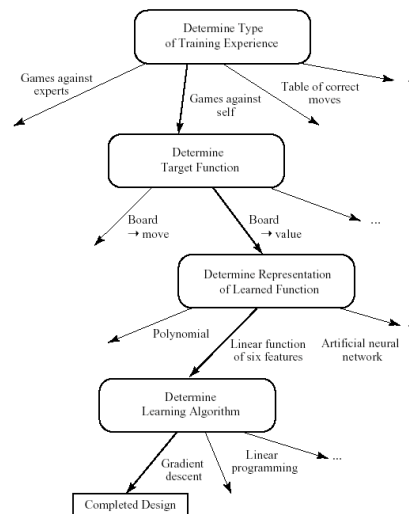
$$error(b) = V_{train}(b) - \hat{V}(b)$$

2. For each board feature $f_i(b)$, update weight w_i :

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

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

Design Choices



Some Issues in ML

- 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?