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 machine learning:
- 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

Data:

<table>
<thead>
<tr>
<th>Patient101 time=1</th>
<th>Patient101 time=2</th>
<th>Patient101 time=n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age: 23</td>
<td>FirstPregnancy: no</td>
<td>Anemia: no</td>
</tr>
</tbody>
</table>

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

Data:

<table>
<thead>
<tr>
<th>Patient103 time=1</th>
<th>Patient103 time=2</th>
<th>Patient103 time=n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age: 23</td>
<td>FirstPregnancy: no</td>
<td>Anemia: no</td>
</tr>
</tbody>
</table>

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
Credit Risk Analysis

Data:

Customer103: (time=t0)  
Years of credit: 9  
Loan balance: $2,400  
Income: $52k  
Own House: Yes  
Other delinquent accts: 2  
Max billing cycles late: 3  
Profitable customer?: ?

Customer103: (time=t1)  
Years of credit: 9  
Loan balance: $3,250  
Income: ?  
Own House: Yes  
Other delinquent accts: 2  
Max billing cycles late: 4  
Profitable customer?: ?

Customer103: (time=tn)  
Years of credit: 9  
Loan balance: $4,500  
Income: ?  
Own House: Yes  
Other delinquent accts: 3  
Max billing cycles late: 6  
Profitable customer?: ?

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]

Other Prediction Problems

Customer purchase behavior:

Customer103: (time=t0)  
Sec: M  
Age: 53  
Income: $50k  
Own House: Yes  
MS Products: Word  
Computer: Pentium  
Purchase Excel?: ?

Customer103: (time=t1)  
Sec: M  
Age: 53  
Income: $50k  
Own House: Yes  
MS Products: Word  
Computer: Pentium  
Purchase Excel?: Yes

Customer retention:

Customer103: (time=t0)  
Sec: M  
Age: 53  
Income: $50k  
Own House: Yes  
Checking: $20k  
Savings: $0  
Current-customer?: yes  
Current-customer?: ?

Customer103: (time=t1)  
Sec: M  
Age: 53  
Income: $50k  
Own House: Yes  
Checking: $20k  
Savings: $0  
Current-customer?: ?

Process optimization:

Product72: (time=t0)  
Stage mix: Stage cool  
Temperature: 350  
Density: 1.1  
Spectral peak: 3100  
Product underweight?: ?

Product72: (time=t1)  
Stage mix: Stage cool  
Temperature: 350  
Density: 1.1  
Spectral peak: 3100  
Product underweight?: Yes

Problems Too Difficult to Program by Hand

ALVINN [Pomerleau] drives 70 mph on highways

Software that Customizes to User

http://www.wisewire.com
Where Is this Headed?

Today: tip of the iceberg
- First-generation algorithms: neural nets, decision trees, regression ...
- Applied to well-formatted database
- Budding industry

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 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

• \textit{ChooseMove} : \textit{Board} \rightarrow \textit{Move} ??
• \( V : \textit{Board} \rightarrow \mathbb{R} ?? \)
• ...

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

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

One rule for estimating training values:
- \( V_{\text{train}}(b) \leftarrow \hat{V}(\text{Successor}(b)) \)

Choose Weight Tuning Rule

LMS Weight update rule:

Do repeatedly:
- Select a training example \( b \) at random
  1. Compute \( \text{error}(b) \):
     \[ \text{error}(b) = V_{\text{train}}(b) - \hat{V}(b) \]
  2. For each board feature \( f_i \), update weight \( w_i \):
     \[ w_i \leftarrow w_i + c \cdot f_i \cdot \text{error}(b) \]

\( c \) 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?