Machine Learning

CSCI 5582, Fall 2007
Assignments

• To read this week: Chapter 18
Machine Learning: Basic Themes

• Method for learning
  (E.g., by analogy; by examples; by direct explanation of concepts or rules; by discovery)

• What, precisely, is being learned?
  (Numeric parameters; classifications or decision trees; rules; grammars or languages; graphs…)

• Type of “architecture” employed
  (E.g., symbolic vs. connectionist)
Different Styles of Feedback for Learning Programs

• *Supervised learning*
  We are given input and output to a function (say, a classification function) and need to learn the function itself.

• *Reinforcement learning*
  We perform actions (possibly in response to input) and are given reward or punishment to reinforce or change those actions.

• *Unsupervised learning*
  We perform actions (possibly in response to input) and must interpret for ourselves, over time, whether those actions should be strengthened or modified.
A “Mystery Concept” (an example of supervised learning)

- (A A B A) NO
- (A B B A B B A) YES
- (B B B) YES
- (A B A B A B) NO
- (B B B A B B B) YES
Some Terminology for Supervised Learning

• Here, the concept to be learned is a function from examples (say, strings over the alphabet \{a,b\}; or pictures composed of black-and-white pixels; or 3-D objects) to the boolean set \{T, F\}. Sometimes we just call this function the goal concept or goal predicate.

• A hypothesis is the learner's current "best guess" at the goal concept.

• A training sample (or training set) is a sequence of classified examples:
  - (a a b a) no
  - (a b b a b b a) yes
  - (b b b) yes

• A learning algorithm, for this sort of supervised situation, is a function from samples to hypotheses
Key Issues in Supervised Learning Scenarios

- Is the training sample presented all at once, or one element at a time? (In the latter case, we think of the learning algorithm creating a sequence of hypotheses, each in response to some fresh input.)
- Can the learner (machine) propose examples, or are examples only provided from the external environment?
- Are samples "representative"? (For instance, are we any more likely to get an unusual example in the training sample than in "real life"?)
- Are training samples error-free? (Are they ever misclassified?) Might the sample-provider ("teacher") ever deliberately try to mislead us?
- Are there limitations on the types of concepts we know we are supposed to learn? Are there limitations on the types of hypotheses we are allowed to construct?
Learning a Decision Tree

• We look at someone making classification decisions, and try to infer the rule that they are using. (E.g., we might look at someone choosing videos and try to predict whether they think a particular title is a “good” video or not.)

• We assume that their rule can be written as a tree in which each node represents a local decision based on an attribute.
Some Attributes for Mike’s Video Choices

• Do things blow up? (Tends to be good.)
• Is the title written in script? (Tends to be bad.)
• Is it a sequel? (Tends to be bad.)
• Is there a monster? (Tends to be good.)
• Is it based on a TV show? (Tends to be bad.)
Decision Trees and Their Limitations

• How many distinct decision rules are there for N attributes?

There are $2^N$ possible values of the attributes, so a classification rule can always be represented as a table of $2^N$ entries. Thus, there are $2^{(2^N)}$ possible rules.
A Training Set of Examples: We Watch Mike Make Lots of Video Choices

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Y</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>N</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

… and others…
Building a Tree from Our Examples

Suppose we have 12 attributes, and 200 examples of Mike’s video choices: 100 positive and 100 negative.
Now, the crucial question: which attribute is most important to Mike?
Which is More Important?

• Attribute A divides the set as follows:
  (70 yes, 30 no) for A true
  (30 yes, 70 no) for A false

• Attribute B divides the set as follows:
  (100 yes, 90 no) for B true
  (0 yes, 10 no) for B false
Information as a Criterion for Attribute Values:

Information value for a set of probabilities:

$$\Sigma (- P_i) \log_2(P_i)$$

So, for a standard coin flip, the information is $2 \times (-1/2) \times (\log 1/2) = 1$ bit
Attribute A

Information at start:
\(-\frac{1}{2} \cdot \log(1/2) + -\frac{1}{2} \cdot \log (1/2) = 1\)

Information after Attribute A:
Choice 1, weighted by 0.5:
\(-0.7 \cdot \log(0.7) + -0.3 \cdot (\log 0.3)\)
Choice 2, weighted by 0.5
\(-0.3 \cdot \log(0.3) + -0.7 \cdot (\log 0.7)\)

Total information: \(0.44065 + 0.4406 = 0.881\)
Attribute B

Choice 1, weighted by 0.95
-10/19 * (log 10/19) + -9/19(log 9/19)

Choice 2, weighted by 0.05
-0 * (log 0) + -1 (log 1)

Total information value:
0.948 + 0

The change in information by using Attribute A is greater, so A is the more informative attribute.
Splitting Examples into “Training” and “Test” Sets

- Given our initial set of examples, split it into a (randomly-chosen) training set and a test set.
- Once the algorithm has generated a tree for the training set, use the test set to gauge the accuracy of the tree (measure the percent of the test set that is correctly classified).
- Repeat this process: we should see that (for large training sets) we converge on a high accuracy for the test set.