Today 11/16

- Finish up ILP/FOIL
- Break
  - HW questions
- Quiz review
  - Probabilistic sequence processing
  - Supervised ML
    - Learning Classifiers
      - DTs, DLs, Naïve Bayes, Ensembles, SVMs
    - Concept Learning
      - Version Spaces, FOIL
Relational Learning and Inductive Logic Programming

- Fixed feature vectors are a very limited representation of objects.
- Examples or target concept may require relational representation that includes multiple entities with relationships among them.
- First-order predicate logic is a more powerful representation for handling such relational descriptions.

ILP Example

- Learn definitions of family relationships given data for primitive types and relations.
  
  \[
  \text{brother}(A,C), \text{parent}(C,B) \rightarrow \text{uncle}(A,B) \\
  \text{husband}(A,C), \text{sister}(C,D), \text{parent}(D,B) \rightarrow \text{uncle}(A,B)
  \]

- Given the relevant predicates and a database populated with positive and negative examples
- By database I mean sets of tuples for each of the relevant relations
FOIL
First-Order Inductive Logic

• Top-down sequential covering algorithm to learn first order theories.
• Background knowledge provided extensionally (i.e., A model)
• Start with the most general rule possible. (T → P(x))
• Specialize it on demand...
• Specializations of a clause include adding all possible literals one at a time to the antecedent...
  - A → P
  - B → P
  - C → P...
Where A, B and C are predicates already in the domain theory.

We’re working top-down from the most general hypothesis so what’s driving things?

CSCI 5582 Fall 2006

---

FOIL

• At a high level.
  - Start with the most general H
  - Repeatedly constructs clauses that cover a subset of the positive examples and none of the negative examples.
  - Then remove the covered positive examples
  - Constructs another clause
  - Repeat until all the positive examples are covered.

CSCI 5582 Fall 2006
FOIL

• Constructing candidate clauses
  - Any predicate (negated or not), args are variables
    - But every predicate must contain a variable from an earlier literal or the head of the clause (antecedent)
  - Equality constraints on variables
  - Some arithmetic

FOIL Training Data

• Background knowledge consists of complete set of tuples for each background predicate for this universe.
• Example: Consider learning a definition for the target predicate path for finding a path in a directed acyclic graph.

```plaintext
edge(X,Y) -> path(X,Y)
edge(X,Z) ^ path(Z,Y) -> path(X,Y)
```

edge: \{<1,2>,<1,3>,<3,6>,<4,2>,<4,6>,<6,5>\}
path: \{<1,2>,<1,3>,<1,6>,<1,5>,<3,6>,<3,5>,<4,2>,<4,6>,<4,5>,<6,5>\}
FOIL Negative Training Data

- Negative examples of target predicate can be provided directly, or generated indirectly by making a closed world assumption.
  - Every pair of constants \( \langle X,Y \rangle \) not in positive tuples for path predicate.

Negative path tuples:
\[
\{ \langle 1,1 \rangle, \langle 1,4 \rangle, \langle 2,1 \rangle, \langle 2,2 \rangle, \langle 2,3 \rangle, \langle 2,4 \rangle, \langle 2,5 \rangle, \langle 2,6 \rangle,
\langle 3,1 \rangle, \langle 3,2 \rangle, \langle 3,3 \rangle, \langle 3,4 \rangle, \langle 4,1 \rangle, \langle 4,3 \rangle, \langle 4,4 \rangle, \langle 5,1 \rangle,
\langle 5,2 \rangle, \langle 5,3 \rangle, \langle 5,4 \rangle, \langle 5,5 \rangle, \langle 5,6 \rangle, \langle 6,1 \rangle, \langle 6,2 \rangle, \langle 6,3 \rangle,
\langle 6,4 \rangle, \langle 6,6 \rangle \}
\]

Sample FOIL Induction

\[
\text{Pos: } \{ \langle 1,2 \rangle, \langle 1,3 \rangle, \langle 1,6 \rangle, \langle 1,5 \rangle, \langle 3,6 \rangle, \langle 3,5 \rangle,
\langle 4,2 \rangle, \langle 4,6 \rangle, \langle 4,5 \rangle, \langle 6,5 \rangle \}
\]
\[
\text{Neg: } \{ \langle 1,1 \rangle, \langle 1,4 \rangle, \langle 2,1 \rangle, \langle 2,2 \rangle, \langle 2,3 \rangle, \langle 2,4 \rangle, \langle 2,5 \rangle, \langle 2,6 \rangle,
\langle 3,1 \rangle, \langle 3,2 \rangle, \langle 3,3 \rangle, \langle 3,4 \rangle, \langle 4,1 \rangle, \langle 4,3 \rangle, \langle 4,4 \rangle, \langle 5,1 \rangle,
\langle 5,2 \rangle, \langle 5,3 \rangle, \langle 5,4 \rangle, \langle 5,5 \rangle, \langle 5,6 \rangle, \langle 6,1 \rangle, \langle 6,2 \rangle, \langle 6,3 \rangle,
\langle 6,4 \rangle, \langle 6,6 \rangle \}
\]

Start with clause:
\[
T \rightarrow \text{path}(X,Y)
\]
Possible literals to add:
\[
\text{edge}(X,X), \text{edge}(Y,Y), \text{edge}(X,Y), \text{edge}(Y,X), \text{edge}(X,Z),
\text{edge}(Y,Z), \text{edge}(Z,X), \text{edge}(Z,Y), \text{path}(X,X), \text{path}(Y,Y),
\text{path}(X,Y), \text{path}(Y,X), \text{path}(X,Z), \text{path}(Y,Z), \text{path}(Z,X),
\text{path}(Z,Y), X=Y,
\]
plus negations of all of these. CSCI 5582 Fall 2006
Sample FOIL Induction

Pos: \{<1,2>,<1,3>,<1,6>,<1,5>,<3,6>,<3,5>,<4,2>,<4,6>,<4,5>,<6,5>\}

Neg: \{<1,1>,<1,4>,<2,1>,<2,2>,<2,3>,<2,4>,<2,5>,<2,6>,<3,1>,<3,2>,<3,3>,<3,4>,<4,1>,<4,3>,<4,4>,<5,1>,<5,2>,<5,3>,<5,4>,<5,5>,<5,6>,<6,1>,<6,2>,<6,3>,<6,4>,<6,6>\}

Test:
\texttt{edge(X,X) -> path(X,Y)}

Covers 0 positive examples
Covers 6 negative examples
Not a good literal to try.

CSCI 5582 Fall 2006
Sample FOIL Induction

Pos: \{<1, 6>, <1, 5>, <3, 5>, <4, 5>\}

Neg: \{<1, 1>, <1, 4>, <2, 1>, <2, 2>, <2, 3>, <2, 4>, <2, 5>, <2, 6>, <3, 1>, <3, 2>, <3, 3>, <3, 4>, <4, 1>, <4, 3>, <4, 4>, <5, 1>, <5, 2>, <5, 3>, <5, 4>, <5, 5>, <5, 6>, <6, 1>, <6, 2>, <6, 3>, <6, 4>, <6, 6>\}

Test: 
edge(X, Y) \rightarrow \text{path}(X, Y)

Covers 6 positive examples
Covers 0 negative examples
Chosen as best literal. Result is base clause.

Remove covered positive tuples.

Sample FOIL Induction

Pos: \{<1, 6>, <1, 5>, <3, 5>, <4, 5>\}

Neg: \{<1, 1>, <1, 4>, <2, 1>, <2, 2>, <2, 3>, <2, 4>, <2, 5>, <2, 6>, <3, 1>, <3, 2>, <3, 3>, <3, 4>, <4, 1>, <4, 3>, <4, 4>, <5, 1>, <5, 2>, <5, 3>, <5, 4>, <5, 5>, <5, 6>, <6, 1>, <6, 2>, <6, 3>, <6, 4>, <6, 6>\}

Start new clause
T \rightarrow \text{path}(X, Y)
Sample FOIL Induction

Pos: \{\langle 1,6\rangle,\langle 1,5\rangle,\langle 3,5\rangle,
\langle 4,5\rangle\}

Neg: \{\langle 1,1\rangle,\langle 1,4\rangle,\langle 2,1\rangle,\langle 2,2\rangle,\langle 2,3\rangle,\langle 2,4\rangle,\langle 2,5\rangle,\langle 2,6\rangle,
\langle 3,1\rangle,\langle 3,2\rangle,\langle 3,3\rangle,\langle 3,4\rangle,\langle 4,1\rangle,\langle 4,4\rangle,\langle 5,1\rangle,\langle 5,2\rangle,\langle 5,3\rangle,\langle 5,4\rangle,\langle 5,5\rangle,\langle 5,6\rangle,\langle 6,1\rangle,\langle 6,2\rangle,\langle 6,3\rangle,
\langle 6,4\rangle,\langle 6,6\rangle\}

Test:
edge(X,Y) \rightarrow path(X,Y)
Covers 0 positive examples
Covers 0 negative examples
Not a good literal.

---

Sample FOIL Induction

Pos: \{\langle 1,6\rangle,\langle 1,5\rangle,\langle 3,5\rangle,
\langle 4,5\rangle\}

Neg: \{\langle 1,1\rangle,\langle 1,4\rangle,\langle 2,1\rangle,\langle 2,2\rangle,\langle 2,3\rangle,\langle 2,4\rangle,\langle 2,5\rangle,\langle 2,6\rangle,
\langle 3,1\rangle,\langle 3,2\rangle,\langle 3,3\rangle,\langle 3,4\rangle,\langle 4,1\rangle,\langle 4,3\rangle,\langle 4,4\rangle,\langle 5,1\rangle,\langle 5,2\rangle,\langle 5,3\rangle,\langle 5,4\rangle,\langle 5,5\rangle,\langle 5,6\rangle,\langle 6,1\rangle,\langle 6,2\rangle,\langle 6,3\rangle,
\langle 6,4\rangle,\langle 6,6\rangle\}

Test:
edge(X,2) \rightarrow path(X,Y)
Covers all 4 positive examples
Covers 14 of 26 negative examples
Eventually chosen as best possible literal
Sample FOIL Induction

Pos:  \{<1,6>,<1,5>,<3,5>,
       <4,5>\}

Neg:  \{<1,1>,<1,4>,
       <3,1>,<3,2>,<3,3>,<3,4>,<4,1>,<4,3>,<4,4>,
       <6,1>,<6,2>,<6,3>,
       <6,4>,<6,6>\}

Test:
  \text{edge}(X,Z) \rightarrow \text{path}(X,Y)

Covers all 4 positive examples  Covers 14 of 26 negative examples
Eventually chosen as best possible literal
Negatives still covered, remove uncovered examples.

Sample FOIL Induction

Pos:  \{<1,6,2>,<1,6,3>,<1,5>,<3,5>,
       <4,5>\}

Neg:  \{<1,1>,<1,4>,
       <3,1>,<3,2>,<3,3>,<3,4>,<4,1>,<4,3>,<4,4>,
       <6,1>,<6,2>,<6,3>,
       <6,4>,<6,6>\}

Test:
  \text{Edge}(X,Z) \rightarrow \text{path}(X,Y)

Covers all 4 positive examples  Covers 14 of 26 negative examples
Eventually chosen as best possible literal
Negatives still covered, remove uncovered examples.
Expand tuples to account for possible \(Z\) values. \(<X,Y,Z>\)
Sample FOIL Induction

Pos: \{<1, 6, 2>, <1, 6, 3>, <1, 5, 2>, <1, 5, 3>, <3, 5>, <4, 5>\}

Neg: \{<1, 1>, <1, 4>, <3, 1>, <3, 2>, <3, 3>, <3, 4>, <4, 1>, <4, 3>, <4, 4>, <6, 1>, <6, 2>, <6, 3>, <6, 4>, <6, 6>\}

Test: \text{edge}(X, Z) \rightarrow \text{path}(X, Y)

Covers all 4 positive examples
Covers 14 of 26 negative examples
Eventually chosen as best possible literal
Negatives still covered, remove uncovered examples.
Expand tuples to account for possible Z values. <X, Y, Z>

Sample FOIL Induction

Pos: \{<1, 6, 2>, <1, 6, 3>, <1, 5, 2>, <1, 5, 3>, <3, 5, 6>, <4, 5>\}

Neg: \{<1, 1>, <1, 4>, <3, 1>, <3, 2>, <3, 3>, <3, 4>, <4, 1>, <4, 3>, <4, 4>, <6, 1>, <6, 2>, <6, 3>, <6, 4>, <6, 6>\}

Test: \text{edge}(X, Z) \rightarrow \text{path}(X, Y)

Covers all 4 positive examples
Covers 14 of 26 negative examples
Eventually chosen as best possible literal
Negatives still covered, remove uncovered examples.
Expand tuples to account for possible Z values. <X, Y, Z>
Sample FOIL Induction

Pos: \{<1,6,2>,<1,6,3>,<1,5,2>,<1,5,3>,<3,5,6>,<4,5,2>,<4,5,6>\}

Neg: \{<1,1>,<1,4>,<3,1>,<3,2>,<3,3>,<3,4>,<4,1>,<4,3>,<4,4>,<6,1>,<6,2>,<6,3>,<6,4>,<6,6>\}

Test: edge (X, Z) \rightarrow path (X, Y)

Covers all 4 positive examples
Covers 14 of 26 negative examples
Eventually chosen as best possible literal
Negatives still covered, remove uncovered examples.
Expand tuples to account for possible Z values. <X,Y,Z>
Sample FOIL Induction

Pos: \{<1, 6, 2>, <1, 6, 3>, <1, 5, 2>, <1, 5, 3>, <3, 5, 6>, 
<4, 5, 2>, <4, 5, 6>\}

Neg: \{<1, 1, 2>, <1, 1, 3>, <1, 4, 2>, <1, 4, 3>, <3, 1, 6>, <3, 2, 6>, 
<3, 3, 6>, <3, 4, 6>, <4, 1, 2>, <4, 1, 6>, <4, 3, 2>, <4, 3, 6> 
<4, 4, 2>, <4, 4, 6>, <6, 1, 5>, <6, 2, 5>, <6, 3, 5>, 
<6, 4, 5>, <6, 6, 5>\}

Continue specializing clause:
\text{edge}(X, Z) → \text{path}(X, Y)

Covers 3 positive examples
Covers 0 negative examples
Sample FOIL Induction

Pos: \{<1, 6, 2>, <1, 6, 3>, <1, 5, 2>, <1, 5, 3>, <3, 5, 6>, <4, 5, 2>, <4, 5, 6>\}

Neg: \{<1, 1, 2>, <1, 1, 3>, <1, 4, 2>, <1, 4, 3>, <3, 1, 6>, <3, 2, 6>, <3, 3, 6>, <3, 4, 6>, <4, 1, 2>, <4, 1, 6>, <4, 3, 2>, <4, 3, 6>, <4, 4, 2>, <4, 4, 6>, <6, 1, 5>, <6, 2, 5>, <6, 3, 5>, <6, 4, 5>, <6, 6, 5>\}

Test:
\[\text{edge}(X, Z) \land \text{path}(Z, Y) \rightarrow \text{path}(X, Y)\]

Covers 4 positive examples \hspace{1cm} Covers 0 negative examples
Eventually chosen as best literal; completes clause.
Definition complete, since all original \(X,Y\) tuples are covered (by way of covering at least one of each positive \(X,Y,Z\) tuple.)

More Realistic Applications

- Classifying chemical compounds as mutagenic (cancer causing) based on their graphical molecular structure and chemical background knowledge.
- Classifying web documents based on both the content of the page and its links to and from other pages with particular content.
  - A web page is a university faculty home page if:
    - It contains the words “Professor” and “University”, and
    - It is pointed to by a page with the word “faculty”, and
    - It points to a page with the words “course” and “exam”
Rule Learning and ILP

Summary

• There are effective methods for learning symbolic rules from data using greedy sequential covering and top-down or bottom-up search.

• These methods have been extended to first-order logic to learn relational rules and recursive Prolog programs.

• Knowledge represented by rules is generally more interpretable by people, allowing human insight into what is learned and possible human approval and correction of learned knowledge.

Break

• HW questions?
  - Elizabeth?
Break

- The next quiz will be on 11/28.
- It will cover the ML material and the probabilistic sequence material.
- The readings for this quiz are:
  - Chapter 18
  - Chapter 19
  - Chapter 20: 712-718
  - HMM chapter posted on the web

Quiz Topics

- Sequence processing
- Classifiers
- Concept Learning
Probabilistic Sequence Processing

- Reading: Assigned Chapter
  - Know the three problems
    - \( P(\text{observations} \mid \text{model}) \)
    - \( \text{Argmax} \ P(\text{state sequence} \mid \text{observations, model}) \)
    - \( \text{Argmax} \ P(\text{model parameter} \mid \text{observations, model structure}) \)

Probabilistic Sequence Processing

- The basis for the techniques
  - Independence assumptions
  - For a first order HMM
    1. ?
    2. ?
Probabilistic Sequence Processing

- The basis for the techniques
  - Independence assumptions
  - For a first order HMM
    1. Current state depends only on the previous
    2. ?
Probabilistic Sequence Processing

- Know the computations
- Know the algorithms

Classifiers

- Chapter 18
- Basic induction task
  - DT, DL, Naïve Bayes
  - Ensembles (bagging, boosting)
  - For SVMs, just focus on the two main ideas (not the math)
- Learning theory
  - What are the key elements?
Classifiers

Theory: three main things we discussed:
1. Size of the hypothesis space
2. ?
3. ?
Classifiers

Theory: three main things we discussed:
1. Size of the hypothesis space
2. Number of training examples
3. Occam
Classifiers

Practice:
1. Training and testing
2. Learning as search
   1. Managing choices
   2. Making choices
   3. Termination conditions

Training Set

<table>
<thead>
<tr>
<th>Example</th>
<th>Attributes</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Alt</td>
<td>Bar</td>
</tr>
<tr>
<td>X₁</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>X₂</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>X₃</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X₄</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>X₅</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>X₆</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X₇</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X₈</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X₉</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X₁₀</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X₁₁</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X₁₂</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Concept Learning

- Chapter 19:
- Focus on what the task is
- How it’s different from classifier induction
  - Version space learning
  - How FOIL works