PURDUE
UNIVERSITY

CS590D: Data Mining

Prof. Chris Clifton

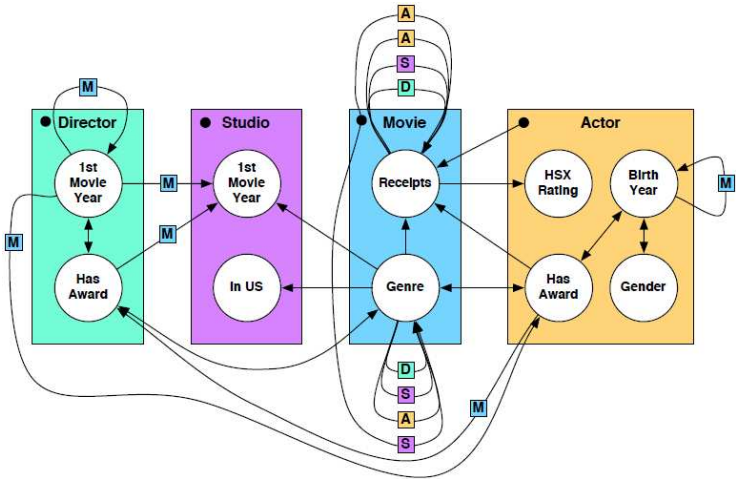
April 21, 2005

Multi-Relational Data Mining

What is MRDM?

- Problem: Data in multiple tables
  - Want rules/patterns/etc. across tables
- Solution: Represent as single table
  - Join the data
  - Construct a single view
  - Use standard data mining techniques
- Example: "Customer" and "Married-to"
  - Easy single-table representation
- Bad Example: *Ancestor of*

Relational Data Network



Basis of Solutions: Inductive Logic Programming

- ILP Rule:
  - customer(CID,Name,Age,yes)
    Age > 30 ∧ purchase(CID,PID,D,Value,PM) ∧
    PM = credit card ∧ Value > 100
- Learning methods:
  - Database represented as clauses (rules)
  - Unification: Given rule (function/clause), discover values for which it holds

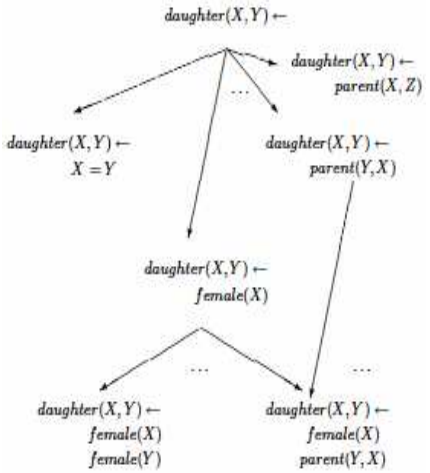
Example

- How do we learn the “daughter” relationship?
  - Is this classification? Association?
- Covering Algorithm: “guess” at rule explaining only positive examples
  - Remove positive examples explained by rule
  - Iterate

Training examples		Background knowledge	
$daughter(mary, ann).$	$\oplus$	$parent(ann, mary).$	$female(ann).$
$daughter(eve, tom).$	$\oplus$	$parent(ann, tom).$	$female(mary).$
$daughter(tom, ann).$	$\ominus$	$parent(tom, eve).$	$female(eve).$
$daughter(eve, ann).$	$\ominus$	$parent(tom, ian).$	

How to make a good “guess”

- Clause subsumption: Generalize
  - More general clause (daughter(mary,Y) subsumes daughter(mary,ann)
- Start with general hypotheses and move to more specific



Issues

- Search space – efficiency
- Noisy data
  - positive examples labeled as negative
  - Missing data (e.g., a daughter with no parents in the database)
- What else might we want to learn?

WARMR: Multi-relational association rules

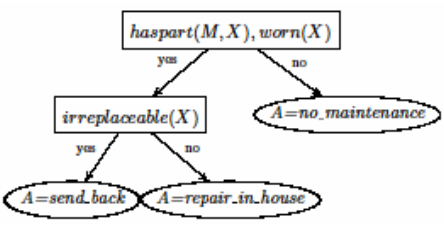
Algorithm WARMR($\mathbf{r}$, $\mathcal{L}$, *key*, *minfreq*; Q)
Input: Database $\mathbf{r}$; Declarative language bias $\mathcal{L}$ and *key*; threshold *minfreq*;
Output: All queries $Q \in \mathcal{L}$ with frequency $\geq$ *minfreq*

1. Initialize level $d := 1$
2. Initialize the set of candidate queries $Q_1 := \{?\text{- } key\}$
3. Initialize the set of (in)frequent queries $\mathcal{F} := \emptyset$; $\mathcal{I} := \emptyset$
4. While Q_d not empty
5. Find frequency of all queries $Q \in Q_d$
6. Move those with frequency below *minfreq* to $\mathcal{I}$
7. Update $\mathcal{F} := \mathcal{F} \cup Q_d$
8. Compute new candidates:
   Q_{d+1} = WARMRgen($\mathcal{L}$; $\mathcal{I}$; $\mathcal{F}$; Q_d))
9. Increment d
10. Return $\mathcal{F}$

Function WARMRgen($\mathcal{L}$; $\mathcal{I}$; $\mathcal{F}$; Q_d);

1. Initialize $Q_{d+1} := \emptyset$
2. For each $Q_j \in Q_d$, and for each refinement $Q'_j \in \mathcal{L}$ of Q_j:
   Add Q'_j to Q_{d+1}, unless:
   (i) Q'_j is more specific than some query $\in \mathcal{I}$, or
   (ii) Q'_j is equivalent to some query $\in Q_{d+1} \cup \mathcal{F}$
3. Return Q_{d+1}

Multi-Relational Decision Trees



$maintenance(M,A) \leftarrow haspart(M,X), worn(X),$
$\quad irreplaceable(X)\ !,\ A = send\_back$
$maintenance(M,A) \leftarrow haspart(M,X), worn(X),\ !,$
$\quad A = repair\_in\_house$
$maintenance(M,A) \leftarrow A = no\_maintenance$

```
procedure DIVIDEANDCONQUER(TestsOnYesBranchesSofar, DeclarativeBias, Examples)
if TERMINATIONCONDITION(Examples)
then
    NewLeaf = CREATENEWLEAF(Examples)
    return NewLeaf
else
    PossibleTestsNow = GENERATETESTS(TestsOnYesBranchesSofar, DeclarativeBias)
    BestTest = FINDBESTTEST(PossibleTestsNow, Examples)
    (Split1, Split2) = SPLITEXAMPLES(Examples, TestsOnYesBranchesSofar, BestTest)
    LeftSubtree = DIVIDEANDCONQUER(TestsOnYesBranchesSofar ∧ BestTest, Split1)
    RightSubtree = DIVIDEANDCONQUER(TestsOnYesBranchesSofar, Split2)
    return [BestTest, LeftSubtree, RightSubtree]
```