Machine Learning and Data Mining: Challenges and Opportunities for CP

Luc De Raedt and Siegfried Nijssen
Dagstuhl Workshop

CP meets DM/ML
WARNING!

TUTORIAL IS INCOMPLETE WRT STATE OF ART

WE PRESENT A FLAVOR OF TECHNIQUES THAT WE FEEL ARE USEFUL

LOGIC-BASED
Questions

1. Can CP problems and CP solvers help to formulate and solve ML / DM problems?

2. Can ML and DM help to formulate and solve constraint satisfaction problems?

We shall argue that the answer to both questions is YES. At the same time, we shall introduce some ML/DM techniques as well as some challenges and opportunities.
The CP perspective

Formulating the model is a knowledge acquisition task.

Improving the performance of solvers is speed-up learning.

Machine learning may help as shown by several initial works.
The ML/DM Perspective

Machine Learning is a (constrained) optimization problem

- learning functions

Data mining is often constraint satisfaction

- “Constraint based mining”

Still ML/DM do not really use CP ...
Constraint-Based Mining

Numerous constraints have been used
Numerous systems have been developed

And yet,

- new constraints often require new implementations
- very hard to combine different constraints
Constraint Programming

Exists since about 20 years

A general and generic methodology for dealing with constraints across different domains

Efficient, extendable general-purpose systems exist, and key principles have been identified

Surprisingly CP has not been used for data mining?

CP systems often more elegant, more flexible and more efficient than special purpose systems

Also true for Data Mining?
Overview

How CP can be used in ML / DM (Siegfried)

- introduction to constraint-based mining
- introduction to constraint-clustering
- challenges

How ML might help CP (Luc)

- learning the model from data
- introduction to some ML techniques
How CP can help DM
Constraints in Data Mining

- Pattern Mining
- Decision Trees
- Clustering
Pattern Mining

• Basic setting: frequent itemset mining
  - Data miner's solution
  - Constraint programming solution

• Extensions
  • Constraint-based mining
    - Common constraints
    - Constraint programming solution
  • Other types of data
  • Pattern set mining
Frequent Itemset Mining

- Market basket data

<table>
<thead>
<tr>
<th></th>
<th>Pampers</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
</tr>
<tr>
<td>2</td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
</tr>
<tr>
<td>4</td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
</tr>
<tr>
<td>5</td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
<td><img src="https://via.placeholder.com/15" alt="Image" /></td>
</tr>
</tbody>
</table>

support( , , ) = 3
Frequent Itemset Mining

• **Given**
  
  • A database with sets of items
  
  • A support threshold

• **Find**
  
  • **ALL** subsets of items \( I \) for which \( \text{support}(I) > \text{threshold} \)
Frequent Itemset Mining

- Gene expression data

\[(\text{Set of Conditions, Set of Genes}) \mid |\text{Set of conditions}| > \text{threshold}\]
### Frequent Itemset Mining

<table>
<thead>
<tr>
<th>Petal length</th>
<th>Petal width</th>
<th>Species</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.9</td>
<td>1.5</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>1.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.5</td>
<td>1.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.9</td>
<td>1.5</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.4</td>
<td>1.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.1</td>
<td>1.5</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>1.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.5</td>
<td>1.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.4</td>
<td>1.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>1.6</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.1</td>
<td>1.9</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.8</td>
<td>1.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.1</td>
<td>1.6</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.6</td>
<td>1.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.3</td>
<td>1.5</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>1.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>7.0</td>
<td>1.4</td>
<td>Iris-versicolor</td>
</tr>
<tr>
<td>6.4</td>
<td>1.5</td>
<td>Iris-versicolor</td>
</tr>
<tr>
<td>6.9</td>
<td>1.5</td>
<td>Iris-versicolor</td>
</tr>
<tr>
<td>5.2</td>
<td>1.3</td>
<td>Iris-versicolor</td>
</tr>
</tbody>
</table>

**Rule**

\[
\text{if} \quad \text{Petal length} \geq 2.0 \quad \text{and} \quad \text{Petal width} \leq 0.5 \\
\text{then Iris-Setosa} \\
\text{else Iris-Versicolor}
\]
Frequent Itemset Mining

• Algorithms
  • Pruning based on “anti-monotonicity”
  • Many different search orders
    – Breadth-first
    – Depth-first
  • Many different data structures
    – How to store lots of data in memory during the search?
Anti-monotonocity
Anti-monotononocity
Anti-monotonocity

- Anti-monotonocity: subsets of frequent itemsets are frequent
Anti-monotonicity
Anti-monotonicity
Anti-monotonicity

\[
\begin{align*}
\{A\} & \quad \{B\} & \quad \{C\} & \quad \{D\} \\
\{A, B\} & \quad \{A, C\} & \quad \{A, D\} & \quad \{B, C\} & \quad \{B, D\} & \quad \{C, D\} \\
\{A, B, C\} & \quad \{A, B, D\} & \quad \{A, C, D\} & \quad \{B, C, D\} \\
\{A, B, C, D\}
\end{align*}
\]
Search: Apriori

\[
\begin{align*}
\{A\} & \quad \{B\} & \quad \{C\} & \quad \{D\} \\
\{A, B\} & \quad \{A, C\} & \quad \{A, D\} & \quad \{B, C\} & \quad \{B, D\} & \quad \{C, D\} \\
\{A, B, C\} & \quad \{A, B, D\} & \quad \{A, C, D\} & \quad \{B, C, D\} \\
\{A, B, C, D\} & 
\end{align*}
\]
Search: Apriori

Candidate Itemset: \{A, B, C, D\}
Search: Apriori

Candidate Itemset

\{A, B, C, D\}

Counted Frequent Itemset

\{A, B, C, D\}
Search: Apriori

Candidate Itemset: \{A, B, C, D\}

Counted Frequent Itemset: \{A, B, C, D\}
Search: Apriori

Candidate Itemset

{A, B, C, D}

Counted Frequent Itemset
Search: Apriori

Candidate Itemset: {A, B, C, D}

Counted Frequent Itemset:

- {A, B, C, D}
Search: Apriori
Search: Apriori

Candidate Itemset

Counted Frequent Itemset
Search: Apriori

{A}  {B}  {C}  {D}

{A, B}  {A, C}  {A, D}  {B, C}  {B, D}  {C, D}

{A, B, C}  {A, B, D}  {A, C, D}  {B, C, D}

Candidate Itemset  {A, B, C, D}

Counted Frequent Itemset
Search: Apriori

Candidate Itemset  {A, B, C, D}

Counted Frequent Itemset
Search: Apriori

• Benefits:
  • Limited number of passes when the database is on disk
  • Maximal pruning before counting
Depth-First Search
Depth-First Search
Depth-First Search
Depth-First Search
Depth-First Search
Depth-First Search
Depth-First Search
Depth-First Search
Depth-First Search

• Benefits:
  • Less candidates at the same time in main memory \(\Rightarrow\) memory can be used for other purposes
  • More efficient in practice
Frequent Itemset Mining in CP

- variables
  \[ [I_1 \ldots I_n], [T_1 \ldots T_m] \]

- domains
  \[ I_x, T_y = \{0, 1\} \]

- constraints
  - support
  \[ \sum_{t} T_t \geq \text{minsup} \]

[De Raedt et al. 2008]
Frequent Itemset Mining in CP

- **variables**
  
  \([I_1 \ldots I_n], [T_1 \ldots T_m]\)

- **domains**
  
  \(I, T = \{0, 1\}\)

- **constraints**
  
  - support

  \[\sum_t T_t \geq \text{minsup}\]

  or reified:

  \[I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{minsup}\]
Frequent Itemset Mining in CP

- **variables**
  \[ [I_1 \ldots I_n], [T_1 \ldots T_m] \]

- **domains**
  \( I_x, T_y = \{0, 1\} \)

- **constraints**
  - **support**
    \[ \sum_t T_t \geq \text{minsup} \]
    or reified:
    \[ I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{minsup} \]
  
  - **coverage**
    \[ T_t = 1 \Leftrightarrow \sum_i I_i (1 - D_{ti}) = 0 \]
Frequent Itemset Mining in CP

- A transaction is covered iff $l \subseteq D_t$
Frequent Itemset Mining in CP

- A transaction is covered iff \( I \subseteq D_t \)

\[ T_t = 1 \iff I \subseteq D_t \]
Frequent Itemset Mining in CP

- A transaction is covered iff $l \subseteq D_t$

\[
T_t = 1 \iff l \subseteq D_t \\
T_t = 1 \iff \forall i \in I : l_i = 1 \rightarrow D_{ti} = 1
\]
Frequent Itemset Mining in CP

- A transaction is covered iff $I \subseteq D_t$

\[
T_t = 1 \iff I \subseteq D_t
\]

\[
T_t = 1 \iff \forall i \in \mathcal{I} : l_i = 1 \rightarrow D_{ti} = 1
\]

\[
T_t = 1 \iff \forall i \in \mathcal{I} : l_i = 0 \lor D_{ti} = 1
\]
Frequent Itemset Mining in CP

- A transaction is covered iff $l \subseteq D_t$

$$T_t = 1 \iff l \subseteq D_t$$

$$T_t = 1 \iff \forall i \in \mathcal{I} : l_i = 1 \rightarrow D_{ti} = 1$$

$$T_t = 1 \iff \forall i \in \mathcal{I} : l_i = 0 \lor D_{ti} = 1$$

$$T_t = 1 \iff \forall i \in \mathcal{I} : l_i = 0 \lor 1 - D_{ti} = 0$$
Frequent Itemset Mining in CP

- A transaction is covered iff $l \subseteq D_t$

\[
T_t = 1 \iff l \subseteq D_t
\]

\[
T_t = 1 \iff \forall i \in I : l_i = 1 \rightarrow D_{ti} = 1
\]

\[
T_t = 1 \iff \forall i \in I : l_i = 0 \lor D_{ti} = 1
\]

\[
T_t = 1 \iff \forall i \in I : l_i = 0 \lor 1 - D_{ti} = 0
\]

\[
T_t = 1 \iff \sum_{i \in I} l_i(1 - D_{ti}) = 0
\]
Frequent Itemset Mining in CP

Model in Minizinc

```minizinc
int: NrI; int: NrT;
array [1..NrT,1..NrI] of bool: TDB;
int: Freq;
array [1..NrI] of var bool: Items;
array [1..NrT] of var bool: Trans;

constraint % coverage
    forall(t in 1..NrT) (Trans[t] <-> sum(i in 1..NrI) (bool2int(TDB[t,i] → Items[i])) <= 0);

constraint % frequency
    forall(i in 1..NrI) (Items[i] -> sum(t in 1..NrT) (bool2int(TDB[t,i] ∧ Trans[t])) >= Freq);

solve satisfy;
```
freq $\geq 2$: $I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{minsup}$

coverage: $T_t = 1 \iff \sum_i I_i (1 - D_{ti}) = 0$

- propagate i2 (freq)
  
  Intuition: infrequent
  
  i2 can never be part of freq. superset
Search

freq >= 2: \[ I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{minsup} \]

coverage: \[ T_t = 1 \iff \sum_i I_i (1 - D_{ti}) = 0 \]

- propagate i2 (freq)
- propagate t1 (coverage)

*Intuition: unavoidable t1 will always be covered*
Search

freq >= 2: \[ I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{minsup} \]

coverage: \[ T_t = 1 \Leftrightarrow \sum_i I_i (1 - D_{ti}) = 0 \]

- propagate i2 (freq)
- propagate t1 (coverage)
Search

freq >= 2: \[ I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{minsup} \]

coverage: \[ T_t = 1 \iff \sum_i I_i (1 - D_{ti}) = 0 \]

- propagate \( i2 \) (freq)
- propagate \( t1 \) (coverage)
- branch \( i1 = 1 \)
- propagate \( t3 \) (coverage)

Intuition: \( t4 \) is missing an item of the itemset
Search

freq $\geq 2$: $I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{minsup}$

coverage: $T_t = 1 \iff \sum_i I_i (1 - D_{ti}) = 0$

- propagate $i2$ (freq)
- propagate $t1$ (coverage)
- branch $i1 = 1$
- propagate $t3$ (coverage)
- propagate $i3$ (freq)

*Intuition: infrequent*
Search

freq \geq 2: \quad I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \minsup

coverage: \quad T_t = 1 \iff \sum_i I_i (1 - D_{ti}) = 0

- propagate i2 (freq)
- propagate t1 (coverage)
- branch i1 = 1
- propagate t3 (coverage)
- propagate i3 (freq)
- propagate t2 (coverage)

Search is similar to depth-first itemset mining algorithms!
Experimental Comparison

Runtime (s)

Minimum support

T10I4D100K (Frequent)

Mushroom (Frequent)
Pattern Explosion
Constraint-based Pattern Mining

• **Given**
  • A database D with sets of items
  • A constraint $\varphi(I,D)$

• **Find**
  • **ALL** subsets of items I for which $\varphi(I,D)$ is true
Inductive Databases

- Inspired by database technology
- Use special purpose logics and solvers to find patterns under constraints

[Imielinski & Mannila, 1996]
Constraint-based Pattern Mining

- Types of constraints
  - Condensed representations
  - Supervised
  - Syntactical constraints
  - ...

Constraints:
Condensed Representations

The full set of patterns can be determined from a subset.
Constraints: Closed Itemsets
(Formal Concepts)

\[
\text{closure ( } \text{Pampers} \text{ ) } = \{ \text{Pampers}, \text{Beer}, \text{ Chips} \}
\]

Maximal rectangles

\[\text{closed}(I, D) \leftrightarrow \text{closure}(I, D) = I\]
(Maximal rectangles)

[Pasquier et al., 1999]
Constraints: Maximal Itemsets
(Borders in Version Spaces)

[Bayardo, 1998]
Constraints: Maximal Itemsets

(Borders in Version Spaces)
Constraints: Maximal Itemsets
(Borders in Version Spaces)
Constraints: Maximal Itemsets
(Borders in Version Spaces)
Constraints: Maximal Itemsets
(Borders in Version Spaces)
Constraints: Condensed Representations

- Maximal frequent itemset $I$: 
  there is no $I' \supset I$ and $I'$ frequent
- Closed itemset $I$: 
  there is no $I' \supset I$ and $\text{support}(I')=\text{support}(I)$
- Free itemset $I$: 
  there is no $I' \subset I$ and $\text{support}(I')=\text{support}(I)$
Search

- Many specialized algorithms developed in data mining (breadth-first, depth-first, ...)

- Can CP be a general framework?
Condensed Representations in CP

- **Frequent Itemset Mining**
  \[ I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{mins}up \]
  \[ T_t = 1 \iff \sum_i I_i (1 - D_{ti}) = 0 \]

- **Maximal Frequent Itemset Mining**
  \[ I_i = 1 \iff \sum_t D_{ti} T_t \geq \text{mins}up \]
  \[ T_t = 1 \iff \sum_i I_i (1 - D_{ti}) = 0 \]

- **Closed Itemset Mining**
  \[ I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{mins}up \]
  \[ I_i = 1 \iff \sum_t T_t (1 - D_{ti}) = 0 \]

- **(δ-)Closed Itemset Mining**
  \[ T_t = 1 \iff \sum_i I_i (1 - D_{ti}) = 0 \]
  \[ I_i = 1 \Rightarrow \sum_t D_{ti} T_t \geq \text{mins}up \]
  \[ I_i = 1 \iff \sum_t T_t (1 - \delta - D_{ti}) \leq 0 \]

Emulates...

- Eclat
- Mafia
- LCM
Itemsets in Supervised Data

Contingency Table

<table>
<thead>
<tr>
<th></th>
<th>OWNs_real_estat</th>
<th>Has_savings</th>
<th>Has_loans</th>
<th>Good_customer</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP: 3</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>FP: 0</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>FN: 1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TN: 3</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>P: 4</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>N: 3</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

[Nijssen et al., 2009]
Itemsets in Supervised Data

- Frequent in negatives
- Infrequent in negatives
- Frequent in positives
- Infrequent in positives
Itemsets in Supervised Data

Contingency Table

<table>
<thead>
<tr>
<th></th>
<th>TP: 3 (=p)</th>
<th>FP: 0 (=n)</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>FN: 1</td>
<td></td>
<td>TN: 3</td>
<td>4</td>
</tr>
<tr>
<td>P: 4</td>
<td></td>
<td>N: 3</td>
<td></td>
</tr>
</tbody>
</table>

Best itemset
Many correlation functions (chi2, fisher, inf. gain) are convex and zero on the diagonal.
Itemsets in Supervised Data

• Again, many different algorithms

• In CP:

\[ I_i = 1 \Rightarrow f \left( \sum_{t \in T^+} D_{ti} T_t, \sum_{t \in T^-} D_{ti} T_t \right) \geq \text{mincorr} \]

\[ T_t = 1 \leftrightarrow \sum_i I_i (1 - D_{ti}) = 0 \]
Itemsets in Supervised Data

General to specific search

- Adding an item will give equal or lower $p$ and $n$
Itemsets in Supervised Data

Key observation: unavoidable transactions
Itemsets in Supervised Data

Key observation: unavoidable transactions
Itemsets in Supervised Data

iterative propagation:
Experimental Comparison

<table>
<thead>
<tr>
<th>Name</th>
<th>corrmine</th>
<th>cimcp</th>
<th>ddpmine</th>
<th>lcm</th>
</tr>
</thead>
<tbody>
<tr>
<td>anneal</td>
<td>0.02</td>
<td>0.22</td>
<td>22.46</td>
<td>7.92</td>
</tr>
<tr>
<td>australian-credit</td>
<td>0.01</td>
<td>0.30</td>
<td>3.40</td>
<td>1.22</td>
</tr>
<tr>
<td>breast-wisconsin</td>
<td>0.03</td>
<td>0.28</td>
<td>96.75</td>
<td>27.49</td>
</tr>
<tr>
<td>diabetes</td>
<td>0.36</td>
<td>2.45</td>
<td>-</td>
<td>697.12</td>
</tr>
<tr>
<td>german-credit</td>
<td>0.07</td>
<td>2.39</td>
<td>-</td>
<td>30.84</td>
</tr>
<tr>
<td>heart-cleveland</td>
<td>0.03</td>
<td>0.19</td>
<td>9.49</td>
<td>2.87</td>
</tr>
<tr>
<td>hypothyroid</td>
<td>0.02</td>
<td>0.71</td>
<td>-</td>
<td>&gt;</td>
</tr>
<tr>
<td>ionosphere</td>
<td>0.24</td>
<td>1.44</td>
<td>-</td>
<td>&gt;</td>
</tr>
<tr>
<td>kr-vs-kp</td>
<td>0.02</td>
<td>0.92</td>
<td>125.60</td>
<td>25.62</td>
</tr>
<tr>
<td>letter</td>
<td>0.65</td>
<td>52.66</td>
<td>-</td>
<td>&gt;</td>
</tr>
<tr>
<td>mushroom</td>
<td>0.03</td>
<td>14.11</td>
<td>0.09</td>
<td>0.03</td>
</tr>
<tr>
<td>pendigits</td>
<td>0.18</td>
<td>3.68</td>
<td>-</td>
<td>&gt;</td>
</tr>
<tr>
<td>primary-tumor</td>
<td>0.01</td>
<td>0.03</td>
<td>0.26</td>
<td>0.08</td>
</tr>
<tr>
<td>segment</td>
<td>0.06</td>
<td>1.45</td>
<td>-</td>
<td>&gt;</td>
</tr>
<tr>
<td>soybean</td>
<td>0.01</td>
<td>0.05</td>
<td>0.05</td>
<td>0.02</td>
</tr>
<tr>
<td>splice-1</td>
<td>0.05</td>
<td>30.41</td>
<td>1.86</td>
<td>0.02</td>
</tr>
<tr>
<td>vehicle</td>
<td>0.07</td>
<td>0.85</td>
<td>-</td>
<td>&gt;</td>
</tr>
<tr>
<td>yeast</td>
<td>0.80</td>
<td>5.67</td>
<td>-</td>
<td>185.28</td>
</tr>
<tr>
<td><strong>avg. when found:</strong></td>
<td><strong>0.15</strong></td>
<td><strong>6.55</strong></td>
<td><strong>28.88⁺</strong></td>
<td><strong>81.54⁺</strong></td>
</tr>
</tbody>
</table>
CP for Pattern Mining

• Promising results
  – More general framework: combining constraints, formalizing new constraints
  – Sometimes more efficient
Challenges

- Other pattern languages
- Pattern set mining
Other Pattern Languages

Graphs

[Inokuchi & Washio, 2003]
Other Pattern Languages

- Graphs [Inokuchi & Washio, 2003]
- Trees [Zaki, 2002]
- Strings [Fischer & Kramer, 2006]
- Sequences [Agrawal & Srikant, 1995]
- Clausal formulas [Dehaspe & De Raedt, 1997]
- ...

See also http://usefulpatterns.org/msop/
Pattern Set Mining

- Constraints on individual patterns do not solve the pattern explosion

Aim: to find a small set of patterns that together are representative / useful
Pattern Set Mining

- **Given**
  - A database D with sets of items
  - A constraint \( \varphi(l,D) \) on patterns \( l \)
  - A constraint \( \Phi(I,D) \) on a set of patterns \( I \)
  - An optimization criterion \( f(I,D) \) on a set of patterns \( I \)

- **Find** the set of patterns \( I \) such that
  - \( f(I,D) \) is maximized
  - Each \( l \) in \( I \) satisfies \( \varphi(l,D) \)
  - \( I \) satisfies \( \Phi(I,D) \)

[De Raedt & Zimmermann, 2007]
Pattern Set Mining

- Co-clustering (aka tiling): “covering the black parts of a matrix with rectangles”
  → Many different formalizations
  (overlap/size/tolerance for errors/...)

TSA 48h
TSA 24h
TSA 12h
TSA  6h
BMP-2 6h
BMP-2 12h
BMP-2 24h
BMP-2 48h
Pattern Set Mining

- Rule-based classification: “predict examples”
  - Many different formalizations (error/ordering of patterns/label in rules/...)

<table>
<thead>
<tr>
<th>Owns real estate</th>
<th>Has savings</th>
<th>Has loans</th>
<th>Good customer</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>-</td>
<td>+</td>
<td>smiley</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>sad</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>+</td>
<td>sad</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>smiley</td>
</tr>
<tr>
<td>-</td>
<td>+</td>
<td>+</td>
<td>sad</td>
</tr>
<tr>
<td>-</td>
<td>+</td>
<td>+</td>
<td>happy</td>
</tr>
<tr>
<td>-</td>
<td>+</td>
<td>+</td>
<td>happy</td>
</tr>
<tr>
<td>-</td>
<td>+</td>
<td>+</td>
<td>happy</td>
</tr>
</tbody>
</table>

- Examples:
  - 30x: + - + smiley
  - 20x: - - - sad
  - 8x: - - - sad
  - 12x: - - + sad
  - 12x: - + - smiley
  - 18x: - + + sad
  - 2x: - + + smiley
Pattern Set Mining

- A general declarative approach?
  - CP systems (Gecode) on declarative formalization of problems with **fixed** pattern set size [Guns et al.]
    (does not scale)
  - SAT solvers (Minisat) on declarative formalization of problems with **fixed** pattern set size [Cremilleux et al.]
    (does not scale)
  - Local search systems (Comet)
    (scales better, but still cumbersome when pattern set size is not fixed in advance) [Guns et al.]
Decision Trees

- Special type of classifier for which more general solvers have been developed

- **Most common approach:** use heuristics to build a tree
  - no constraints
  - no global optimization criterion

- In some cases unsatisfactory
What is a Decision Tree?

- Interpretability
  - Find trees that are small, generalizing, prefer certain tests, ...

Tree learner
What is a Decision Tree?

Tree learner

Public

Private

Privacy preservation

Cost-based constraints

misclassification
Finding Decision Trees: DL8

- Support constraints on leaves → exploit relationship to itemset mining

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Tids</th>
</tr>
</thead>
<tbody>
<tr>
<td>{ C, A }</td>
<td>![Person]</td>
</tr>
<tr>
<td>{ C, ¬A }</td>
<td>![Person1] ![Person2]</td>
</tr>
<tr>
<td>{ ¬C, B }</td>
<td>![Person3]</td>
</tr>
<tr>
<td>{ ¬C, ¬B }</td>
<td>![Person4] ![Person5]</td>
</tr>
</tbody>
</table>

[Nijssen & Fromont, 2007, 2010]
• Decision Trees are hidden in the lattice; if lattices is stored, one can do dynamic programming
Finding Decision Trees: Any Time Algorithm

• Discover the smallest 100% accurate decision tree

• First proposed solution:
  • Greedy algorithm
  • Sample from space of trees to determine expected size after a split (increased sample size → better estimate)
    - Sampling biased by traditional heuristics

• Second proposed solution:
  • Use the first proposed solution to iterative improve subtrees of a tree by using more resources (sample size)

[Esmeir & Markovitch, 2007]
Finding Decision Trees: SAT solvers, CP systems, LP

- Discover the smallest 100% accurate decision tree by means of encoding

- SAT encoding: $O(kn^2m^2 + nk^2 + kn^3)$ space.
  (n = maximum number of nodes in complete tree, k = number of features, m = number of examples)

- CP encoding:
  - Variables for tree nodes
  - Variables for examples in tree nodes
  - Constraints enforcing tree structure (global constraint), binary splits, examples in tree nodes, leafs are pure (logical constraints)
  - Search heuristics, random restarts
  - Improvement by means of LP with $m^2$ variables (on small sets)

[Bessiere, Hebrard, O'Sullivan, 2009]
Clustering

- What is clustering?
- What are constraints in clustering?
- Using solvers for clustering
Clustering

- Fixed number of clusters

[Basu & Davidson 2006, 2011]
Clustering

- Hierarchical clustering
Constraints in Clustering

- Express preferences directly
- Help clustering algorithm finding the “right” solution
- Find alternative clusterings (subspace clustering)
- Semi-supervised learning
Constraints in Clustering

• In hierarchical clustering:
  - Must-link-before constraint
    \textit{a and b must both be in the same cluster before being merged with c}
  - Level specific constraints
    \textit{a and b can only be merged in the top n layers}
Algorithms

- Traditional algorithm + modified distance function either learned, or hand-tuned

- Traditional algorithm + tweaks to enforce hard constraints (i.e. must-link constraints)

- New algorithms few
Hierarchical Clustering

- Traditional algorithm without constraints: 
  *iteratively merge the two clusters that are most near*

- Modified algorithm:
  1. *Encode constraints in Horn clauses*
  2. *Calculate valid merges, i.e. merges that can lead to a valid solution*
  3. *Select most promising merge*
  4. *Go to 2.*

- Valid merges are calculated in polynomial time $O(n^2)$

[Gilpin & Davidson, 2011]
Intermediate Conclusions

• Many problems in data mining can be seen as constraint optimisation problems

• Scalability with respect to data size (both rows and columns) is important

• Most algorithms are not generic algorithms

• There are opportunities to exploit constraint solving technology in data mining
How ML might help CP
Machine Learning for CP

CSP \( (V,D,C,f) \)  \( (f: \text{Optimisation function}) \)

At least three interpretations

- Learning CSP\((V,D,C,f)\) from examples
- Learning to solve for better performance
- “clause” learning etc. (speed-up learning, explanation based learning)
- learning portfolio’s of solvers (meta-learning, preference learning)
Structure Activity Relationship Prediction

[Srinivasan et al. AIJ 96]

Data = Set of Small Graphs
Machine Learning

Given

• a space of possible instances $X$
• an unknown target function $f: X \rightarrow Y$
• a hypothesis space $L$ containing functions $X \rightarrow Y$
• a set of examples $E = \{ (x, f(x)) \mid x \in X \}$
• a loss function $\text{loss}(h, E) \rightarrow \mathbb{R}$

Find $h \in L$ that minimizes $\text{loss}(h, E)$
Classification

Given - Molecular Data Sets

- a space of possible instances $X$ -- Molecular Graphs
- an unknown target function $f: X \rightarrow Y$ -- $\{\text{Active, Inactive}\}$
- a hypothesis space $L$ containing functions $X \rightarrow Y$ -- $L = \{\text{Active iff structural alert } s \text{ covers instance } x \in X | s \in X\}$
- a set of examples $E = \{(x, f(x)) | x \in X\}$
- a loss function $\text{loss}(h, E) \rightarrow \mathbb{R}$ $|\{x \in E | f(x) \neq h(x)\}|$

Find $h \in L$ that minimizes $\text{loss}(h, E)$

If classes = $\{\text{positive, negative}\}$ then this is concept-learning
Regression

Given - Molecular Data Sets

• a space of possible instances $X$ -- Molecular Graphs
• an unknown target function $f: X \rightarrow Y$ -- $\mathbb{R}$
• a hypothesis space $L$ containing functions $X \rightarrow Y$ -- a linear function of some features
• a set of examples $E = \{ (x, f(x)) | x \in X \}$
• a loss function $\text{loss}(h, E) \rightarrow \mathbb{R}$

• **Find** $h \in L$ that minimizes $\text{loss}(h, E)$
Learning Probabilistic Models

**Given**

- a space of possible instances $X$
- an unknown target function $P: X \rightarrow Y \quad Y=[0,1]$  
- a hypothesis space $L$ containing functions $X \rightarrow Y$ (graphical models)
- a set of examples $E = \{ (x, _) \mid x \in X \}$
- a loss function $\text{loss}(h, E) \rightarrow \mathbb{R}$

**Find** $h \in L$ that minimizes $\text{loss}(h, E)$
Boolean Concept-Learning

\[ X = \{(X_1, ..., X_n) \mid X_i = 0 / 1\} \]

\[ Y = \{+,-\} \]

\[ L = \text{boolean formulae} \]

\[ \text{loss}(h, E) = \text{training set error} \]

\[ = \mid \{e \mid e \in E, h(e) \neq f(e)\} \mid / |E| \]

sometimes required to be 0

Simplest setting for learning, compatible with DM part and with CP
### Boolean Concept-Learning

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ex 1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>..</td>
</tr>
<tr>
<td>ex 2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>ex 3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ex 4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

X₂ and X₄
Dimensions

Given

- a space of possible instances $X$
- an unknown target function $f: X \rightarrow Y$
- a hypothesis space $L$ containing functions $X \rightarrow Y$
- a set of examples $E = \{ (x, f(x)) \mid x \in X \}$
- a loss function $\text{loss}(h, E) \rightarrow \mathbb{R}$

Find $h \in L$ that minimizes $\text{loss}(h, E)$

ability to ask questions?
Why boolean concept-learning? constraint networks

<table>
<thead>
<tr>
<th>((V_1, V_2, V_3))</th>
<th>(V_1 &lt; V_2)</th>
<th>(V_1 &gt; V_2)</th>
<th>(V_1 = V_2)</th>
<th>(V_1 &lt; V_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>((1,2,3))</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>((2,3,1))</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>((3,2,1))</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>((1,3,2))</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Propositionalization

CONACQ example [Bessiere et al.]
Monomials

Given

- a space of possible instances $X$
- an unknown target function $f: X \rightarrow Y$
- a hypothesis space $L$ containing functions $X \rightarrow Y$
- a set of examples $E = \{ (x, f(x)) | x \in X \}$
- a loss function $loss(h, E) \rightarrow \mathbb{R}$

Find $h \in L$ that minimizes $loss(h, E)$
Learning monomials

Represent each example by its set of literals

- \{\neg X_1, X_2, \neg X_3, X_4, \neg X_5 \}

Compute the intersection of all positive examples

- intersection = least general generalization

A cautious algorithm

Makes prudent generalizations

[Mitchell, ML textbook 97]
\textbf{k-CNF}

**Given**

- a space of possible instances \( X \)
- an unknown target function \( f: X \to Y \)
- a hypothesis space \( L \) containing functions \( X \to Y \)
- a set of examples \( E = \{ (x, f(x)) \mid x \in X \} \) \textit{pos only}
- a loss function \( \text{loss}(h, E) \to \mathbb{R} \)

**Find** \( h \in L \) that minimizes \( \text{loss}(h, E) \)
Learning $k$-CNF

Naive Algorithm [Valliant CACM 84]

• Let $S$ be the set of all clauses with $k$ literals
• for each positive example $e$
  • for all clauses $s$ in $S$
    • if $e$ does not satisfy $s$ then remove $s$ from $S$

polynomial (for fixed $k$) -- PAC-learnable
Where do the examples come from?

Unknown probability distribution $P$ is assumed on $X$.

The examples in $E$ are drawn at random according to $P$.

The i.i.d. assumption:

- identically and independently distributed

(often does not hold for network / relational data)
Interpretation

Probability Distribution $P$
Classification Revisited

Make predictions about unseen data

\[ \text{loss}_t(h, E) = \frac{|\{e \mid e \in E, h(e) \neq f(e)\}|}{|E|} = \text{training set error} \]

\[ \text{loss}_t(h, X) = P(\{e \mid e \in X, h(e) \neq f(e)\}) = \text{true error} \]
Formal Frameworks Exist

**Probably Approximately Correct** learning (PAC) requires that learner finds with high probability approximately correct hypotheses.

So, \( P(\text{loss}_t(h, X) < \varepsilon) > 1 - \delta \)

Typically combined with complexity requirements

- sample complexity: number of examples
- computational complexity

Valliant proved polynomial PAC-learnability (fixed \( k \))
Learning $(k)$-CNF

Alternative algorithm using Item-Set Mining principles

• minimum frequency = 100%

• clauses are disjunctions; itemsets conjunctions

• monotonicity property :
  • if $e$ satisfies clause $C$ then $e$ also satisfies $C \cup \{ \text{lit} \}$
  • interest in smallest clauses that satisfy 100% freq.

• frequency( $\{ \} \) = 0$, so refinement needed as for item-sets

• find upper border ...
Given

• a space of possible instances $X$
• an unknown target function $f: X \rightarrow Y$
• a hypothesis space $L$ containing functions $X \rightarrow Y$
• a set of examples $E = \{ (x, f(x)) \mid x \in X \}$ pospos and neg
• a loss function $\text{loss}(h, E) \rightarrow \mathbb{R}$ error need not be 0

Find $h \in L$ that minimizes $\text{loss}(h, E)$
Rule learning

Learning from Positives and Negatives

Learn a formula in Disjunctive Normal Form

Rule learning algorithms (machine learning)

Similar issues to pattern set mining (data mining perspective)

Rule learning is often heuristic

Set-covering algorithm

- repeatedly search for one rule (conjunction) that covers many positives and no negative
- discard covered positive examples and repeat

Asking Queries
Active Learning

Provide the learner with the opportunity to ask questions

Let $T$ be the (unknown) target theory

- Does $x$ satisfy $T$? (membership)
- Does $T \models X$? (subset)
- Does $X \models T$? (superset)
- Are $T$ and $X$ logically equivalent? (equivalence)
- ...

The oracle has to provide a counter-example in case the answer is negative [Angluin, MLJournal 88]
How can we use this?

Reconsider learning monomials  (cf. [Mitchell], Conacq [Bessiere et al])

Current hypothesis / conjunction

- \{\neg X_1, X_2, \neg X_3, X_4, \neg X_5 \}

- generate example \{X_1, X_2, \neg X_3, X_4, \neg X_5 \}

- if positive, delete \(X_1\), if negative, keep

- only \(n+1\) questions needed to converge on unique solution (mistake bound)

Very interesting polynomial time algorithms for learning horn sentences [Angluin et al. MLJ 92; Frazier and Pitt, ICML 93] by asking queries
Generalizations

From propositional logic to first order logic

- Inductive Logic Programming

From ILP to Equation Discovery

From hard to soft constraints

- weighted MAX-SAT
- probabilistic models

Learning preferences
Inductive Logic Programming

Instead of learning propositional formulae, learn first order formulae

Usually (definite) clausal logic

Generalizations of many algorithms exist

Rule learning, decision tree learning

Clausal discovery  [De Raedt MLJ 97, De Raedt AIJ 94]

• generalizes k-CNF of Valliant to first order case
• enumeration process as for k-CNF with border ...
Clausal Discovery in ILP

\[
\begin{align*}
\text{train}(\text{utrecht}, 8, 8, \text{denbosch}) & \leftarrow \\
\text{train}(\text{maastricht}, 8, 10, \text{weert}) & \leftarrow \\
\text{train}(\text{utrecht}, 9, 8, \text{denbosch}) & \leftarrow \\
\text{train}(\text{maastricht}, 9, 10, \text{weert}) & \leftarrow \\
\text{train}(\text{utrecht}, 8, 13, \text{eindhoven}) & \leftarrow \\
\text{train}(\text{utrecht}, 8, 43, \text{eindhoven}) & \leftarrow \\
\text{train}(\text{utrecht}, 9, 13, \text{eindhoven}) & \leftarrow \\
\text{train}(\text{utrecht}, 9, 43, \text{eindhoven}) & \leftarrow \\
\text{train}(\text{tilburg}, 8, 10, \text{tilburg}) & \leftarrow \\
\text{train}(\text{utrecht}, 8, 25, \text{denbosch}) & \leftarrow \\
\text{train}(\text{tilburg}, 9, 10, \text{tilburg}) & \leftarrow \\
\text{train}(\text{utrecht}, 9, 25, \text{denbosch}) & \leftarrow \\
\text{train}(\text{tilburg}, 8, 17, \text{eindhoven}) & \leftarrow \\
\text{train}(\text{tilburg}, 8, 47, \text{eindhoven}) & \leftarrow \\
\text{train}(\text{tilburg}, 9, 17, \text{eindhoven}) & \leftarrow \\
\text{train}(\text{tilburg}, 9, 47, \text{eindhoven}) & \leftarrow \\
\end{align*}
\]

From1 = From2 \leftarrow \text{train}(\text{From1}, \text{Hour1}, \text{Min}, \text{To}), \text{train}(\text{From2}, \text{Hour2}, \text{Min}, \text{To})

Inducing constraints that hold in data points here functional dependencies

[De Raedt 97 MLJ, Flach AIComm 99, Abdennaher CP 00, Lopez et al ICTAI 10, ...]
Equation Discovery

Instead of learning clauses, learn equations [Dzeroski and Todorovski, Langley and Bridewell].

As Valiant’s algorithm

• generate and test candidate equations, e.g., $ax + byz = c$
• fit parameters using regression
• possibly compute values for additional variables (partial derivatives w.r.t. time, etc.)
• include a grammar to specify “legal equations” (bias)
Table 1
Variables used in the NPPc portion of the CASA model

\[ NPPc = \max(0, E \cdot IPAR) \]
\[ E = 0.312 \cdot T1^{1.36} \cdot T2^{0.728} \cdot W^0 \]
\[ T1 = 3.65 - 0.992 \cdot topt + 0.137 \cdot topt^2 - 0.00679 \cdot topt^3 + 0.000111 \cdot topt^4 \]
\[ T2 = 0.818/((1 + \exp(0.0521 \cdot (TDIFF - 10))) \cdot (1 + \exp(0 \cdot (-TDIFF - 10)))) \]
\[ TDIFF = topt - tempc \]
\[ W = 0.5 + 0.5 \cdot eet/PET \]
\[ PET = 1.6 \cdot (10 \cdot \max(tempe,0)/ahi)^A \cdot \text{pet_tw_m} \]
\[ A = 0.000000675 \cdot ahi^3 - 0.0000771 \cdot ahi^2 + 0.01792 \cdot ahi + 0.49239 \]
\[ IPAR = FPAR_FAS \cdot \text{monthly_solar} \cdot SOL_CONV \cdot 0.5 \]
\[ FPAR_FAS = \min((SR_FAS - 1.08)/srdiff, 0.95) \]
\[ SR_FAS = (1 + fas_nedvi/750)/(1 - fas_nedvi/750) \]
\[ SOL_CONV = 0.0864 \cdot \text{days_per_month} \]

Using equation discovery to revise an Earth ecosystem model of the carbon net production

Ljupčo Todorovski a,*, Sašo Džeroski a, Pat Langley b, Christopher Potter c
Learning Soft Constraints

Let us look at weighted MAX-SAT problems

Quite popular today in Statistical Relational Learning

- combining first order logic, machine learning and uncertainty
- One example is Markov Logic, many others exist
Factors and Logic

- Propositional atoms are binary (0-1) variables.
- A joint instantiation of all atoms/variables satisfying a propositional formula is a model of that formula.
- If $A$ and $B$ are the only propositions in our language then $A, \neg A \lor B$ has only one model.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 *</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Slide James Cussens
Generalizing Propositional Logic

- Allow arbitrary non-negative values in the factors.
- Allow variables to have more than 2 values.

<table>
<thead>
<tr>
<th>A</th>
<th>A</th>
<th>B</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>*</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Dividing by a normalising constant $Z$ defines a probability distribution over full joint instantiations (when $Z > 0$). Here $Z = 20 + 20 + 0 + 42 = 82$.  

Slide James Cussens
Weighted Clauses

\( \infty : A \text{ and } 2 : \neg A \lor B \)

<table>
<thead>
<tr>
<th>A</th>
<th></th>
<th>A</th>
<th>B</th>
<th></th>
<th>A</th>
<th>B</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>*</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>=</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>exp(-2)</td>
<td>1</td>
<td>0</td>
<td>exp(-2)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Finding the most probable instantiation (highest weighted model) is the weighted MAX-SAT problem.

e^{-w} \text{ where } w = \text{weight of clause if clause not satisfied; weight} = 0 \text{ otherwise}
weighted MAX-SAT

Markov Logic uses weighted (first order logic) clauses to represent a Markov Network

Interesting inference and learning problems

- Compute $P(X|Y)$ ... (CP-techniques can help, weighted model counting)
- Compute most likely state (MAX-SAT)
- Learn parameters (weights of clauses)
  - e.g., using gradient descent on likelihood
- Learn structure and parameters

[Domingos et al], related to [Rossi, Sperduti KR, JETAI etc]
Learning Probabilistic Models

Given

- a space of possible instances \( X \)
- an unknown target function \( P: X \rightarrow Y \) \( Y=[0,1] \)
- a hypothesis space \( L \) containing functions \( X \rightarrow Y \) (graphical models)
- a set of examples \( E = \{ (x, _) \mid x \in X \} \)
- a loss function \( \text{loss}(h,E) \rightarrow \mathbb{R} \)

Find \( h \in L \) that minimizes \( \text{loss}(h,E) \)
## Parameter Estimation

**incomplete data set**

States of some random variables are missing. E.g. medical diagnosis

<table>
<thead>
<tr>
<th></th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
<th>A6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>true</strong></td>
<td><strong>true</strong></td>
<td>?</td>
<td><strong>true</strong></td>
<td>false</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td><strong>?</strong></td>
<td><strong>true</strong></td>
<td>?</td>
<td>?</td>
<td><strong>false</strong></td>
<td>false</td>
<td></td>
</tr>
<tr>
<td><strong>...</strong></td>
<td><strong>...</strong></td>
<td><strong>...</strong></td>
<td><strong>...</strong></td>
<td><strong>...</strong></td>
<td><strong>...</strong></td>
<td></td>
</tr>
<tr>
<td><strong>true</strong></td>
<td>false</td>
<td>?</td>
<td><strong>false</strong></td>
<td><strong>true</strong></td>
<td>?</td>
<td></td>
</tr>
</tbody>
</table>
Parameter Estimation

incomplete data set

states of some random variables are missing
E.g. medical diagnosis

<table>
<thead>
<tr>
<th>A1</th>
<th>A2</th>
<th>A3 (hidden/latent)</th>
<th>A4</th>
<th>A5</th>
<th>A6</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>?</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>?</td>
<td>true</td>
<td>?</td>
<td>?</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>?</td>
<td>false</td>
<td>true</td>
<td>?</td>
</tr>
</tbody>
</table>

missing value
Preference learning

Problem with previous approach

• hard to sample examples from probability distribution in CP context; or to give examples with target probability

A hot topic today in ML, many variations exist, cf. [Furnkranz and Eykemuller, 10, book & tutorial – videolectures]

Two main settings

• learning object preferences (model acquisition)
• learning label preferences (portfolio’s)
Object Preferences

Given

- a space of possible instances $X$
- an unknown ranking function $r(.)$, given $O \subseteq X$, rank instances in $O$
- a hypothesis space $L$ containing ranking functions
- a set of examples $E = \{ (x > y) \mid x, y \in X \}$
- a loss function $\text{loss}(h, E) \rightarrow \mathbb{R}$

Find $h \in L$ that minimizes $\text{loss}(h, E)$
Possible approaches

Explicit relation learning

- Learn a relation $Q(x,y)$ from examples $x < y$
- Determine $r(O)$ as the ordering that is maximally consistent with $Q$

Learn latent utility function

- an unknown utility function $f: X \rightarrow \mathbb{R}$
- examples only impose constraints on $f$
- values of $f$ not known
Label Preferences

Given

- a space of possible instances $X$
- a set of labels $Y = \{Y_1, \ldots, Y_n\}$
- an unknown target function $f(x) = \text{permutation of } Y$
- a set of examples $E = \{ (x , \{ Y_i > Y_j \}) \}$
- a loss function $\text{loss}(h,E) \rightarrow \mathbb{R}$

Find $h \in L$ that minimizes $\text{loss}(h,E)$
Possible approaches

Learn set of relations for each $Y_i > Y_j$

Learn latent utility function for each label $Y_i$

An unknown utility function $f_i : X \rightarrow \mathbb{R}$

• examples only impose constraints on $f_i$
• values of $f$ not known
Summary

The learning of CSPs is possible, so let’s do it

Many settings exist

• data, hypothesis language, active, soft constraints, preference learning, etc

Still we did not touch upon

• bayesian and statistical learning methods

One interesting approach that learns MAX-SAT and MAX-SMT by asking preference questions and using statistical learning techniques

    . Campigotto, A. Passerini and R. Battiti, Lion 10 workshop

Further reading -- Encyclopedia of Machine Learning
• Campigotto, A. Passerini and R. Battiti, Lion 10 workshop
• Alessandro Biso, Francesca Rossi, Alessandro Sperduti: Experimental Results on Learning Soft Constraints. KR 2000: 435-444 and later papers
• Luc De Raedt, Logical and Relational Learning, Springer, 2008.
References [Part II]

• Arnaud Lallouet, Matthieu Lopez, Lionel Martin, Christel Vrain: On Learning Constraint Problems. ICTAI (1) 2010: 45-52
References [Part III]

- Rakesh Agrawal, Ramakrishnan Srikant: Mining Sequential Patterns. ICDE 1995: 3-14
- Roberto J. Bayardo Jr.: Efficiently Mining Long Patterns from Databases. SIGMOD Conference 1998: 85-93
- Luc Dehaspe, Luc De Raedt: Mining Association Rules in Multiple Relations. ILP 1997: 125-132
- Luc De Raedt, Albrecht Zimmermann: Constraint-Based Pattern Set Mining. SDM 2007
- Luc De Raedt, Tias Guns, Siegfried Nijssen: Constraint programming for itemset mining. KDD 2008: 204-212
References [Part IV]

- Johannes Fischer, Volker Heun, Stefan Kramer: Optimal String Mining Under Frequency Constraints. PKDD 2006: 139-150
- Sean Gilpin, Ian Davidson: Incorporating SAT solvers into hierarchical clustering algorithms: an efficient and flexible approach. KDD 2011: 1136-1144
- Siegfried Nijssen, Élisa Fromont: Mining optimal decision trees from itemset lattices. KDD 2007: 530-539
- Siegfried Nijssen, Tias Guns, Luc De Raedt: Correlated itemset mining in ROC space: a constraint programming approach. KDD 2009: 647-656
- Nicolas Pasquier, Yves Bastide, Rafik Taouil, Lotfi Lakhal: Discovering Frequent Closed Itemsets for Association Rules. ICDT 1999: 398-416
- Mohammed Javeed Zaki: Efficiently mining frequent trees in a forest. KDD 2002: 71-80
Thank you