Analysis of Patterns

Introduction to Pattern Discovery

Florent Nicart

University of Rouen

September 29th, 2009
Introduction

1. Why Pattern Discovery?
2. Data sets
3. Pattern Matching vs Pattern Discovery

2. Association Patterns

1. Frequent Set Mining
2. Association Rules Mining

3. Sequential Patterns

1. Sequences
2. Languages
Why Pattern Discovery?

- Increase of data storage capacity,
- Increase of computational power,
- Inter-disciplinary techniques,
- Objective: to understand the underlying process generating the dataset.
Introduction to Pattern Discovery

Cagliari’09

Introduction

Why Pattern Discovery?

Data sets

Pattern Matching vs Pattern Discovery

Association Patterns

Frequent sets

Association Rules Mining

Sequential Patterns

Sequences

Languages

Availability of huge datasets

Databases are everywhere

- Supermarket transactions,
- credit card records,
- telephone call details,
- weblogs, ISP logs,
- genetic databases.

What is a pattern?

Pattern matching: a collection of similar values

Pattern discovery: a particular combination of values
Pattern Matching vs Pattern Discovery

Pattern Matching
- a dataset,
- a pattern,
- a metric and a distance.

Dataset
- Ref. Pattern
- Instance 1
- Instance 2
- Instance 3
- Instance 4
...
Pattern Matching vs Pattern Discovery

Pattern Discovery
- a dataset,
- a metric and a distance
- "a pattern space"
Association Patterns
The pattern:

Finding (subsets) events that occur together [Agrawal93].

Typical application (origin):

Market basket analysis:
- "10% of customers are buying wine and cheese",
- "15% are buying crisps and beer",

and the legendary\(^a\):
- "People buying beer on saturday are very likely to buy nappies" (Wal-Mart).

\(^a\)Beer and Nappies – A Data Mining Urban Legend
http://web.onetel.net.uk/~hibou/Beer and Nappies.html
Introduction to Pattern Discovery (10/59)

Cagliari'09

Introduction

Why Pattern Discovery?

Data sets

Pattern Matching vs Pattern Discovery

Association Patterns

Frequent sets

Association Rules Mining

Sequential Patterns

Sequences

Languages

Example (Amazon associations)

An Introduction to Support Vector Machines and Other Kernel-based Learning Methods (Hardcover)
by Nello Cristianini (Author), John Shawe-Taylor (Author)

RRP: £41.00

Price: £38.95 & this item Delivered FREE in the UK with Super Saver Delivery. See details and conditions

You Save: £2.05 (5%)

In stock.

Dispatched from and sold by Amazon.co.uk. Gift-wrap available.

Want guaranteed delivery by Tuesday, August 25? Order it in the next 3 hours and 55 minutes, and choose Express delivery at checkout. See Details

25 new from £32.80 9 used from £29.52

Price For All Three: £141.54

Add all three to Basket

Show availability and shipping details

Frequently Bought Together

This item: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods by Nello Cristianini

Kernel Methods for Pattern Analysis by John Shawe-Taylor

Pattern Recognition and Machine Learning (Information Science and Statistics) (Information Science and Statistics) by Christopher M. Bishop
Other typical applications:

- Financial-services/telecommunications companies: to which combination of services customers most often subscribe,
- "Quality-assurance: which combinations of components are most likely to fail at the same time,
- Web log analysis,
- Finding Association Rules,
- Classification, ...
Frequent Set Mining

Defining the problem

The input data

- \( \mathcal{I} = \{i_1, i_2, \ldots, i_n\} \), set of items,
- itemset \( S \) : any subset\(^a\) of \( \mathcal{I} \),
- transaction : \( t = \langle tid, S_{tid} \rangle \), \( tid \in \mathbb{N} \) and \( S_{tid} \subseteq \mathcal{I} \)
- \( \mathcal{T} = \{t_1, t_2, \ldots, t_m\} \), set of transactions,

\(^a\)Usually excluding the empty set

Common itemsets

\( t \cap t' = S \cap S' \) with \( t = \langle tid, S \rangle \) and \( t' = \langle tid', S' \rangle \)

\( \text{common}(t, t') = 2^{t \cap t'} \setminus \emptyset \)
### Example (Transactions and itemsets)

<table>
<thead>
<tr>
<th>basket</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
<th>$P_5$</th>
<th>$I_{tid}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>${P_1}$</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>${P_1, P_2, P_3, P_4}$</td>
</tr>
<tr>
<td>$t_3$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>${P_1, P_3, P_5}$</td>
</tr>
<tr>
<td>$t_4$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>${P_3}$</td>
</tr>
<tr>
<td>$t_5$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>${P_2, P_3, P_4}$</td>
</tr>
<tr>
<td>$t_6$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>${P_1, P_2, P_3}$</td>
</tr>
<tr>
<td>$t_7$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>${P_1, P_3, P_4}$</td>
</tr>
<tr>
<td>$t_8$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>${P_2, P_3, P_5}$</td>
</tr>
<tr>
<td>$t_9$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>${P_1, P_4}$</td>
</tr>
<tr>
<td>$t_{10}$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>${P_2, P_3, P_5}$</td>
</tr>
</tbody>
</table>

$common(t_2, t_3) = \{\{P_1\}, \{P_3\}, \{P_1, P_3\}\}$
Frequent Set Mining
Defining the problem

Frequent Itemset Mining (FIM)
- Find the most frequent itemsets that occur amongst all the transactions of $\mathcal{T}$.
- The pattern space is $2^{|\mathcal{I}|}$

Complexity
- Size of the space of patterns : $2^{|\mathcal{I}|}$

|       | #items sold  | $2^{|\mathcal{I}|}$       |
|-------|--------------|---------------------------|
| example |             |                           |
| Small shops | 5000       | $1,4 \times 10^{1505}$   |
| ebay.com  | 30.147.410  | $5 \times 10^{9075274}$ |
| amazon(+third) | +5.000.000 | $9,5 \times 10^{1505149}$ |
| ebay.com(cat) | 20.000   | $4 \times 10^{6020}$     |
Definition (Support Count of an itemset)

The *Support count* (or simply count) of an itemset is the number of transactions where it occurs.

\[ \text{Count}(S) = |\{ < tid, S_{tid} > | S \subseteq S_{tid} \}| \]

Definition (Support of an itemset)

The *Support* of an itemset is the proportion of transactions where it occurs. \[ \text{Support}(S) = \text{Count}(S)/n \]

Definition (Minimal support)

\( \text{minsup} \) is the minimal support of itemset to be considered.

Definition (Supported itemsets)

S is a supported itemset iff \( \text{Support}(S) \geq \text{minsup} \).
**Frequent Set Mining**

**Definitions**

**Example (Supported itemsets)**

<table>
<thead>
<tr>
<th>basket</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
<th>$P_5$</th>
<th>$l_{tid}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>{} $P_1$</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>{} $P_1, P_2, P_3, P_4$</td>
</tr>
<tr>
<td>$t_3$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>{} $P_1, P_3, P_5$</td>
</tr>
<tr>
<td>$t_4$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>{} $P_3$</td>
</tr>
<tr>
<td>$t_5$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>{} $P_2, P_3, P_4$</td>
</tr>
<tr>
<td>$t_6$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>{} $P_1, P_2, P_3$</td>
</tr>
<tr>
<td>$t_7$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>{} $P_1, P_3, P_4$</td>
</tr>
<tr>
<td>$t_8$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>{} $P_2, P_3, P_5$</td>
</tr>
<tr>
<td>$t_9$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>{} $P_1, P_4$</td>
</tr>
<tr>
<td>$t_{10}$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>{} $P_2, P_3, P_5$</td>
</tr>
</tbody>
</table>

\[ \text{Support}(\{P_1, P_3\}) = \frac{|\{t_2, t_3, t_6, t_7\}|}{10} = 0.4 \]

\[ \text{Support}(\{P_2, P_4\}) = \frac{|\{t_2, t_5\}|}{10} = 0.2 \]

with \( \text{minsup} = 0.25 \), \( \{P_1, P_3\} \) is supported, \( \{P_2, P_4\} \) is not.
Theorem (Downward closure property/Support monotonicity.)

\[ S \subseteq S' \iff \text{Support}(S) \geq \text{Support}(S') \]

By corollary, if \( L_k \) is the set of supported sets with cardinality \( k \):

**Theorem**

\[ L_k = \emptyset \Rightarrow \forall i > k, L_i = \emptyset \]
The Apriori algorithm

\[ L_1 \leftarrow \text{supported itemsets of cardinality one} \]
\[ k \leftarrow 2 \]
\[ \text{while } (L_{k-1} = \emptyset) \{ \]
\[ C_k \leftarrow \text{create\_candidates}(L_{k-1}) \]
\[ L_k \leftarrow \text{prune}(C_k) \]
\[ k \leftarrow k + 1 \]
\[ \} \]
\[ \text{return } L_1 \cup L_2 \cup \ldots L_k \]
Generating $C_k$ from $L_{k-1}$

**Join Step**
Compare each member of $L_{k-1}$, say A, with every other member, say B, in turn. If the first $k-2$ items in A and B (i.e. all but the rightmost elements of the two itemsets) are identical, place set $A \cup B$ into $C_k$.

**Prune Step**
For each member $c$ of $C_k$ in turn {
    
    Examine all subsets of $c$ with $k-1$ elements
    
    Delete $c$ from $C_k$ if any of the subsets is not a member of $L_{k-1}$
}


Frequent Set Mining
The Apriori algorithm: an example

Example (Joint step for $C_5'$)

$L_4 = \{\{p, q, r, s\}, \{p, q, r, t\}, \{p, q, r, z\}, \{p, q, s, z\},$
$\{p, r, s, z\}, \{q, r, s, z\}, \{r, s, w, x\}, \{r, s, w, z\}, \{r, t, v, x\},$
$\{r, t, v, z\}, \{r, t, x, z\}, \{r, v, x, y\}, \{r, v, x, z\}, \{r, v, y, z\},$
$\{r, x, y, z\}, \{t, v, x, z\}, \{v, x, y, z\}\}$

<table>
<thead>
<tr>
<th>$l_1$</th>
<th>$l_2$</th>
<th>Contribution to $C_5'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>{p, q, r, s}</td>
<td>{p, q, r, t}</td>
<td>{p, q, r, s, t}</td>
</tr>
<tr>
<td>{p, q, r, s}</td>
<td>{p, q, r, z}</td>
<td>{p, q, r, s, z}</td>
</tr>
<tr>
<td>{p, q, r, t}</td>
<td>{p, q, r, z}</td>
<td>{p, q, r, t, z}</td>
</tr>
<tr>
<td>{r, s, w, x}</td>
<td>{r, s, w, z}</td>
<td>{r, s, w, x, z}</td>
</tr>
<tr>
<td>{r, t, v, x}</td>
<td>{r, t, v, z}</td>
<td>{r, t, v, x, z}</td>
</tr>
<tr>
<td>{r, v, x, y}</td>
<td>{r, v, x, z}</td>
<td>{r, v, x, y, z}</td>
</tr>
</tbody>
</table>

$\rightarrow C'_5 = \{\{p, q, r, s, t\}, \{p, q, r, s, z\}, \{p, q, r, t, z\}, \{r, s, w, x, z\},$
$\{r, t, v, x, z\}, \{r, v, x, y, z\}\}$
Frequent Set Mining
The Apriori algorithm: an example

Example (Pruning step for \( C_5 \))

\[
L_4 = \{ \{ p, q, r, s \}, \{ p, q, r, t \}, \{ p, q, r, z \}, \{ p, q, s, z \}, \{ p, r, s, z \}, \{ q, r, s, z \}, \{ r, s, w, x \}, \{ r, s, w, z \}, \{ r, t, v, x \}, \{ r, t, v, z \}, \{ r, t, x, z \}, \{ r, v, x, y \}, \{ r, v, x, z \}, \{ r, v, y, z \}, \{ r, x, y, z \}, \{ t, v, x, z \}, \{ v, x, y, z \} \}
\]

\[
C'_5 = \{ \{ p, q, r, s, t \}, \{ p, q, r, s, z \}, \{ p, q, r, t, z \}, \{ r, s, w, x, z \}, \{ r, t, v, x, z \}, \{ r, v, x, y, z \} \}
\]

<table>
<thead>
<tr>
<th>Itemset in ( C'_5 )</th>
<th>Subsets all in ( L_4 )?</th>
</tr>
</thead>
<tbody>
<tr>
<td>{ p, q, r, s, t }</td>
<td>No: { p, q, s, t } \not\in L_4</td>
</tr>
<tr>
<td>{ p, q, r, s, z }</td>
<td>Yes</td>
</tr>
<tr>
<td>{ p, q, r, t, z }</td>
<td>No: { p, q, t, z } \not\in L_4</td>
</tr>
<tr>
<td>{ r, s, w, x, z }</td>
<td>No: { r, s, x, z } \not\in L_4</td>
</tr>
<tr>
<td>{ r, t, v, x, z }</td>
<td>Yes</td>
</tr>
<tr>
<td>{ r, v, x, y, z }</td>
<td>Yes</td>
</tr>
</tbody>
</table>

\[
C_5 = \{ \{ p, q, r, s, z \}, \{ r, t, v, x, z \}, \{ r, v, x, y, z \} \}
\]
We would like to use the itemset found to express association rules...
Introduction to Pattern Discovery

Cagliari'09

Introduction

Why Pattern Discovery?
Data sets
Pattern Matching vs Pattern Discovery

Association Patterns
Frequent sets
Association Rules Mining

Sequential Patterns
Sequences
Languages

Association Rules Mining

Introduction

Definition (Rule)

$L \rightarrow R$ with $L$ (antecedant/condition) and $R$ (consequence) being predicates.

- Reads $L$ implies $R$,
- ex: "If it rains then the ground will be wet."
- \{ab\} $\rightarrow$ \{cd\} means \{a, b\} $\subseteq$ $S$ $\rightarrow$ \{c, d\} $\subseteq$ $S$, $\forall S \in T$

Definition (Association Rule)

$L \rightarrow R$ with $L \neq \emptyset$ and $R \neq \emptyset$ being disjoint itemsets.
Introduction to Pattern Discovery

Why Pattern Discovery?

Data sets

Pattern Matching vs Pattern Discovery

Association Patterns

Frequent sets

Association Rules Mining

Sequential Patterns

Sequences

Languages

Association Rules Mining

Introduction

- Probabilistic AR: "When beer and crisps are bought together, cheese is bought in 45% of cases",
- \( L \rightarrow R : L \cup R \) is an itemset:

<table>
<thead>
<tr>
<th>Definition (Support of an Association Rule)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{Support}(L \rightarrow R) = \text{Support}(L \cup R) = \frac{\text{Count}(L \cup R)}{n} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Definition (Confidence of an Association Rule)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{Confidence}(L \rightarrow R) = \frac{\text{Support}(L \cup R)}{\text{Support}(L)} = \frac{\text{Count}(L \cup R)}{\text{Count}(L)} )</td>
</tr>
</tbody>
</table>
Only supported rules: \( \text{Support}(L \rightarrow R) \geq \text{minsup} \),
given \(|L \cup R| = k\), then number of rules that can be generated:

\[
\sum_{i=1}^{k-1} C_i^k = 2^k - 2
\]

Filtering rules: \( \text{Confidence}(L \rightarrow R) \geq \text{minconf} \)

**Definition (Confident Right-hand side)**

The right-hand side of a rule is said **confident** iff \( \text{Confidence}(L \rightarrow R) \geq \text{minconf} \) and **unconfident** otherwise.
**Association Rules Mining**

**Generating rules**

**Theorem**

\[ \text{Confidence}(L \cup \{x\} \rightarrow R) \geq \text{Confidence}(L \rightarrow R \cup \{x\}) \]

**Corollary**

- Any superset of an unconfident right-hand itemset is unconfident.
- Any (non-empty) subset of a confident right-hand itemset is confident.

- Rules can be generated with an Apriori-like algorithm.
Sequential Patterns

- Temporal sequence data sets:
  - Supermarket customer transactions,
  - Weblogs: user navigation,
  - Alarm logs: fault analysis,

- Text sequence data sets:
  - Genomics,
  - Documents,
  - etc.
Finding sequential patterns in sequences of events:

- Data set: supermarket customer transactions,
- Pattern: customers renting "The Fellowship of the Ring" rent "The Two Towers" and then "The Return of the King"

## Definitions

### Definition (Sequence)

\[ s = \langle s_1 \ldots s_n \rangle \text{ where } s_k \text{ is an itemset} \]

### Definition (Contained sequence)

A sequence \( \langle a_1 \ldots a_n \rangle \) is contained in a sequence \( \langle b_1 \ldots b_m \rangle \) if there exists integers \( i_1 < i_2 < \cdots < i_n \) such that \( a_1 \subseteq b_{i_1}, a_2 \subseteq b_{i_2}, \ldots, a_n \subseteq b_{i_n} \).

### Examples

- \( \langle \{3\}\{4, 5\}\{8\} \rangle \) is contained in \( \langle \{7\}\{3, 8\}\{9\}\{4, 5, 6\}\{8\} \rangle \)
- \( \langle \{3\}\{5\} \rangle \) is not contained in \( \langle \{3, 5\} \rangle \)
Sequences

Examples

<table>
<thead>
<tr>
<th>Customer Id</th>
<th>Transaction Time</th>
<th>Items Bought</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>June 25 ’93</td>
<td>30</td>
</tr>
<tr>
<td>1</td>
<td>June 30 ’93</td>
<td>90</td>
</tr>
<tr>
<td>2</td>
<td>June 10 ’93</td>
<td>10, 20</td>
</tr>
<tr>
<td>2</td>
<td>June 15 ’93</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>June 20 ’93</td>
<td>40, 60, 70</td>
</tr>
<tr>
<td>3</td>
<td>June 25 ’93</td>
<td>30, 50, 70</td>
</tr>
<tr>
<td>4</td>
<td>June 25 ’93</td>
<td>30</td>
</tr>
<tr>
<td>4</td>
<td>June 30 ’93</td>
<td>40, 70</td>
</tr>
<tr>
<td>4</td>
<td>July 25 ’93</td>
<td>90</td>
</tr>
<tr>
<td>5</td>
<td>June 12 ’93</td>
<td>90</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Customer Id</th>
<th>Customer Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>⟨{30}{90}⟩</td>
</tr>
<tr>
<td>2</td>
<td>⟨{10, 20}{30}{40, 60, 70}⟩</td>
</tr>
<tr>
<td>3</td>
<td>⟨{30, 50, 70}{30}{40, 70}{90}⟩</td>
</tr>
<tr>
<td>5</td>
<td>⟨{90}⟩</td>
</tr>
</tbody>
</table>
**Sequences**

**Sequence Support**

**Definition (Sequence Support)**

Let $T_i$ be the set of transactions of customer $i$,

$$Support(s) = |\{i \mid s \subseteq T_i\}|$$

**Definition (Maximal sequence)**

In a set of sequences, a sequence is *maximal* if it is not contained in any other sequence.

**Pattern discovery task:**

Find all maximal sequences (*Sequential Patterns*) in a set of customer transactions having a support greater than $\text{minseqsup}$.
### Sequences

#### Sequence Support

<table>
<thead>
<tr>
<th>Customer Id</th>
<th>Customer Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>⟨{30}{90}⟩</td>
</tr>
<tr>
<td>2</td>
<td>⟨{10, 20}{30}{40, 60, 70}⟩</td>
</tr>
<tr>
<td>3</td>
<td>⟨{30, 50, 70}{30}{40, 70}{90}⟩</td>
</tr>
<tr>
<td>5</td>
<td>⟨{90}⟩</td>
</tr>
</tbody>
</table>

#### Sequential Patterns (\(\text{minseqsup} = 0.25\))

- ⟨{30}{90}⟩
- ⟨{30}{40, 70}⟩

Sequences ⟨{30}⟩, ⟨{40}⟩, ⟨{70}⟩, ⟨{90}⟩, ⟨{30}{40}⟩, ⟨{30}{70}⟩, ⟨{40, 70}⟩ are not maximal.
Algorithm \textit{Agrawal et al}

- Sorting phase: (customer id, transaction time)
- Customer-supported itemset phase: customer count based Apriori
- Transformation phase: transforms the customer transaction to accelerate the containing tests
- Sequence phase: find supported sequences from (customer)supported itemsets (AprioriAll, AprioriSome/DynamicSome)
- Maximal phase: keep only the maximal supported sequences.
Finding *episodes* in sequences of events:

- Data set: set of timed events,
- Pattern: groups of events occurring frequently *close* to each other

---

*Heikki Mannila, Hannu Toivonen and A. Inkeri Verkamo,*
*Discovery of Frequent Episodes in Event Sequences,*
*Data Mining and Knowledge Discovery, 1997.*
Introduction to Pattern Discovery (35/59)

Cagliari'09

Introduction
Why Pattern Discovery?

Data sets
Pattern Matching vs Pattern Discovery

Association Patterns
Frequent sets
Association Rules Mining

Sequential Patterns
Sequences
Languages

Sequences
Event sequences

**Definition (event)**

Let \( E \) be a set of event types, an event is a pair \((A, t)\) where \( A \in E \) and \( t \) is an integer (timestamp).

\[
 s = \langle (E, 31), (D, 32), (F, 33), (A, 35), (B, 37), \ldots, (D, 67) \rangle
\]
Introduction to Pattern Discovery (36/59)

Cagliari'09

Introduction

Why Pattern Discovery?

Data sets

Pattern Matching vs Pattern Discovery

Association Patterns

Frequent sets

Association Rules Mining

Sequential Patterns

Sequences

Languages

Sequences

Episodes

\[ s = \langle (E, 31), (D, 32), (F, 33), (A, 35), (B, 37), \ldots, (D, 67) \rangle \]

**Definition (episode)**

An episode is a partially ordered subset of events.

\[ e = (V, \leq, g) \text{, } g : V \rightarrow E \]
**Sequences**

**Subepisodes**

**Definition (Subepisode)**

An episode \( e = (V', \leq', g') \) is a subepisode of \( e = (V, \leq, g) \) if \( V' \subseteq V, \forall v \in V', g'(v) = g(v), \) and \( \forall v, w \in V', v \leq' w \rightarrow v \leq w \)
Introduction to Pattern Discovery (38/59)

Cagliari’09

Introduction

Why Pattern Discovery?

Data sets

Pattern Matching vs Pattern Discovery

Association Patterns

Frequent sets

Association Rules Mining

Sequential Patterns

Sequences

Languages

Sequences

Mining episodes

Pattern discovery task

Given

- a window size,
- a minimal frequency,
- a class of episodes,

find all the frequent episodes in the class.

Note: frequency of an episode = amount of windows in which it occurs.
Sequences
Algorithm

Algorithm \((\text{Mannila et al})\)

\[
C_1 \leftarrow \{\{e\} \mid e \in E\}
\]
\[
k \leftarrow 1
\]
\[
\text{while } (C_k = \emptyset) \{ \\
    L_k \leftarrow \text{frequent}(C_k, s, \text{minfr}, \text{winsize}) \\
    C_{k+1} \leftarrow \text{generate}(L_k, C) \\
    k \leftarrow k + 1
\}
\]
\[
\text{return } L_1 \cup L_2 \cup \ldots L_k
\]
Languages

Discovering (rational) languages
Languages
What is Gramatical Inference?

- A grammar describes a language/set of words (possibly infinite).
- Is it possible, from a finite sample (or a positive and negative sample) to guess the grammar?
- Grammatical inference: find a description of the generating process.
Introduction to Pattern Discovery (42/59)

Cagliari’09

Introduction

Why Pattern Discovery?

Data sets

Pattern Matching vs Pattern Discovery

Association Patterns

Frequent sets

Association Rules

Mining

Sequential Patterns

Sequences

Languages

Languages

Grammars and their hierarchy

Chomsky hierarchy:

<table>
<thead>
<tr>
<th>Type</th>
<th>Grammar</th>
<th>Language</th>
<th>Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>type-0</td>
<td>Unrestricted</td>
<td>Recurs. enumerable. Context-sensitive</td>
<td>Turing machine</td>
</tr>
<tr>
<td>type-1</td>
<td>Context-sensitive</td>
<td>Context-sensitive Context-sensitive</td>
<td>Linear bounded</td>
</tr>
<tr>
<td>type-2</td>
<td>Context-free</td>
<td>Context-free</td>
<td>Pushdown aut.</td>
</tr>
<tr>
<td>type-3</td>
<td><strong>Regular</strong></td>
<td><strong>Regular</strong></td>
<td><strong>Finite aut.</strong></td>
</tr>
</tbody>
</table>
Languages
Definitions: regular language

Definition (Regular language)
Let $\Sigma$ be a finite set of symbols (ex: $\Sigma = \{a, b, c\}$), a regular language $L \subseteq \Sigma^*$ recursively defined by mean of concatenations, unions and stars.

Example (Regular language)
$L = a \cdot b^* \cdot c = \{ac, abc, abbc, abbbbc, \ldots\}$

Theorem (Kleene)
$Rat = Rec$
**Languages**

Definitions: finite state automaton

**Definition (Finite state automaton)**

Let $\Sigma$ be an alphabet, a finite automaton is a 5-tuple $\mathcal{A} = \langle \Sigma, Q, I, F, E \rangle$ where

- $Q$ is a finite set of states,
- $I \subseteq Q$ is the set of initial states,
- $F \subseteq Q$ is the set of final states,
- $E \subseteq Q \times \Sigma \times Q$ is the set of transitions.

**Example (Finite state automaton)**

![Finite state automaton diagram](https://example.com/automaton.png)
Languages
Definitions: finite state automaton

Example (Finite state automaton)

- Path, accepted path
- accepted word
- Language $L(\mathcal{A})$ recognized by $\mathcal{A}$
Definition (Maximal automaton)

Given a sample $I_+$, the maximal automaton recognizing $I_+$ is the biggest automaton recognizing $I_+$.

Example (Maximal automaton)

$I_+ = \{a, ab, bab\}$:

![Maximal automaton diagram]
Languages
Quotient of an automaton

Definition (Partition of a set)
For any set $S$, a partition $\pi$ is defined by $\pi = \{s \mid s \subseteq S\}$
such that $\forall s, s' \in \pi$, $s \cap s' = \emptyset$,
$\bigcup_{s \in \pi} = S$

Definition (Quotient of an automaton)
$A/\pi = \langle \Sigma, Q', I', F', E' \rangle$ is a quotient of $A = \langle \Sigma, Q, I, F, E \rangle$ iff
$Q' = Q/\pi$
$I = \{B \in Q' \mid \exists q \in Bs.t.q \in I\}$
$F = \{B \in Q' \mid \exists q \in Bs.t.q \in F\}$
$E = \{\langle B, a, B' \rangle \in Q' \times \Sigma \times Q' \mid \exists q \in B, q' \in B', a \in \Sigma$
s.t.$\langle q, a, q' \rangle \in E\}$
Languages

Quotient of an automaton: property

Property

If $\mathcal{A}/\pi$ is the quotient of $\mathcal{A}$ w.r.t. relation $\pi$, then $L(\mathcal{A}) \subseteq L(\mathcal{A}/\pi)$

- any derivation produces a generalisation,
- the set of automata obtained by subsequent derivation forms a lattice.
Languages
Example: partition $\pi_1$

$L(A) = a(aa)^*b + ab(bb)^*$

$\pi_1 = \{\{0, 1\}, \{2\}, \{3, 4\}\}$

$L(A/\pi_1) = a^*(b + ab(bb)^*)$
Languages
Example: partition $\pi_2$

$L(A) = a(aa)^*b + ab(bb)^*$

$\pi_2 = \{\{0, 1, 2\}, \{3, 4\}\}$

$L(A/\pi_2) = a^*b(ba^*b)^*$
Introduction to Pattern Discovery (51/59)

Cagliari'09

Example: partition $\pi_3$

$L(\mathcal{A}) = a(aa)^*b + ab(bb)^*$

$\pi_3 = \{\{0\}, \{1, 3\}, \{2, 4\}\}$

$L(\mathcal{A}/\pi_3) = (aa^*a)^*(a + ab^*)$
Languages
Example partition $\pi_4$

$L(A) = a(aa)^* b + ab(bb)^*$

$\pi_4 = \{\{0, 1, 3, 2, 4\}\}$

$L(A/\pi_4) = (a + b)^*$
Languages

The mining task

- Find an automaton that generalizes, *not too much*, the input sample: characterizable, heuristic methods;
- the space of possibilities is the number of partitions of the set of states of the sample maximal automaton;
- the size of the exploration space is the number of partitions of the maximal automaton coding the sample: $B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_k$

$B_5 = 52, \ldots, B_{10} = 115975, B_{20} \approx 5.10^{13}, B_{30} \approx 8.10^{23}, B_{50} \approx 2.10^{49} \ldots$
Languages
A characterizable method: the $k - RI$ algorithm

Definition ($k$-deterministic automaton)

$A$ is $k$-deterministic if, from any state, there is at most one path for any given word of length $\geq k$.

Example (A 2-deterministic automaton)

Deterministic automaton are 0-deterministic.
Languages
A characterizable method: the $k$ – $RI$ algorithm

**Definition (Reverse automaton)**

The reverse automaton of $\mathcal{A} = \langle \Sigma, Q, I, F, E \rangle$ is defined by $\mathcal{A}^R = \langle \Sigma, Q, F, I, E' \rangle$ with $E' = \{ \langle q, a, q' \rangle | \langle q', a, q \rangle \in E \}$.

**Example (Reverse automaton)**

![Example Diagram](image-url)
Languages
A characterizable method: the $k - RI$ algorithm

**Definition ($k$-reversible automaton)**

An automaton is $k$-reversible if it is deterministic and its reverse automaton is $k$-deterministic.

Dana Angluin [Ang82] gave a characterization of reversible languages and an inference algorithm: $k - RI$.

Dana Angluin.
Inference of Reversible Languages,
Languages

A characterizable method: the \( k - RI \) algorithm

The \( k - RI \) algorithm

\[ k \leftarrow \text{order of the model}, \ I_+ \leftarrow \text{positive sample} \]
\[ A \leftarrow PTA(I_+) \]
\[ \pi \leftarrow Q \]

while \( \neg k\text{-reversible}(A/\pi) \) {

\[ (B_1, B_2) \leftarrow \text{non-reversible}(A/\pi, \pi) \]
\[ \pi \leftarrow \pi \setminus \{B_1, B_2\} \cup \{B_1 \cup B_2\} \]

}\n
return \( A/\pi \)
Languages
A characterizable method: the $k - RI$ algorithm

Example ($k - RI$ run with $l_+ = \{ab, bb, aab, abb\}, k = 1$)

$$L(\mathcal{A}/\pi) = b^+ + a^+ b^+ = a^* b^+$$
Some references to go further ...


WEKA : associations.apriori