Data Mining II
Time Series Analysis

Heiko Paulheim, Robert Meusel
Introduction

• So far, we have only looked at data without a time dimension
  – or simply ignored time dimension

• Many “classic” DM problems have variants that respect time
  – frequent pattern mining → sequential pattern mining
  – classification → predicting sequences of nominals
  – regression → predicting the continuation of a numeric series
Contents

• Sequential Pattern Mining
  – Finding frequent subsequences in set of sequences
  – the GSP algorithm

• Trend analysis
  – Is a time series moving up or down?
  – Simple models and smoothing
  – Identifying seasonal effects

• Forecasting
  – Predicting future developments from the past
  – The windowing technique
Mining Time Series Data in RapidMiner

- Basic methods are covered in standard edition
- Powerful (and complex) series extension available
Sequential Pattern Mining: Application 1

- Web usage mining (navigation analysis)
- Input
  - Server logs
- Patterns
  - typical sequences of pages
- Usage
  - restructuring web sites
Sequential Pattern Mining: Application 2

• Recurring customers
  – Typical book store example:
    • (Twilight) (New Moon) → (Eclipse)

• Recommendation in online stores
• Allows more fine grained suggestions than frequent pattern mining
• Example:
  – mobile phone → charger vs. charger → mobile phone
    • are indistinguishable by frequent pattern mining
  – customers will select a charger after a mobile phone
    • but not the other way around!
    • however, Amazon does not respect sequences...
Sequential Pattern Mining: Application 3

- Using texts as a corpus
  - looking for common sequences of words
  - allows for intelligent suggestions for autocompletion
Sequential Pattern Mining: Application 4

- Chord progressions in music
  - supporting musicians (or even computers) in jam sessions
  - supporting producers in writing top 10 hits :-)

http://www.hooktheory.com/blog/i-analyzed-the-chords-of-1300-popular-songs-for-patterns-this-is-what-i-found/
Sequence Data

- Data Model: transactions containing items

<table>
<thead>
<tr>
<th>Sequence Database</th>
<th>Sequence</th>
<th>Element (Transaction)</th>
<th>Event (Item)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Data</td>
<td>Purchase history of a given customer</td>
<td>A set of items bought by a customer at time t</td>
<td>Books, dairy products, CDs, etc</td>
</tr>
<tr>
<td>Web Server Logs</td>
<td>Browsing activity of a particular Web visitor</td>
<td>A collection of files viewed by a Web visitor after a single mouse click</td>
<td>Home page, index page, contact info, etc</td>
</tr>
<tr>
<td>Sensor Data</td>
<td>History of events generated by a given sensor</td>
<td>Events triggered by a sensor at time t</td>
<td>Types of alarms generated by sensors</td>
</tr>
</tbody>
</table>

Element (Transaction) E1 E2 E3 Event (Item) E2 E3 E4

Sequence: E1 E2 E3 E2 E4
### Sequence Data

#### Sequence Database

<table>
<thead>
<tr>
<th>Object</th>
<th>Timestamp</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>2, 3, 5</td>
</tr>
<tr>
<td>A</td>
<td>20</td>
<td>6, 1</td>
</tr>
<tr>
<td>A</td>
<td>23</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>11</td>
<td>4, 5, 6</td>
</tr>
<tr>
<td>B</td>
<td>17</td>
<td>2</td>
</tr>
<tr>
<td>B</td>
<td>21</td>
<td>7, 8, 1, 2</td>
</tr>
<tr>
<td>B</td>
<td>28</td>
<td>1, 6</td>
</tr>
<tr>
<td>C</td>
<td>14</td>
<td>1, 8, 7</td>
</tr>
</tbody>
</table>

![Timeline Graph]

Object A:

Object B:

Object C:
Formal Definition of a Sequence

- A sequence is an ordered list of elements (transactions)
  \[ s = < e_1, e_2, e_3, \ldots > \]
  - Each element contains a collection of events (items)
    \[ e_i = \{i_1, i_2, \ldots, i_k\} \]
  - Each element is attributed to a specific time or location

- Length of a sequence |s| is given by the number of elements of the sequence.

- A k-sequence is a sequence that contains k events (items).
Further Examples of Sequences

• Web browsing sequence:

\[
< \text{Homepage} \quad \text{Electronics} \quad \text{Digital Cameras} \quad \text{Canon Digital Camera} \quad \text{Shopping Cart} \quad \text{Order Confirmation} \quad \text{Homepage} >
\]

• Sequence of books checked out at a library:

\[
< \text{Fellowship of the Ring} \quad \text{The Two Towers, Return of the King} >
\]

• Sequence of initiating events causing the nuclear accident at 3-mile Island:

\[
< \text{clogged resin} \quad \text{outlet valve closure} \quad \text{loss of feedwater} \quad \text{condenser polisher outlet valve shut} \quad \text{booster pumps stop} \quad \text{main waterpump stops, main turbine stops} \quad \text{reactor pressure increases} >
\]
Formal Definition of a Subsequence

- A sequence $<a_1 \ldots a_n>$ is contained in another sequence $<b_1 \ldots b_m>$ ($m \geq n$) if there exist integers $i_1 < i_2 < \ldots < i_n$ such that $a_1 \subseteq b_{i_1}, a_2 \subseteq b_{i_2}, \ldots, a_n \subseteq b_{i_n}$

<table>
<thead>
<tr>
<th>Data sequence $&lt;b&gt;$</th>
<th>Subsequence $&lt;a&gt;$</th>
<th>Contain?</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt;{2,4} {3,5,6} {8}&gt;$</td>
<td>$&lt;{2} {3,5}&gt;$</td>
<td>Yes</td>
</tr>
<tr>
<td>$&lt;{1,2} {3,4}&gt;$</td>
<td>$&lt;{1} {2}&gt;$</td>
<td>No</td>
</tr>
<tr>
<td>$&lt;{2,4} {2,4} {2,5}&gt;$</td>
<td>$&lt;{2} {4}&gt;$</td>
<td>Yes</td>
</tr>
</tbody>
</table>

- The support of a subsequence $w$ is defined as the fraction of data sequences that contain $w$

- A sequential pattern is a frequent subsequence (i.e., a subsequence whose support is $\geq \text{minsup}$)
Examples of Sequential Patterns

<table>
<thead>
<tr>
<th>Object</th>
<th>Timestamp</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>1,2,4</td>
</tr>
<tr>
<td>A</td>
<td>2</td>
<td>2,3</td>
</tr>
<tr>
<td>A</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1,2</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>2,3,4</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1, 2</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>2,3,4</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>2,4,5</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
<td>3, 4</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
<td>4, 5</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>1, 3</td>
</tr>
<tr>
<td>E</td>
<td>2</td>
<td>2, 4, 5</td>
</tr>
</tbody>
</table>

\[ \text{Minsup} = 50\% \]

Examples of frequent subsequences:

\[ \langle \{1,2\} \rangle \quad s=60\% \]
\[ \langle \{2,3\} \rangle \quad s=60\% \]
\[ \langle \{2,4\} \rangle \quad s=80\% \]
\[ \langle \{3\} \{5\} \rangle \quad s=80\% \]
\[ \langle \{1\} \{2\} \rangle \quad s=80\% \]
\[ \langle \{2\} \{2\} \rangle \quad s=60\% \]
\[ \langle \{1\} \{2,3\} \rangle \quad s=60\% \]
\[ \langle \{2\} \{2,3\} \rangle \quad s=60\% \]
\[ \langle \{1,2\} \{2,3\} \rangle \quad s=60\% \]
Examples of Sequential Patterns

Table 1. A set of transactions sorted by customer ID and transaction time

<table>
<thead>
<tr>
<th>Customer ID</th>
<th>Transaction Time</th>
<th>Transaction (items bought)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>July 20, 2005</td>
<td>30</td>
</tr>
<tr>
<td>1</td>
<td>July 25, 2005</td>
<td>90</td>
</tr>
<tr>
<td>2</td>
<td>July 9, 2005</td>
<td>10, 20</td>
</tr>
<tr>
<td>2</td>
<td>July 14, 2005</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>July 20, 2005</td>
<td>40, 60, 70</td>
</tr>
<tr>
<td>3</td>
<td>July 25, 2005</td>
<td>30, 50, 70</td>
</tr>
<tr>
<td>4</td>
<td>July 25, 2005</td>
<td>30</td>
</tr>
<tr>
<td>4</td>
<td>July 29, 2005</td>
<td>40, 70</td>
</tr>
<tr>
<td>4</td>
<td>August 2, 2005</td>
<td>90</td>
</tr>
<tr>
<td>5</td>
<td>July 12, 2005</td>
<td>90</td>
</tr>
</tbody>
</table>
Examples of Sequential Patterns

Table 2. Data sequences produced from the transaction database in Table 1.

<table>
<thead>
<tr>
<th>Customer ID</th>
<th>Data 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}</td>
</tr>
<tr>
<td>4</td>
<td>{30} {40, 70} {90}</td>
</tr>
<tr>
<td>5</td>
<td>{90}</td>
</tr>
</tbody>
</table>

Table 3. The final output sequential patterns

<table>
<thead>
<tr>
<th>1-sequences</th>
<th>Sequential Patterns with Support ≥ 25%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{30}, {40}, {70}, {90}</td>
</tr>
<tr>
<td>2-sequences</td>
<td>{30} {40}, {30} {70}, {30} {90}, {40, 70}</td>
</tr>
<tr>
<td>3-sequences</td>
<td>{30} {40, 70}</td>
</tr>
</tbody>
</table>

03/10/14   Heiko Paulheim, Robert Meusel
Sequential Pattern Mining

• Given:
  – a database of sequences
  – a user-specified minimum support threshold, \( \text{minsup} \)

• Task:
  – Find all subsequences with support \( \geq \text{minsup} \)

• Challenge:
  – Very large number of candidate subsequences that need to be checked against the sequence database
  – By applying the Apriori principle, the number of candidates can be pruned significantly
Determining the Candidate Subsequences

- Given \( n \) events: \( i_1, i_2, i_3, \ldots, i_n \)

- Candidate 1-subsequences:
  \(<\{i_1\}>, <\{i_2\}>, <\{i_3\}>, \ldots, <\{i_n\}>\)

- Candidate 2-subsequences:
  \(<\{i_1, i_2\}>, <\{i_1, i_3\}>, \ldots, <\{i_{n-1}, i_n\}>, <\{i_1\} \{i_1\}>, <\{i_1\} \{i_2\}>, \ldots, <\{i_{n-1}\} \{i_n\}>, <\{i_n\} \{i_n\}>\)

- Candidate 3-subsequences:
  \(<\{i_1, i_2, i_3\}>, <\{i_1, i_2, i_4\}>, \ldots, <\{i_1, i_2\} \{i_1\}>, <\{i_1, i_2\} \{i_2\}>, \ldots, <\{i_1\} \{i_1, i_2\}>, <\{i_1\} \{i_1, i_3\}>, \ldots, <\{i_1\} \{i_1\} \{i_1\}>, <\{i_1\} \{i_1\} \{i_2\}>, \ldots\)
Generalized Sequential Pattern Algorithm (GSP)

- **Step 1:**
  - Make the first pass over the sequence database D to yield all the 1-element frequent subsequences

- **Step 2:** Repeat until no new frequent subsequences are found
  1. **Candidate Generation:**
     - Merge pairs of frequent subsequences found in the \((k-1)th\) pass to generate candidate sequences that contain \(k\) items
  2. **Candidate Pruning:**
     - Prune candidate \(k\)-sequences that contain infrequent \((k-1)\)-subsequences (Apriori principle)
  3. **Support Counting:**
     - Make a new pass over the sequence database D to find the support for these candidate sequences
  4. **Candidate Elimination:**
     - Eliminate candidate \(k\)-sequences whose actual support is less than \(\text{minsup}\)
Candidate Generation

• Base case (k=2):
  – Merging two frequent 1-sequences \( \langle i_1 \rangle \) and \( \langle i_2 \rangle \) will produce two candidate 2-sequences: \( \langle i_1 \rangle \{i_2\} \) and \( \langle i_1, i_2 \rangle \)

• General case (k>2):
  – A frequent \((k-1)\)-sequence \( w_1 \) is merged with another frequent \((k-1)\)-sequence \( w_2 \) to produce a candidate \( k \)-sequence if the subsequence obtained by removing the first event in \( w_1 \) is the same as the subsequence obtained by removing the last event in \( w_2 \)
  – The resulting candidate after merging is given by the sequence \( w_1 \) extended with the last event of \( w_2 \).
    • If the last two events in \( w_2 \) belong to the same element, then the last event in \( w_2 \) becomes part of the last element in \( w_1 \)
    • Otherwise, the last event in \( w_2 \) becomes a separate element appended to the end of \( w_1 \)
Candidate Generation Examples

• Merging the sequences $w_1 = \langle \{1\} \{2\ 3\} \{4\} \rangle$ and $w_2 = \langle \{2\ 3\} \{4\ 5\} \rangle$
  will produce the candidate sequence $\langle \{1\} \{2\ 3\} \{4\ 5\} \rangle$ because the last two events in $w_2$ (4 and 5) belong to the same element.

• Merging the sequences $w_1 = \langle \{1\} \{2\ 3\} \{4\} \rangle$ and $w_2 = \langle \{2\ 3\} \{4\} \{5\} \rangle$
  will produce the candidate sequence $\langle \{1\} \{2\ 3\} \{4\ \{5\} \rangle$ because the last two events in $w_2$ (4 and 5) do not belong to the same element.
GSP Example

- Only one 4-sequence survives the candidate pruning step
- All other 4-sequences are removed because they contain subsequences that are not part of the set of frequent 3-sequences

Frequent 3-sequences

- \(< \{1\} \{2\} \{3\} > \)
- \(< \{1\} \{2\} \{5\} > \)
- \(< \{1\} \{5\} \{3\} > \)
- \(< \{2\} \{3\} \{4\} > \)
- \(< \{2\} \{5\} \{3\} > \)
- \(< \{3\} \{4\} \{5\} > \)
- \(< \{5\} \{3\} \{4\} > \)

Candidate Generation

- \(< \{1\} \{2\} \{3\} \{4\} > \)
- \(< \{1\} \{2\} \{5\} \{3\} > \)
- \(< \{1\} \{5\} \{3\} \{4\} > \)
- \(< \{2\} \{3\} \{4\} \{5\} > \)
- \(< \{2\} \{5\} \{3\} \{4\} > \)

Candidate Pruning

- \(< \{1\} \{2\} \{5\} \{3\} > \)
Comparison of Apriori and GSP

- Apriori finds frequent patterns in non-sequential data
- Differences:
  - definition of containment (subset vs. subsequence)
  - generation of candidates (set union vs. merging sequences)
Timing Constraints

• Timing constraints allow us to pose additional restrictions on whether a sequence is counted to support a pattern or not.

• Motivating Example:
  
  < {Statistics} {Database Systems} {Data Mining} >
  
  < {Database Systems} {Statistics} {Data Mining} >

• We are interested in students that support the pattern
  
  < {Database Systems, Statistics} {Data Mining} >

• We don’t care about the order of Database Systems and Statistics

• We care about that the gap between these courses and Data Mining is not too long
Window Size

- Specifies a time window in the data sequence in which all events will be considered to belong to the same element
- Given a candidate pattern: \(<\{a, c\}>\)
- Any data sequences that contain
  \(<\ldots \{a \ c\} \ldots >\),
  \(<\ldots \{a\} \ldots \{c\} \ldots >\) \((\text{where } \text{time}\{\{c\}\} – \text{time}\{\{a\}\} \leq ws)\)
  \(<\ldots \{c\} \ldots \{a\} \ldots >\) \((\text{where } \text{time}\{\{a\}\} – \text{time}\{\{c\}\} \leq ws)\)
will contribute to the support count of the candidate pattern.

\[
\begin{array}{|c|c|c|}
\hline
\text{Data sequence} & \text{Subsequence} & \text{Contain?} \\
\hline
< \{2,4\} \{3,5,6\} \{4,7\} \{4,6\} \{8\} > & < \{3\} \{5\} > & \text{No} \\
\hline
< \{1\} \{2\} \{3\} \{4\} \{5\}> & < \{1,2\} \{3\} > & \text{Yes} \\
\hline
< \{1,2\} \{2,3\} \{3,4\} \{4,5\}> & < \{1,2\} \{3,4\} > & \text{Yes} \\
\hline
\end{array}
\]
Max-Gap, Min-Gap

- **Max-Gap**: Sequence is counted if gap between *consecutive* elements is smaller than $\max_g$.
- **Min-Gap**: Sequence is counted if gap between *consecutive* elements is bigger than $\min_g$.

$max_g = 2, \ min_g = 0$

<table>
<thead>
<tr>
<th>Data sequence</th>
<th>Subsequence</th>
<th>Contain?</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; {2,4} {3,5,6} {4,7} {4,5} {8} &gt;</td>
<td>&lt; {6} {5} &gt;</td>
<td>Yes</td>
</tr>
<tr>
<td>&lt; {1} {2} {3} {4} {5}&gt;</td>
<td>&lt; {1} {4} &gt;</td>
<td>No</td>
</tr>
<tr>
<td>&lt; {1} {2,3} {3,4} {4,5}&gt;</td>
<td>&lt; {2} {3} {5} &gt;</td>
<td>Yes</td>
</tr>
<tr>
<td>&lt; {1,2} {3} {2,3} {3,4} {2,4} {4,5}&gt;</td>
<td>&lt; {1,2} {5} &gt;</td>
<td>No</td>
</tr>
</tbody>
</table>
Sequential Patterns in RapidMiner

- Input data needs to contain:
  - *customer id* attribute being of type integer and
  - *Sequence* attribute being of type integer, real, or date/time
  - all other attributes need to be of type *binominal*
Mining Sequential Patterns with RapidMiner

All parameters must be filled!
Mining Sequential Patterns with RapidMiner

GSPSet

0.500: <Movie ID = StarWars1> <Movie ID = StarWars2>
0.500: <Movie ID = StarWars1> <Movie ID = StarWars3>
0.500: <Movie ID = StarWars2> <Movie ID = StarWars3>
0.500: <Movie ID = StarWars1> <Movie ID = StarWars2, Movie ID = StarWars3>
Wrap Up Sequential Patterns

• Data model: sequences of transactions
• Goal: find frequent sub sequences
  – with a generalized version of Apriori (GSP)
• Relaxing criteria:
  – window size
  – min and max gap
Trend Detection

• Task
  – given a time series
  – find out what the general trend is
    (e.g., rising or falling)

• Possible obstacles
  – random effects: ice cream sales have been low this week due to rain
    • but what does that tell about next week?
  – seasonal effects: sales have risen in December
    • but what does that tell about January?
  – cyclical effects: less people attend a lecture towards the end of the semester
    • but what does that tell about the next semester?
Estimation of Trend Curves

- The freehand method
  - Fit the curve by looking at the graph
  - Costly and barely reliable for large-scaled data mining

- The least-squares method
  - Find the curve minimizing the sum of the squares of the deviation of points on the curve from the corresponding data points
  - cf. linear regression

- The moving-average method

The time series exhibit a downward trend pattern.
Example: German DAX 2013
Linear Trend in RapidMiner
Example: German DAX 2013
A Component Model of Time Series

A time series can consist of four components:

- Long-term trend ($T_t$)
- Cyclical effect ($C_t$)
- Seasonal effect ($S_t$)
- Random variation ($R_t$)

Additive Model:
- Series = $T_t + C_t + S_t + R_t$

Multiplicative Model:
- Series = $T_t \times C_t \times S_t \times R_t$
Seasonal and Cyclical Effects

- Seasonal effects occur regularly each year
  - quarters
  - months
  - ...

- Cyclical effects occur regularly over other intervals
  - every N years
  - in the beginning/end of the month
  - on certain weekdays or on weekends
  - at certain times of the day
  - ...
Identifying Seasonal and Cyclical Effects

• There are methods of identifying and isolating those effects
  – given that the periodicity is known

• Unfortunately, no simple operator in RapidMiner
  – you'll learn how it's done in the exercise
  – Example on the right: R
Identifying Seasonal and Cyclical Effects

• Variation may occur within a year or another period
• To measure the seasonal effects we compute *seasonal indexes*
• Seasonal index
  – degree of variation of seasons in relation to global average

[Link to Seasons Blog Post](http://davidsills.blogspot.de/2011/10/seasons.html)
Identifying Seasonal and Cyclical Effects

• Algorithm
  – Compute the trend $\hat{y}_t$ (i.e., linear regression)
  – For each time period
    • compute the ratio $y_t/\hat{y}_t$
  – For each season (or other relevant period)
    • compute the average of $y_t/\hat{y}_t$
    • this gives us the average deviation for that season

\[
\frac{y_t}{\hat{y}_t} = \frac{T_t \times S_t \times R_t}{T_t} = S_t \times R_t
\]

here, we assume the multiplicative model

the computed ratios isolate the seasonal and random variation from the overall trend*

*) given that no additional cyclical variation exists
Example for Seasonal Effects

- Calculate the quarterly seasonal indexes for hotel occupancy rate in order to measure seasonal variation

- Data:

<table>
<thead>
<tr>
<th>Year</th>
<th>Quarter</th>
<th>Rate</th>
<th>Year</th>
<th>Quarter</th>
<th>Rate</th>
<th>Year</th>
<th>Quarter</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1996</td>
<td>1</td>
<td>0.561</td>
<td>1998</td>
<td>1</td>
<td>0.594</td>
<td>2000</td>
<td>1</td>
<td>0.665</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.702</td>
<td></td>
<td>2</td>
<td>0.738</td>
<td></td>
<td>2</td>
<td>0.835</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.8</td>
<td></td>
<td>3</td>
<td>0.729</td>
<td></td>
<td>3</td>
<td>0.873</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.568</td>
<td></td>
<td>4</td>
<td>0.6</td>
<td></td>
<td>4</td>
<td>0.67</td>
</tr>
<tr>
<td>1997</td>
<td>1</td>
<td>0.575</td>
<td>1999</td>
<td>1</td>
<td>0.622</td>
<td></td>
<td>2</td>
<td>0.708</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.738</td>
<td></td>
<td>2</td>
<td>0.708</td>
<td></td>
<td>3</td>
<td>0.806</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.868</td>
<td></td>
<td>3</td>
<td>0.806</td>
<td></td>
<td>4</td>
<td>0.632</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.605</td>
<td></td>
<td>4</td>
<td>0.632</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This example is taken from the course “Regression Analysis” at University of Umeå, Department of Statistics
Example for Seasonal Effects

- First step: compute trend from the data
  - i.e., linear regression

\[
\hat{y} = 0.639368 + 0.005246t
\]

<table>
<thead>
<tr>
<th>Time (t)</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.561</td>
</tr>
<tr>
<td>2</td>
<td>0.702</td>
</tr>
<tr>
<td>3</td>
<td>0.800</td>
</tr>
<tr>
<td>4</td>
<td>0.568</td>
</tr>
<tr>
<td>5</td>
<td>0.575</td>
</tr>
<tr>
<td>6</td>
<td>0.738</td>
</tr>
<tr>
<td>7</td>
<td>0.868</td>
</tr>
<tr>
<td>8</td>
<td>0.605</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Example for Seasonal Effects

- Second step: compute ratios $y_t/\hat{y}_t$

<table>
<thead>
<tr>
<th>$t$</th>
<th>$y_t$</th>
<th>$\hat{y}_t$</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.561</td>
<td>.645</td>
<td>.561/.645</td>
</tr>
<tr>
<td>2</td>
<td>.702</td>
<td>.650</td>
<td>.702/.650</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

No trend is observed, but seasonality and randomness still exist.

\[ y_t = 0.639368 + 0.005245 \times t \]
Example for Seasonal Effects

- Third step: compute average ratios by season

\[
\text{Average ratio for quarter 1: } \frac{(0.870 + 0.864 + 0.865 + 0.879 + 0.913)}{5} = 0.878
\]

\[
\text{Average ratio for quarter 2: } \frac{(1.080 + 1.100 + 1.067 + 0.993 + 1.138)}{5} = 1.076
\]

\[
\text{Average ratio for quarter 3: } \frac{(1.221 + 1.284 + 1.046 + 1.122 + 1.181)}{5} = 1.171
\]

\[
\text{Average ratio for quarter 4: } \frac{(0.860 + 0.888 + 0.854 + 0.874 + 0.900)}{5} = 0.875
\]
Example for Seasonal Effects

- Interpretation of seasonal indexes:
  - ratio between the time series' value at a certain season and the overall seasonal average

- In our problem:
Example for Seasonal Effects

- Deseasonalizing time series
  - when ignoring seasonal effects, is there still an increase?

\[
\text{Seasonally adjusted time series} = \frac{\text{Actual time series}}{\text{Seasonal index}}
\]

Trend on deseasonalized time series: slightly positive
Dealing with Random Variations

- **Moving average** of order $n$
  \[
  \frac{y_1 + y_2 + \cdots + y_n}{n}, \quad \frac{y_2 + y_3 + \cdots + y_{n+1}}{n}, \quad \frac{y_3 + y_4 + \cdots + y_{n+2}}{n}, \cdots
  \]

- **Key idea:**
  - upcoming value is the average of the last $n$
  - cf.: nearest neighbors

- **Properties:**
  - Smoothes the data
  - Eliminates *random* movements
  - Loses the data at the beginning or end of a series
  - Sensitive to outliers (can be reduced by weighted moving average)
Moving Average in RapidMiner

• Alternatives for average:
  – median, mode, ...
• Alternatives for average:
  – median, mode, ...
Missing Values in Series Data

• Remedies in non-series data:
  – replace with average
  – replace with most frequent
  – …

• Alternatives for series data:
  – replace with previous/next
  – linear interpolation
Dealing with Random Variations

- Exponential Smoothing
  - $S_t = \alpha y_t + (1-w)S_{t-1}$
  - $\alpha$ is a smoothing factor
  - recursive definition
    - in practice, start with $S_0 = y_0$

- Properties:
  - Smoothes the data
  - Eliminates random movements
    - and even seasonal effects for smaller values of $\alpha$
  - Smoothing values for whole series
  - More recent values have higher influence
Dealing with Random Variations
Recap: Trend Analysis

- Allows to identify general trends (upward, downward)
- Overall approach:
  - eliminate all other components so that only the trend remains
- Method for factoring out seasonal variations
  - and compute deseasonalized time series
- Methods for eliminating with random variations (smoothing)
  - moving average
  - exponential smoothing
Time Series Prediction: Definition

- Given a sequence of events
  - predict the next event(s)

<table>
<thead>
<tr>
<th>Day</th>
<th>Weather</th>
<th>Temperature</th>
<th>Wind Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday</td>
<td>Sunny</td>
<td>28°C</td>
<td>13 km/h</td>
</tr>
<tr>
<td>Tuesday</td>
<td>Cloudy</td>
<td>25°C</td>
<td>18 km/h</td>
</tr>
<tr>
<td>Wednesday</td>
<td>Cloudy</td>
<td>26°C</td>
<td>21 km/h</td>
</tr>
<tr>
<td>Thursday</td>
<td>Rain</td>
<td>19°C</td>
<td>35 km/h</td>
</tr>
<tr>
<td>Friday</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Saturday</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Sunday</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
Time Series Prediction: Definition

http://xkcd.com/1245/
Time Series Prediction by Windowing

• Idea: transformation of prediction into “classical” learning problem
• Example: weather forecasting
  – using the weather from the three previous days
• Possible model:
  – sunny, sunny, sunny → sunny
  – sunny, cloudy, rainy → rainy
  – sunny, cloudy, cloudy → rainy
  – ...
## Time Series Prediction by Windowing

<table>
<thead>
<tr>
<th>Date</th>
<th>Weather</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.</td>
<td>Sunny</td>
</tr>
<tr>
<td>2.1.</td>
<td>Cloudy</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Date</th>
<th>Date</th>
<th>Weather-3</th>
<th>Weather-2</th>
<th>Weather-1</th>
<th>Weather</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.</td>
<td>1.1.</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>Sunny</td>
</tr>
<tr>
<td>5.1.</td>
<td>2.1.</td>
<td>?</td>
<td>?</td>
<td>Sunny</td>
<td>Cloudy</td>
</tr>
<tr>
<td>6.1.</td>
<td>3.1.</td>
<td>?</td>
<td>Sunny</td>
<td>Cloudy</td>
<td>Cloudy</td>
</tr>
<tr>
<td>7.1.</td>
<td>4.1.</td>
<td>Sunny</td>
<td>Cloudy</td>
<td>Cloudy</td>
<td>Rainy</td>
</tr>
<tr>
<td>8.1.</td>
<td>5.1.</td>
<td>Cloudy</td>
<td>Cloudy</td>
<td>Rainy</td>
<td>Cloudy</td>
</tr>
<tr>
<td>9.1.</td>
<td>6.1.</td>
<td>Cloudy</td>
<td>Rainy</td>
<td>Cloudy</td>
<td>Sunny</td>
</tr>
<tr>
<td></td>
<td>7.1.</td>
<td>Rainy</td>
<td>Cloudy</td>
<td>Sunny</td>
<td>Sunny</td>
</tr>
<tr>
<td></td>
<td>8.1.</td>
<td>Cloudy</td>
<td>Sunny</td>
<td>Sunny</td>
<td>Sunny</td>
</tr>
<tr>
<td></td>
<td>9.1.</td>
<td>Sunny</td>
<td>Sunny</td>
<td>Sunny</td>
<td>Rainy</td>
</tr>
</tbody>
</table>
Time Series Prediction by Windowing

- New task: classify variable “Weather”
  - using “Weather-3”, “Weather-2” and “Weather-1” as attributes
  - any classifier (Naive Bayes, Decision Trees, …) can be used

<table>
<thead>
<tr>
<th>Date</th>
<th>Weather-3</th>
<th>Weather-2</th>
<th>Weather-1</th>
<th>Weather</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>Sunny</td>
</tr>
<tr>
<td>2.1.</td>
<td>?</td>
<td>?</td>
<td>Sunny</td>
<td>Cloudy</td>
</tr>
<tr>
<td>3.1.</td>
<td>?</td>
<td>Sunny</td>
<td>Cloudy</td>
<td>Cloudy</td>
</tr>
<tr>
<td>4.1.</td>
<td>Sunny</td>
<td>Cloudy</td>
<td>Cloudy</td>
<td>Rainy</td>
</tr>
<tr>
<td>5.1.</td>
<td>Cloudy</td>
<td>Cloudy</td>
<td>Rainy</td>
<td>Cloudy</td>
</tr>
<tr>
<td>6.1.</td>
<td>Cloudy</td>
<td>Rainy</td>
<td>Cloudy</td>
<td>Sunny</td>
</tr>
<tr>
<td>7.1.</td>
<td>Rainy</td>
<td>Cloudy</td>
<td>Sunny</td>
<td>Sunny</td>
</tr>
<tr>
<td>8.1.</td>
<td>Cloudy</td>
<td>Sunny</td>
<td>Sunny</td>
<td>Sunny</td>
</tr>
<tr>
<td>9.1.</td>
<td>Sunny</td>
<td>Sunny</td>
<td>Sunny</td>
<td>Rainy</td>
</tr>
</tbody>
</table>
Windowing in RapidMiner

![Diagram showing Windowing process with Read CSV and Windowing components](image)

<table>
<thead>
<tr>
<th>Row No.</th>
<th>Date</th>
<th>Weather-2</th>
<th>Weather-1</th>
<th>Weather-0</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>04.01.2013</td>
<td>sunny</td>
<td>cloudy</td>
<td>cloudy</td>
<td>cloudy</td>
</tr>
<tr>
<td>2</td>
<td>07.01.2013</td>
<td>cloudy</td>
<td>cloudy</td>
<td>cloudy</td>
<td>rainy</td>
</tr>
<tr>
<td>3</td>
<td>08.01.2013</td>
<td>cloudy</td>
<td>cloudy</td>
<td>rainy</td>
<td>sunny</td>
</tr>
<tr>
<td>4</td>
<td>09.01.2013</td>
<td>cloudy</td>
<td>rainy</td>
<td>sunny</td>
<td>rainy</td>
</tr>
<tr>
<td>5</td>
<td>10.01.2013</td>
<td>rainy</td>
<td>sunny</td>
<td>rainy</td>
<td>cloudy</td>
</tr>
<tr>
<td>6</td>
<td>11.01.2013</td>
<td>sunny</td>
<td>rainy</td>
<td>cloudy</td>
<td>sunny</td>
</tr>
<tr>
<td>7</td>
<td>14.01.2013</td>
<td>rainy</td>
<td>cloudy</td>
<td>sunny</td>
<td>sunny</td>
</tr>
<tr>
<td>8</td>
<td>15.01.2013</td>
<td>cloudy</td>
<td>sunny</td>
<td>sunny</td>
<td>rainy</td>
</tr>
<tr>
<td>9</td>
<td>16.01.2013</td>
<td>sunny</td>
<td>sunny</td>
<td>rainy</td>
<td>sunny</td>
</tr>
<tr>
<td>10</td>
<td>17.01.2013</td>
<td>sunny</td>
<td>rainy</td>
<td>sunny</td>
<td>cloudy</td>
</tr>
<tr>
<td>11</td>
<td>18.01.2013</td>
<td>rainy</td>
<td>sunny</td>
<td>cloudy</td>
<td>cloudy</td>
</tr>
<tr>
<td>12</td>
<td>21.01.2013</td>
<td>sunny</td>
<td>cloudy</td>
<td>cloudy</td>
<td>cloudy</td>
</tr>
<tr>
<td>13</td>
<td>22.01.2013</td>
<td>cloudy</td>
<td>cloudy</td>
<td>cloudy</td>
<td>rainy</td>
</tr>
<tr>
<td>14</td>
<td>23.01.2013</td>
<td>cloudy</td>
<td>cloudy</td>
<td>rainy</td>
<td>sunny</td>
</tr>
<tr>
<td>15</td>
<td>24.01.2013</td>
<td>cloudy</td>
<td>rainy</td>
<td>sunny</td>
<td>rainy</td>
</tr>
<tr>
<td>16</td>
<td>25.01.2013</td>
<td>rainy</td>
<td>sunny</td>
<td>rainy</td>
<td>cloudy</td>
</tr>
<tr>
<td>17</td>
<td>28.01.2013</td>
<td>sunny</td>
<td>rainy</td>
<td>cloudy</td>
<td>sunny</td>
</tr>
</tbody>
</table>
Windowing in RapidMiner

RuleModel

if Weather-0 = rainy and Weather-2 = cloudy then sunny  (0 / 0 / 21)
if Weather-2 = rainy and Weather-1 = sunny then cloudy  (41 / 0 / 0)
if Weather-2 = cloudy then rainy  (0 / 62 / 0)
if Weather-2 = rainy then sunny  (0 / 0 / 20)
if Weather-1 = sunny then sunny  (0 / 0 / 20)
if Weather-1 = cloudy then cloudy  (19 / 0 / 0)
if Weather-0 = sunny then cloudy  (21 / 0 / 0)
else sunny  (0 / 0 / 18)

correct: 222 out of 222 training examples.
Windowing in RapidMiner

- Also possible for multi-variate data

<table>
<thead>
<tr>
<th>Row No.</th>
<th>Date</th>
<th>Weather-2</th>
<th>Weather-1</th>
<th>Weather-0</th>
<th>Temperature-2</th>
<th>Temperature-1</th>
<th>Temperature-0</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>04.01.2013</td>
<td>sunny</td>
<td>cloudy</td>
<td>cloudy</td>
<td>23</td>
<td>24</td>
<td>28</td>
<td>cloudy</td>
</tr>
<tr>
<td>2</td>
<td>07.01.2013</td>
<td>cloudy</td>
<td>cloudy</td>
<td>cloudy</td>
<td>24</td>
<td>28</td>
<td>32</td>
<td>sunny</td>
</tr>
<tr>
<td>3</td>
<td>08.01.2013</td>
<td>cloudy</td>
<td>cloudy</td>
<td>rainy</td>
<td>28</td>
<td>32</td>
<td>19</td>
<td>sunny</td>
</tr>
<tr>
<td>4</td>
<td>09.01.2013</td>
<td>cloudy</td>
<td>rainy</td>
<td>sunny</td>
<td>32</td>
<td>19</td>
<td>24</td>
<td>rainy</td>
</tr>
<tr>
<td>5</td>
<td>10.01.2013</td>
<td>rainy</td>
<td>sunny</td>
<td>rainy</td>
<td>19</td>
<td>24</td>
<td>25</td>
<td>cloudy</td>
</tr>
<tr>
<td>6</td>
<td>11.01.2013</td>
<td>sunny</td>
<td>rainy</td>
<td>cloudy</td>
<td>24</td>
<td>25</td>
<td>17</td>
<td>sunny</td>
</tr>
<tr>
<td>7</td>
<td>14.01.2013</td>
<td>rainy</td>
<td>cloudy</td>
<td>sunny</td>
<td>25</td>
<td>17</td>
<td>14</td>
<td>sunny</td>
</tr>
<tr>
<td>8</td>
<td>15.01.2013</td>
<td>cloudy</td>
<td>sunny</td>
<td>sunny</td>
<td>17</td>
<td>14</td>
<td>12</td>
<td>rainy</td>
</tr>
<tr>
<td>9</td>
<td>16.01.2013</td>
<td>sunny</td>
<td>sunny</td>
<td>rainy</td>
<td>14</td>
<td>12</td>
<td>26</td>
<td>sunny</td>
</tr>
<tr>
<td>10</td>
<td>17.01.2013</td>
<td>sunny</td>
<td>rainy</td>
<td>sunny</td>
<td>12</td>
<td>26</td>
<td>23</td>
<td>cloudy</td>
</tr>
<tr>
<td>11</td>
<td>18.01.2013</td>
<td>rainy</td>
<td>sunny</td>
<td>cloudy</td>
<td>26</td>
<td>23</td>
<td>24</td>
<td>cloudy</td>
</tr>
<tr>
<td>12</td>
<td>21.01.2013</td>
<td>sunny</td>
<td>cloudy</td>
<td>cloudy</td>
<td>23</td>
<td>24</td>
<td>28</td>
<td>cloudy</td>
</tr>
<tr>
<td>13</td>
<td>22.01.2013</td>
<td>cloudy</td>
<td>cloudy</td>
<td>cloudy</td>
<td>24</td>
<td>28</td>
<td>32</td>
<td>rainy</td>
</tr>
<tr>
<td>14</td>
<td>23.01.2013</td>
<td>cloudy</td>
<td>cloudy</td>
<td>rainy</td>
<td>28</td>
<td>32</td>
<td>19</td>
<td>sunny</td>
</tr>
<tr>
<td>15</td>
<td>24.01.2013</td>
<td>cloudy</td>
<td>rainy</td>
<td>sunny</td>
<td>32</td>
<td>19</td>
<td>24</td>
<td>rainy</td>
</tr>
<tr>
<td>16</td>
<td>25.01.2013</td>
<td>rainy</td>
<td>sunny</td>
<td>rainy</td>
<td>19</td>
<td>24</td>
<td>25</td>
<td>cloudy</td>
</tr>
<tr>
<td>17</td>
<td>28.01.2013</td>
<td>sunny</td>
<td>rainy</td>
<td>cloudy</td>
<td>24</td>
<td>25</td>
<td>17</td>
<td>sunny</td>
</tr>
</tbody>
</table>
Windowing in RapidMiner

- Also possible for multi-variate data

**RuleModel**

if Temperature-0 ≤ 21 and Temperature-2 > 20.500 then sunny (0 / 0 / 62)
if Temperature-1 > 21 and Temperature-1 ≤ 27 then cloudy (81 / 0 / 0)
if Weather-2 = cloudy then rainy (0 / 62 / 0)
else sunny (0 / 0 / 18)

correct: 223 out of 223 training examples.
Windowing in RapidMiner

- Also possible for numerical prediction
  - the learning problem becomes a regression problem
Wrap-up

• Time series data is data sequentially collected at different times

• Analysis methods discussed in this lecture
  – frequent pattern mining
  – trend analysis
  – predictions with windowing
Questions?