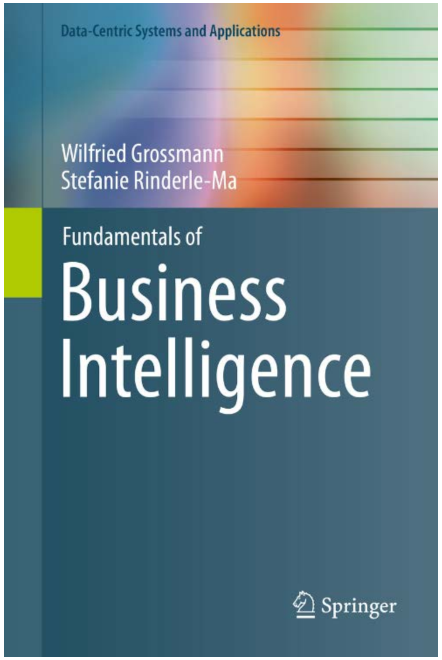


Chapter 6:
Data Mining for Temporal Data

Contents

- Basic ingredients
  - A sequence of ordered time values, called observation times: $t_1 \leq t_2 \leq \cdots \leq tT$
  - Attribute values at these times: $x_1 \leq x_2 \leq \cdots \leq x_T$
- Time sequence (time stamped data):

$$x = < (t_1, x_1), \ldots, (tT, xT) >$$

- Two important goals in temporal data mining:
  - Classification of time sequences into classes
  - Finding clusters of time sequences

Time Sequences, Time Series, State Sequence

- A *time sequence* is defined as a sequence of time-stamped data for which the attribute values are the result of measurements of a quantitative real valued state variable y , i.e., $y \in \mathbb{R}$. We denote the observations of a time sequence by $y = (y(t_1), \dots, y(t_T))$
- A *time series* is a time sequence with equidistant predefined observation times denoted by $y = (y_1, \dots, yT)$
- A *state sequence* is a time sequence where the state variable S attains only a finite number of possible values given by a set $\mathscr{S}$. If the observation times are of minor importance, or even not known, we denote a chain simply as ordered sequence of observations of the state variable $s =< s_1, \dots, sT >, si \in \mathscr{S}$

Event Set,

- Given a set $\mathscr{E}$ of events an event set is subset E of $\mathscr{E}$.
- An event sequence is an ordered list of events $s =< e_1, \ldots, eT >$.If the times of the events are known event sequences are denoted by $s =< (e_1, t_1), \ldots, (eT, tT)) >$.

Problem formulation:

- A main issue is the representation of temporal data for the analysis
  - Non-adaptive representation: transform time sequence into a feature space, e.g. Fourier transformation
  - Adaptive representation: extract features of the time sequence, which can be used for analysis
  - Data clipping: transform time sequence into a bit string
  - Model based representation: time sequence as input for a model, e.g. a Markov chain
- We will focus on methods based on adaptive representation

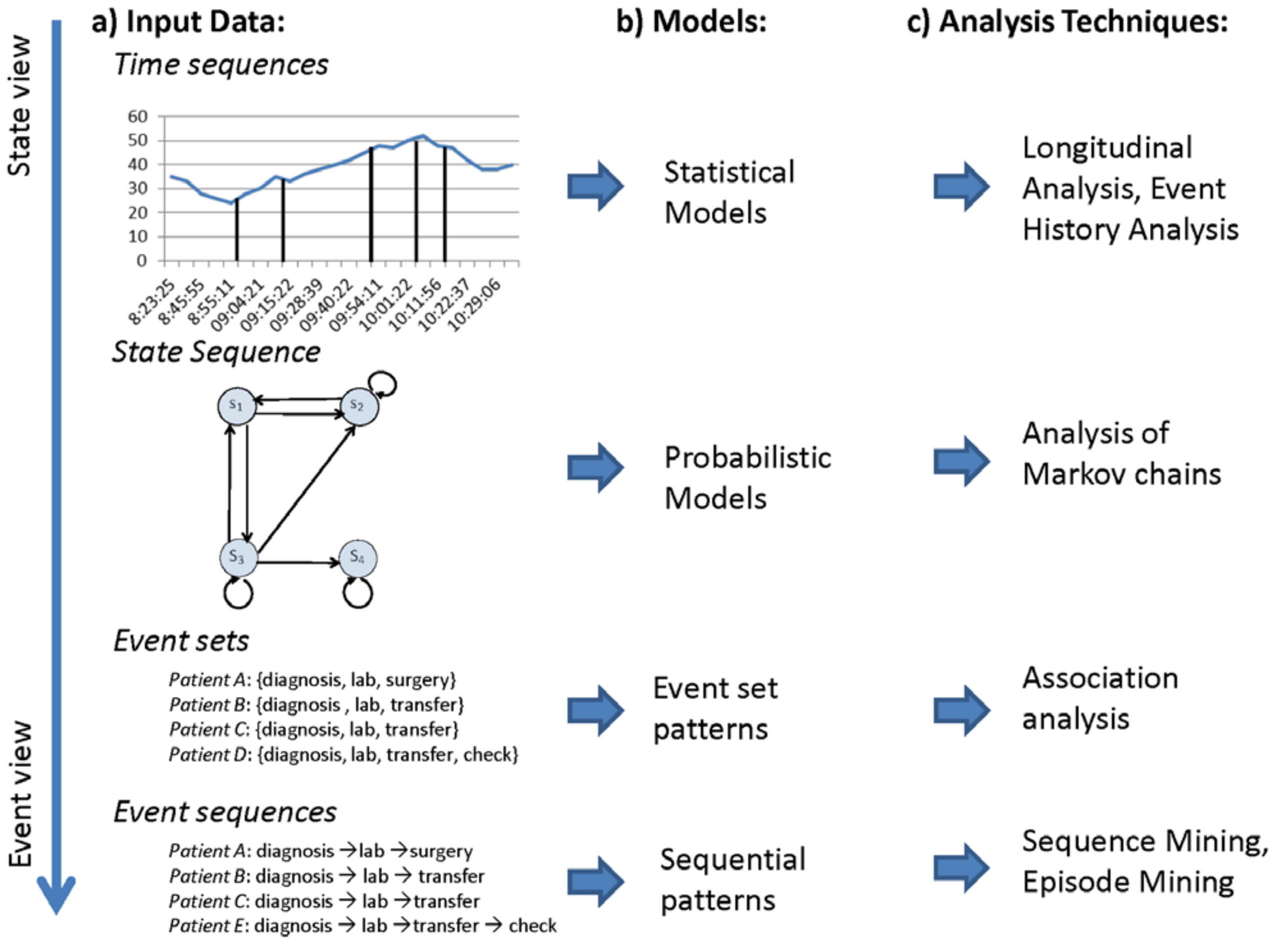
Analysis Template

- *Relevant Business and Data*: Customer behavior represented as time sequence
- *Analytical Goals*:
  - Classification of a new time sequence into one of the possible classes
  - Segmentation of time sequences according to their structural similarity
- *Modeling Task*: Using visualization techniques for the time sequences of the process instances can support decision for a useful method:
  - Time warping for defining distances
  - Response features

Analysis Template

- *Analysis task*:
  - Splitting data: If possible split the data randomly in one set for training and one set for validation
  - Model estimation: Estimate the warping path or the response features
  - Model Assessment: Assess quality of the model
  - Model Selection: Select a model
  - Use the results of model estimation for segmentation or classification
- *Evaluation and Reporting Task*: Evaluate the results of segmentation or classification either with test data or by using cross validation

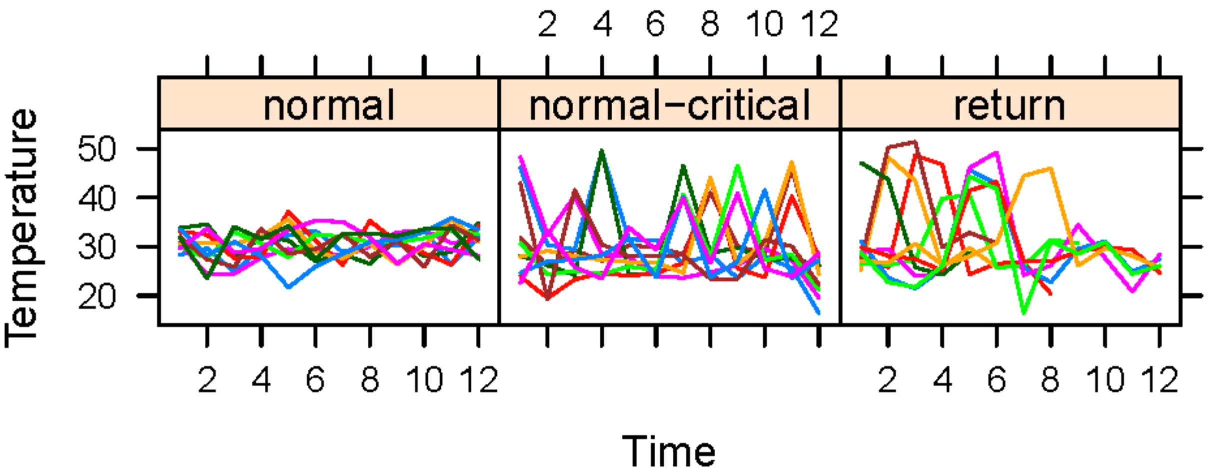
1 Introduction



Contents

Dynamic Time Warping

- Example: Logistic Use Case
- The data show three different kinds of behavior:
  - A normal temperature regime
  - A critical temperature regime
  - A return temperature regime
- Goal is the identification of the regime



R package lattice

Classification based on time warping

- General Problem formulation:
- Given are data of customer behavior represented as time sequences for process instances
- These data are classified into different groups
- Task: Find a classification rule which allows the assignment of a time sequence to one of the classes

Classification based on time warping

- Basic idea behind time warping:
  - Classes are defined by time series which show a “similar pattern”
  - The term similarity is understood in the sense of speech waves: different persons spell words differently but we can classify the waves to words
- Problem which have to be taken into account:
  - Time sequences may have different length
  - Similarity may be blurred by some temporal transformations like stretching or squeezing some parts of the time sequence (see example)
  - We have to define the similarity by matching the observed values of two time sequences in such a way that the above defined effects are compensated

Classification based on time warping

- Dynamic time warping allows the calculation of similarity
- Basic is the definition of a warping path:

Given two sequences $(x_1, \dots, xN)$ and $(y_1, \dots, yM)$:

Define a sequence $(p_1, \dots, p_L)$ of matching indices pairs (i_l, jl) such that

$$p_1 = (1{,}1) \quad p_L = (N, M)$$

$(i_1 \ \leq i_2 \leq \cdots \leq iL)$ and $(j_1 \ \leq j_2 \leq \cdots \leq j_L)$

$$p_{l\ +\ 1\ -}\, pl \ \in \{(0{,}1), (1{,}0), (1{,}1)\}$$

- The last condition means that we increase the matching index at least by one step ahead

Classification based on time warping

- The costs of a warping path is defined by

$$DP = \sum_{l=1}^{L} d(i_l, jl) = \sum_{l=1}^{L} |x_{il} - yil|$$

- The dynamic time warping algorithm finds a warping path for two sequences with minimal costs
- The word "dynamic" indicates that the algorithm is based on dynamic programming

Classification based on time warping

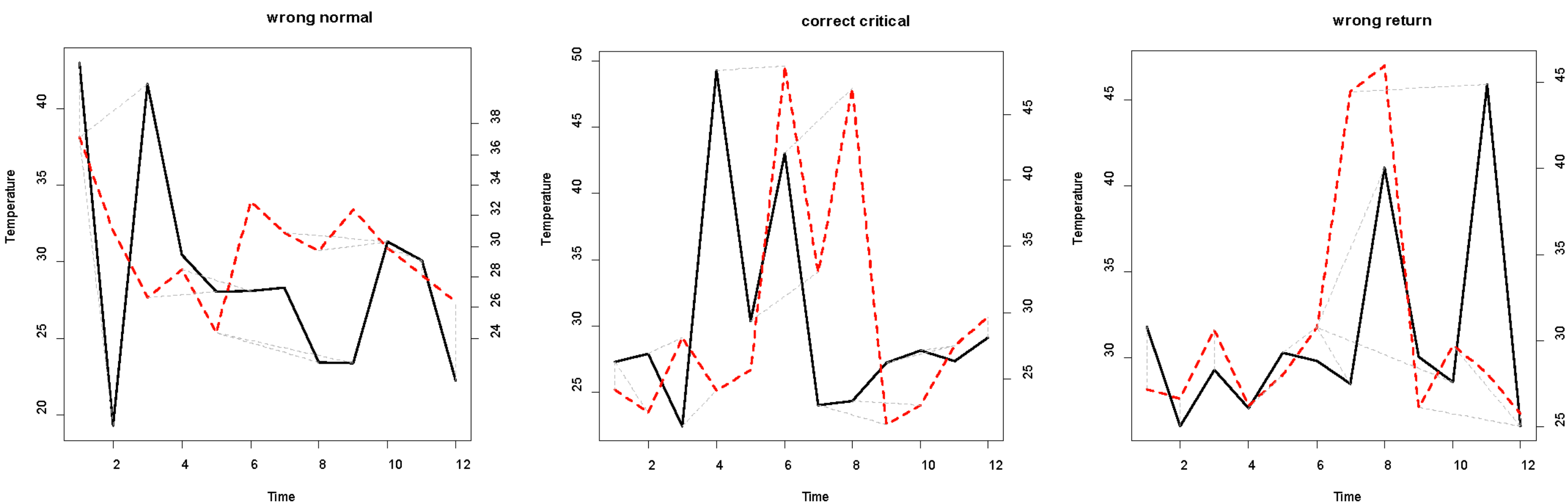
- Application of the dynamic warping algorithm for all pairs of sequences defines a distance matrix for the observed time sequences
- We can apply now k-nearest neighbor classification for obtaining the classification rule

2 Classification and clustering of time sequences

Classification based on time warping

`R package dtw`

– Example: Logistics
  - distance matrix between 100 time sequences defines the input for hierarchical clustering
  - Ward method found 3 clusters: one with 50 normal correctly classified cases; one with 5 normal-critical cases and 5 return cases; one with 40 cases comprising 25 correctly classified return cases and 15 normal-critical cases



Classification Based on Response Features

– In that case we extract from the time sequence a number of time independent characteristic features
– Some examples of features:
  - Maximum and minimum of the time sequence
  - Temporal location of maximum and minimum
  - Breakpoints in the time sequence
  - Largest difference between two sequenced values
  - Length of the sequence
  - Area under the polygon defined by the sequence

Classification Based on Response Features

- More theoretically motivated features:
  - Transformation to frequencies and looking at the maximum frequency (Time sequence is sound or light)
  - Definition of a regression model for the time sequence
    - For equally spaced time measurements this can be done by time series analysis
    - For unequal spaced time measurements this is done by longitudinal data analysis
  - Definition of a representation language
- Based on these attributes one can apply methods of the classification of cross sectional data

Summary:

- Clustering of time sequences can be done using the same principles as in the case of classification
- The definition of time warping defines a distance for the sequences which can be used as input for cluster analysis (hierarchical or k-means)
- In the case of response features the distance between the time sequences is based on the definition of a distance for the response features

Contents

Problem formulation and terminology

- In Time-to-Event Analysis we are interested in modeling and predicting the time up to a certain event[1,2]
- Examples:
  - Prediction of the duration until a customer will quit her/his relationship with a company
  - Prediction of the duration of the lifetime of a certain device
- Other notions for such problems:
  - Event History Analysis
  - Survival Analysis

[1]G. Broström: Event History Analysis with R. CRC Press Taylor & Francis Group 2012
[2]R package `survival`

Problem formulation and terminology (ctd.)

– The time up to the event is called life time

– Main characteristic of the available data:

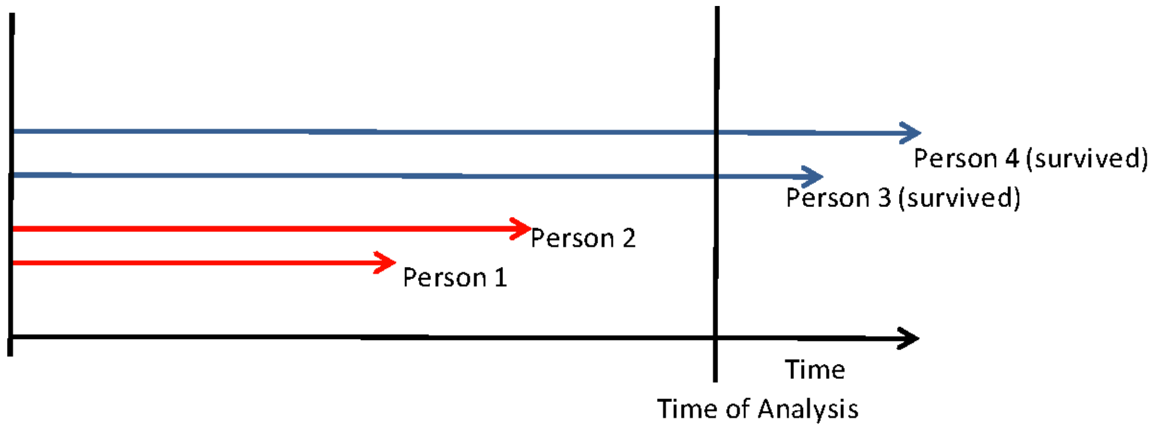
  - The data about the lifetime are **censored**, i.e., for some customers the event is observed, for others the event will occur in the future
  - This type of censoring is called **right censored**

Terminology

- The time up to the event is denoted by and *T* is a random variable
- The probability that the event occurs before time t is denoted by $F(t) = P(T \leq t)$
- The survival function is the probability that the event occurs after time t: $S(t) = 1 - F(t)$
- The mean of the survival function is called the expected survival time
- The hazard function gives the likelihood that the event occurs at time t, given that the event has not occurred up to time t, formally:

$$h(t) = \frac{F'(t)}{1 - F(t)} = \frac{f(t)}{1 - F(t)}$$

- Graphical representation for two complete (red) and two censored (blue) lifetime observations



- Besides the censored lifetime usually other information about the customers is known, e.g. age, occupation, type of machine

Analysis template:

– *Relevant Business and Data*: Customer behavior represented by cross-sectional data and time sequences containing censored information about a terminal event
– *Analytical Goals*: Predict the duration up to the event for the censored time sequences from the uncensored data
– *Modeling Tasks*:
  - Definition of a survival table
  - Definition of a Cox regression model for the time to event
– *Analysis Tasks*:
  - Estimate the time up to the event using the Kaplan Meier estimate
  - Estimation of the coefficients in the Cox regression model
– *Evaluation and Reporting Task*: Evaluate the results using a method for the evaluation of regression

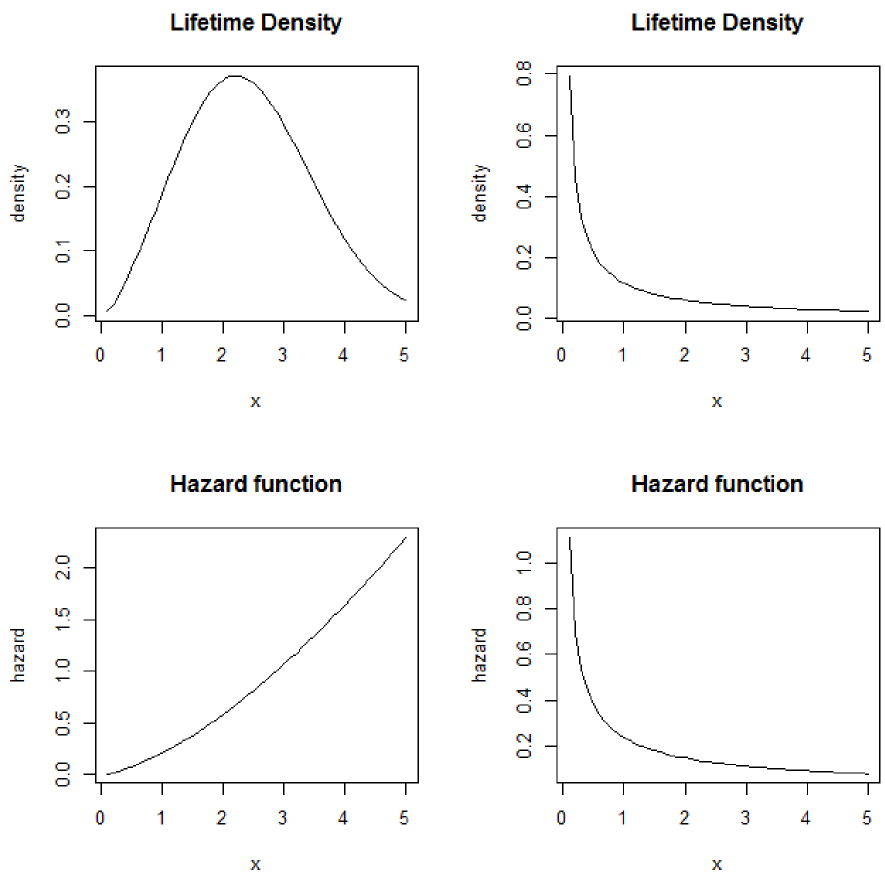
Modeling the survival function

- A frequently used class of model in time-to-event analysis are Weibull distributions defined as:

$$F(t) = 1 - \exp[-(\alpha t)^{\beta}]$$
$$f(t) = \beta * (\alpha t)^{\beta-1} \alpha * \exp[-(\alpha t)^{\beta}]$$

- which allows adaptation to different scenarios like increasing hazard or decreasing hazard by choosing appropriate parameters

– Examples of survival functions



R package graphics

Estimation of the survival function

- The basic information about the survival function is given by the Kaplan Meier estimate, which is summarized in the survival table with the following columns:
  - Time interval
  - Number of persons entering the interval (*n.risk*)
  - Number of events occurred in the interval (*n.event*)
  - Value of the survival function at the end of the time interval (*survival*)
  - The standard error of the estimate for the survival function
  - Confidence interval for the survival function

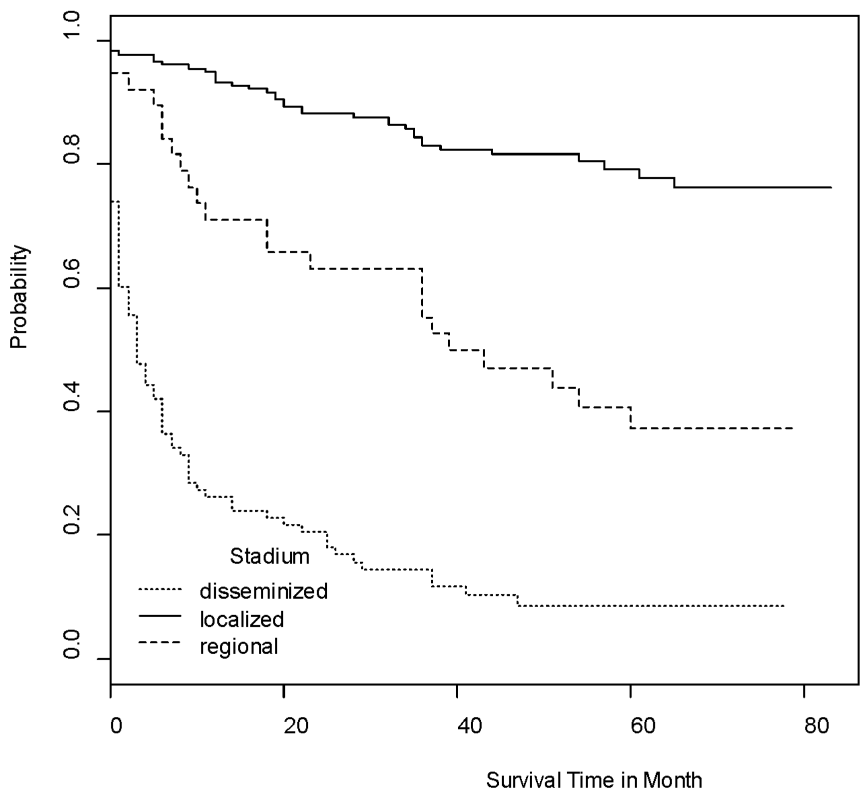
Example: survival table

- 305 patients with different types of melanoma observed from 2006 - 2010

Year	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
0	305	69	0.774	0.0240	0.728	0.822
1	236	23	0.698	0.0263	0.649	0.752
2	213	19	0.636	0.0275	0.584	0.692
3	174	16	0.578	0.0286	0.524	0.637
4	136	6	0.552	0.0292	0.498	0.612
5	86	4	0.526	0.0305	0.470	0.590

- Survival time can be plotted for groups of the population defined by some factors

– Example (ctd.): plot of survival function for the three groups



- **survival functions for the three different values of stadium.**
- **disseminated cases have the worst prognosis for survival time and localized cases the best.**

`R package survival`

Cox Regression

- If there are additional explanatory variables for the occurrence of the event one can estimate the hazard rate with Cox regression, also known as proportional hazard model
- The model defines a time dependent baseline hazard for all observations which is modified according to the explanatory variables

Cox Regression

- Estimation of the survival function, formally:

  $$h(t) = h_0(t)\exp(\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k)$$

- Interpretation of the parameters:
- For a quantitative explanatory variable x the relative risk changes by $\exp(\beta)$ if *x* is increased by one unit
- For a dummy variable representing a factor level the relative risk changes by compared to a reference level
- Example:
- For the 305 patients the influence of the explanatory variables age at diagnosis and stadium of the tumor is of interest
- The results are shown on the next slide

3 Time to event analysis

```
coxph(formula = Surv(Time, Event) ~ Age_Diagnosis + Stadium,
    data = vie1)

   n= 305, number of events= 137

                      coef exp(coef) se(coef)       z Pr(>|z|)
Age_Diagnosis      0.02991   1.03036  0.00653   4.580 4.64e-06 ***
Stadiumlocalized -2.64494   0.07101  0.21324 -12.404  < 2e-16 ***
Stadiumregional  -1.41158   0.24376  0.24521  -5.756 8.59e-09 ***
Stadiumunknown         NA        NA  0.00000      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                 exp(coef) exp(-coef) lower .95 upper .95
Age_Diagnosis      1.03036     0.9705   1.01726    1.0436
Stadiumlocalized   0.07101    14.0826   0.04675    0.1079
Stadiumunknown          NA         NA        NA        NA

Concordance= 0.835  (se = 0.027 )
Rsquare= 0.456   (max possible= 0.992 )
Likelihood ratio test= 185.5  on 3 df,   p=0
Wald test            = 166.9  on 3 df,   p=0
Score (logrank) test = 236.6  on 3 df,   p=0
```

Contents

Two representations:

– Probabilistic representation:
  - Stochastic matrix $P = (p_{ij})$ defined by the transition probabilities from state s_i to state s_j in one time step.
  - All entries are positive and the rows sum up to one.
  - Transition matrix after n steps is denoted by *P(n)*. By using the Chapman-Kolmogorov equations, *P(n)* can be calculated by matrix multiplication, i.e., $P(n) = P^n$.
  - If we denote the initial probabilities for the possible states at *t = 0* by ${\mu_0}^{(i)} = P(S_0 = si)$ and by ${\mu_n}^{(i)}$ the probabilities of the states at time *T* = *n* we can calculate the probabilities of the different states after n time steps by $\mu_n = \mu_0 * P^n,\ \mu_0 = (\mu_0^{(1)}, \ldots, \mu_0^{(K)})$

– Graphical representation obtained by interpreting the matrix of transition probabilities as weighted adjacency matrix of a directed graph with nodes defined by the states of the process.

4 Analysis of Markov Chains

Analysis template:

- *Relevant Business and Data*: Process instances represented as states or event sequences
- *Analytical Goals*:
  - Estimation of state transitions from exiting instances
  - Structural behavior of state transitions in the long run
  - Segmentation of sequences into groups
  - Segmentation of the states
- *Modeling Tasks*: Definition of a stationary Markov chain for state transitions
- *Analysis Tasks*:
  - Estimation of transition probabilities
  - Estimate of a stable distribution
  - Cluster analysis for instances of state or event sequences
  - Cluster analysis of the states or events
- *Evaluation and Reporting Task*: cf. Chapter 5

4 Analysis of Markov Chains

Estimation problems: based on structural analysis of Markov chains

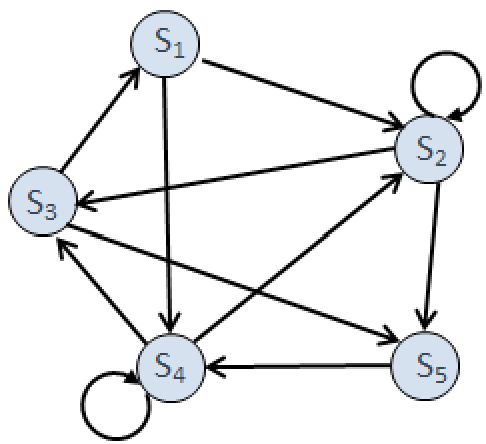
- Goal: finding transition probabilities in the long run
- Important: classification of the states of a Markov chain with respect to the transition behavior
- Basis: typology of states based on *graph representation*
  - state s_i is *reachable* from state s_j if there is a path from s_i to s_j; (s_i,s_j) defines a path of length 1 (then: s_i, s_j directly linked)
  - s_i, s_j are *connected if* s_i is reachable from s_j and vice versa,
  - connected states define a *closed path*
  - *connectedness* defines a partition of all states into classes of connected states
  - A Markov chain is called *irreducible* if each state can be reached from any other state in finite time, i.e., all states belong to one class.
  - Closed set of states as states which cannot be left as soon as we have reached the states.
  - An *absorbing* state is a closed state not connected to any other state. For an absorbing state s_i, we have $p_{ii} = 1$.

Estimation problems:

- A state is called *transient* if there is a positive probability of not returning into the state.
- A state is called *recurrent* if the probability of returning into the state is 1.
- In the case of irreducible Markov chains, all states are either recurrent or transient.
- For recurrent states we can define the period as the largest common divisor of all times *t* for which $p_{ii}(n) > 0$.
- If the period of a state is 1, the state is called *aperiodic*.
- A Markov chain where all states are aperiodic is called *ergodic*.

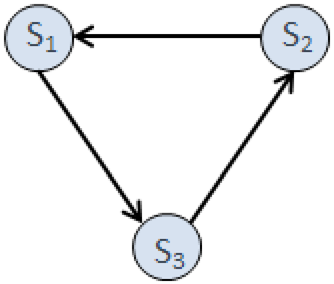
- Left side (a): ergodic
- Right side (b): length of period = 3

	S_1	S_2	S_3	S_4	S_5
S_1	0	0.6	0	0.4	0
S_2	0	0.3	0.4	0	0.3
S_3	0.6	0	0	0	0.4
S_4	0	0.4	0.5	0.1	0
S_5	0	0	0	1.0	0



a) Irreducible Markov Chain

	S_1	S_2	S_3
S_1	0	0	1
S_2	1	0	0
S_3	0	1	0



b) Periodic Markov Chain

Estimation of transition probabilities:

– *Given*: N state sequences $s_1, s_2, \dots, s_N$ of possibly different length, generated by a homogeneous Markov chain with K states $s_1, s_2, \dots, s_K$
– *Goal*: estimate the transition probabilities p_{ij}
– $p_{ij} \neq 0$ only for those transitions for which an edge between the vertices exists in the graph representation.
– If all transitions are generated independently, the distribution of the number of transitions from a state s_i to its directly linked states is a *multinomial* distribution.
– Given a number of state sequences, we denote by n_{ij} the observed number of one step transitions from state s_i to state s_j and by n_i the observed number of occurrences of state s_i.

Estimation of transition probabilities – methods:

– *Maximum likelihood* estimate: $\hat{p}_{ij} = \frac{n_{ij}}{n_i}$

– → if transition from s_i to s_j is not observed, $\hat{p}_{ij} = 0$ though it might be possible from the structure of the Markov chain

– *Bayesian approach*: prior distribution is assumed; estimates are calculated as means of posterior distribution

- Prior Dirichlet distribution: $P(p_{i1}, p_{i2}, \ldots, p_{iK}) = C \prod p_{ij}^{\alpha_{ij}-1}, \alpha_i > 0$ called concentrations
- Estimates: $\tilde{p}_{ij} = \frac{n_{ij}+\alpha_i}{n_i+\alpha_0}$
- Example: prediction of page requests on the Internet; α_j relations between outgoing links of the pages; α_0 number of observations necessary for substantial change of prior beliefs

Cluster analysis for Markov chains:

- Goal: finding groups of Markov chains with similar structure based on cluster algorithm[3]
- Idea:
  - Interpret event sequence s as Markov chain M (events as states s_i)
  - Assign probability that event sequence is generated by a Markov chain, following Maximum likelihood estimate (previous slide):
  - $P(s \mid \mathrm{M}) = \pi(s_1) * \prod P_M(s_i \mid s_k)$ with
  - P_M transition probabilities of Markov chain M and π_M initial probability
  - Clustering of Markov chains based on such transition probabilities resembling k-means clustering

[3]Rebuge A, Ferreira DR (2012) Business process analysis in health care environments: A methodology based on process mining. Information Systems 37(2):99–116

Cluster analysis for Markov chains:

– Example:
– $\mathcal{S}$ = {`CN, CP, HN, HP, EX, start, end`}
– Event sequence `<start, CN, CN, CP, HN, CN, CP, HP, EX, end>`
– This generates a Markov chain with probabilities
  - $P(CN \mid start) = 1$
  - $P(CN \mid CN) = 1/3$, $P(CP \mid CN) = 2/3$
  - $P(HN \mid CP) = P(HP \mid CP) = ½$
  - $P(CN \mid HN) = P(EX \mid HP) = P(end \mid EX) = 1$

Contents

5 Association analysis

– *Input*[4]:
  - Set of items $I := \{i_1, \ldots, i_n\}$
  - T defines a set of transactions; each transaction t ∈ T is defined as a vector $t :=< t[1], \ldots, t[n] >$ with $t[j] = 1$ if item i_j is associated with t and $t[j] = 0$ otherwise
  - $X \subseteq I$ denotes the item set of interest, i.e., we are looking for rules $X \Longrightarrow I_j$ with $I_j \epsilon\ I$ and $I_j \notin X$
  - t ∈ T satisfies X if ∀ $x \in X: t[x] = 1$
- *Explanation*: the goal is to find association rules $A \Longrightarrow B$ where the occurrence of A implies the occurrence of B; A is called the antecedent, B the consequent of the rule

[4]Agrawal R, Imielinski T, Swami A (1993) Mining association rules between sets of items in large databases. ACM SIGMOD Record 22(2):207–216

Example (Hospital):

- Transaction 1:
  - Event 1 = Prescribe Aspirin
  - Event 2 = Prescribe Marcumar
- Transaction 2:
  - Event 1 = Prescribe Aspirin
  - Event 2 = Prescribe Marcumar
- Transaction 3:
  - Event 1 = Prescribe Aspirin
  - Event 2 = Prescribe Paracetamol
- Transaction 4:
  - Event 1 = Prescribe Aspirin
  - Event 2 = Prescribe Marcumar

Example (Hospital), ctd.

- Assume item set I = {Aspirin, Marcumar, Paracetamol}

Transaction	Aspirin	Marcumar	Paracetamol
$t_1 = \langle 1,1,0 \rangle$	1	1	0
$t_2 = \langle 1,1,0 \rangle$	1	1	0
$t_3 = \langle 1,0,1 \rangle$	1	0	1
$t_4 = \langle 1,1,0 \rangle$	1	1	0

- All transactions support item set {Aspirin}

Confidence of Assocation Rule

- Let T be a set of transactions, $A \subseteq I$ be an item set of interest and $B \epsilon I$ be an item set. Then the confidence c of a rule: $A \Longrightarrow B$ is defined as follows: $c(R,T) := \frac{|\{t\ with\ t\ satisfies\ A \cup B\}|}{A}$
- Confidence enables to measure the rule strength
- *Example*: R: Aspirin $\Rightarrow$ Marcumar; c(R,T) = 0.75

Support of Association Reuls

- Let T be a transaction set and R: A $\Rightarrow$ B a rule. Then the support s(R, T) is defined as s(R,T):= $\frac{|\{t\ with\ t\ satisfies\ \{A,B\}\}|}{|T|}$
- *Example*: R: Aspirin $\Rightarrow$ Marcumar; s(R,T) = 0.75

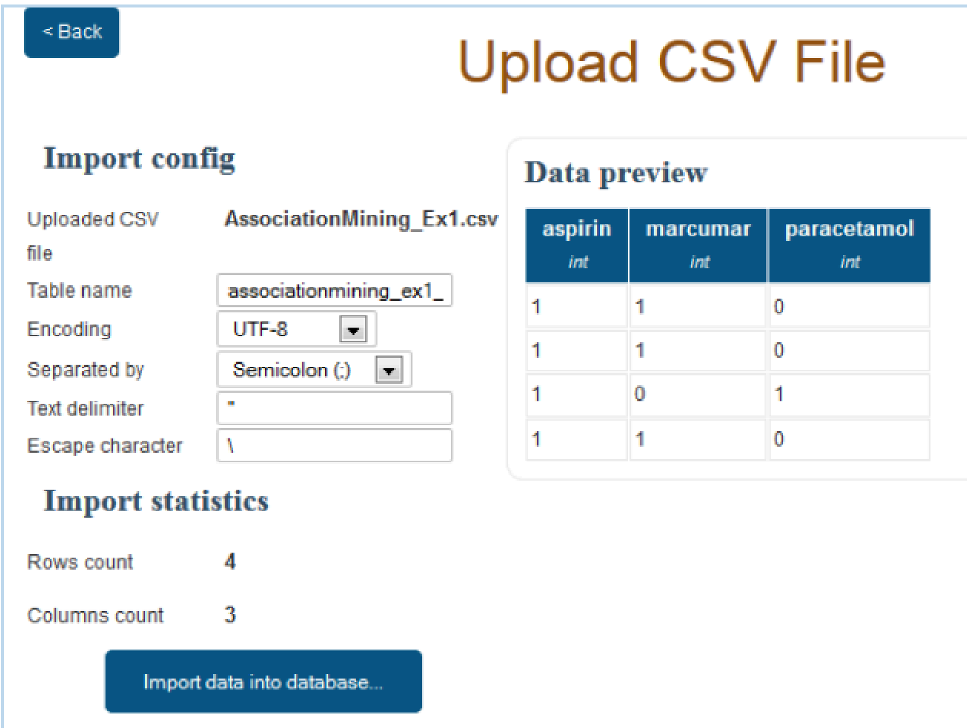
It holds: $c(R,T) = \frac{s(R,T)}{s(A\Rightarrow,T)}$

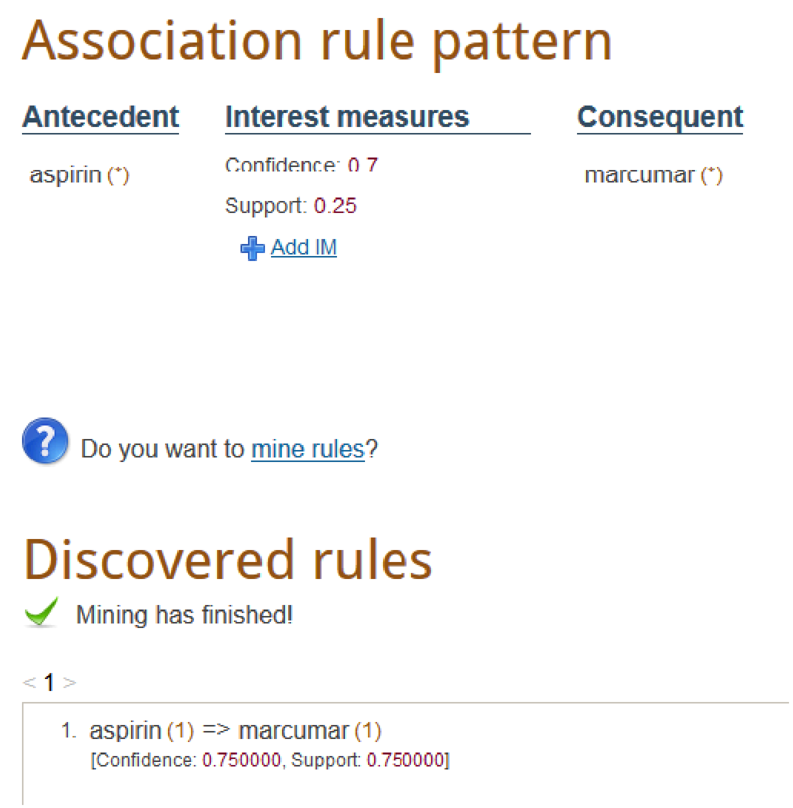
Main Steps in Association Analysis:

- Find large item sets:
  - Define minimal support
  - Find all time sets, for whichtheir support exceeds the threshold (-→ large item sets)
- Discover rules within large item sets:
  - Define minimal confidence
  - Determine all possible rules in the large item sets that exceed confidence and adhere additional syntactical constraints
- *Example*: minimal support = 0.2; minimal confidence: 0.5; syntactical constraints: antecedent must contain Aspirinm consequent not empty
  - Large item sets: {A}, {M}, {P}, {A, M}, {A, P}
  - Rules in large item sets with c>0.2; R1: A $\Rightarrow$ M and R2: A $\Rightarrow$ P
  - Result: R1 with c(R1, T) = 0.75

5 Association analysis

Easy Miner: http://www.easyminer.eu/





Contents

Medical example

Aspirin	BetaBlock	Ibu	Antibiotics	Time stamp	Patient
1	0	0	0	10.10.2013	P1
1	0	1	0	12.10.2013	P2
0	0	0	1	13.10.2013	P2
0	1	0	0	14.10.2013	P1
1	0	0	0	15.10.2013	P3
0	0	1	0	16.10.2013	P3
1	0	1	0	17.10.2013	P4
0	1	0	1	18.10.2013	P4

Patient ID	Item set	Sequence
P1	{Aspirin, BetaBlock}	<Aspirin, BetaBlock>
P2	{Aspirin, Ibu, Antibiotics}	<Aspirin, Ibu, Antibiotics>
P3	{Aspirin, Ibu}	<Aspirin, Ibu>
P4	{Aspirin, Ibu, BetaBlock, Antibiotics}	<Aspirin, Ibu, BetaBlock, Antibiotics>

Sequence mining[5]

- So far: mining of rules, e.g., Aspirin $\Rightarrow$ {Ibu, Antibiotics} with support 0.5 and confidence 0.5
- Not known: order of Ibu and Antibiotics
- Goal of sequence mining:
- If Aspirin, then Ibu followed by Antibiotics

[5]Agrawal R, Srikant R (1995) Mining sequential patterns. In: Yu PS, Chen ALP (eds) ICDE’95: International Conference on Data Engineering, IEEE, pp. 3–14

Sequence mining – input:

- $\mathcal{I} := \{i_1, \ldots, in\}$ defines the set of items
- T defines a set of transactions; a time stamp is assigned to each transactions
- $S \coloneqq \langle s_1, \ldots, sk \rangle$ denotes a sequence of item sets
- A sequence S is contained in another sequence S‘, i.e., $S \prec S'$ if $\forall\, s \ \in S$: $\exists s' \in S' with\ s \ \subseteq s'$
- T is called *customer sequence*; it constitutes an ordered sequence of transactions referring to item sets, i.e., it is a item set itself
- *Example sequences*: S1 = <{Aspirin}, {BetaBlock}>, S2 = <{Aspirin, Ibu}, {BetaBlock, Antibiotics}> with $S1 \prec S2$

Sequence mining – analytical goal:

- *Find the maximum sequences in the customer sequence with a user-defined minimum support.*
- Sub task 1: finding sequences with minimum support → large sequences
- Sub task 2: out of them finding the maximum ones
- Let C be a set of customers and S be a sequence. Then c in C supports S if S is contained in S_c with S_c being the sequence of costumer c: $s(S, C) = \frac{|\{c \; with \; S \prec Sc\}|}{|C|}$
- cf. finding large item sets
- *length(s)*: number of items within sequence s
- S maximal if $\nexists \; S' with \; S \prec S' and \; length(S) \leq length(S')$

Sequence mining – Example (ctd.)

- Assume minimum support of 0.4
- We find the following large sequences:
  - S1 = <Aspirin> with support 1 and length 1
  - S2 = <Ibu> with support 0.5 and length 1
  - S3 = <BetaBlock> with support 0.5 and length 1
  - S4 = <AntiBiotics> with support 0.5 and length 1
  - S5 = <Aspirin, BetaBlock> with support 0.5 and length 2
  - S6 = <Aspirin, AntiBiotics> with support 0.5 and length 2
  - S7 = <Aspirin, Ibu, AntiBiotics> with support 0.5 and length 3

Implementations of algorithms available at: http://www.philippe-fournier-viger.com/spmf/

6 Sequence and episode mining

Sequence versus episode mining[6]

- Input data:
  - Sequence mining: transactional set with time stamps
  - Episode mining: stream of time-stamped events
- Pattern structure:
  - Sequence mining: maximum sequences of item sets
  - Episode mining: partly ordered collection of events occuring together
- Distinction between serial and parallel episodes
- Example: s=`<(A,2), (M,3), (A,4), (B,5), (A,8), (M,9), (B,10), (I,12), (A,13), (A,15), (M,16), (B,18), (A,19) >`
- Serial episode: A is always followed by M
- Parallel episode: A and B frequently occur together

[6]Mannila H, Toivonen H, Verkamo IA (1997) Discovery of frequent episodes in event se-quences. Data Mining and Knowledge Discovery 1(3):259–289

Episode mining – definitions:

- Goal: find the neighborhood where potential patterns occur in
- Occurrence must be confined to a segment of the event stream
- → use of a certain size to subdivide the event stream
- Let E be an event set. Assume an event sequence $s \coloneqq < (e_1, t_1), \ldots, (e_n, t_n) >$ where *e_i* are the events and *t_i* the associated time stamps with $t_i \leq t_{i+1}$. Then an episode $\varepsilon \coloneqq (V, \leq, g)$ with v being a set of nodes, ≤ being a partial order and g being a mapping function *g: V → E*.
- Episodes are represented by graphs where edges represent patterns. Edge(A,B) represented a serial pattern between events A and B; if A and B parallel, no edges between A and B exists.

Episode mining – definitions (ctd.)

- Window $w \coloneqq (s, t_s^w, t_e^w) with\ t_s^w < \mathrm{t_n}, \mathrm{t_s^w} > \mathrm{t_1}$
- Window width: $width(w) \coloneqq t_e^w - t_s^w$
- Set $W(s, ws) \coloneqq \{w\ over\ s\ with\ width(w) = ws\}$
- How often does a given episode ε occur for windows of size ws on sequence s? Frequency for this calculated as follows:

$$f(\varepsilon, s, ws) \coloneqq \frac{|\{w\ \in W(s, ws) with\ \varepsilon\ occurs\ in\ w\}|}{|W(s, ws)|}$$

- Define minimum frequency threshold
- Frequent episodes exceed frequency

Episode mining – example (ctd.)

- Window *w=(s, 3, 5)* containing event occurrences `A, M, A`. Overall, s contains 9 windows of size 3, i.e.:
- *W(s,3)* = `{(A,M,A), (M,A,B), (A,B,A), (B,A,M), (A,M,B), M,B,I), (B,I,A), (I,A,A), (A,A,M)}`
- Assume episode $\varepsilon=(\{v1,v2\}, \leq, g)$ with $g(v1)=A$ and $g(v2)=M$. ε is a sequential episode as we are looking for patterns where event A precedes event M.
- $f(\varepsilon,s,ws) = 4/9 \approx 0.44$

Analytical goal: find all frequent episodes!

Contents

7 Summary and outlook

- A lot of data is temporal
- Newly arising scenarios: machining data with time stamps
- Finding series or patterns in such data can be very interesting for analysis and predictions
- Finding patterns, sequence, and episodes already paves the way to process mining (→ Chapter 7)

7 Summary & outlook