Vorlesung

Methodische Grundlagen des Software-Engineering

im Sommersemester 2014

Prof. Dr. Jan Jürjens

TU Dortmund, Fakultät Informatik, Lehrstuhl XIV

Teil 2.2: Data-Mining

v. 04.06.2014

2.2 Data-Mining

**[mit freundlicher Genehmigung basierend
auf einem englischen Foliensatz von
Prof. Dr. Wil van der Aalst (TU Eindhoven)]**

Literatur:

[vdA11] Wil van der Aalst: **Process Mining: Discovery, Conformance and Enhancement of Business Processes**, Springer-Verlag. 2011.
Unibibliothek (6 Exemplare): http://www.ub.tu-dortmund.de/katalog/titel/1332248
(Bei Engpässen kann eine **Kopiervorlage** der relevanten Ausschnitte zur Verfügung gestellt werden.)

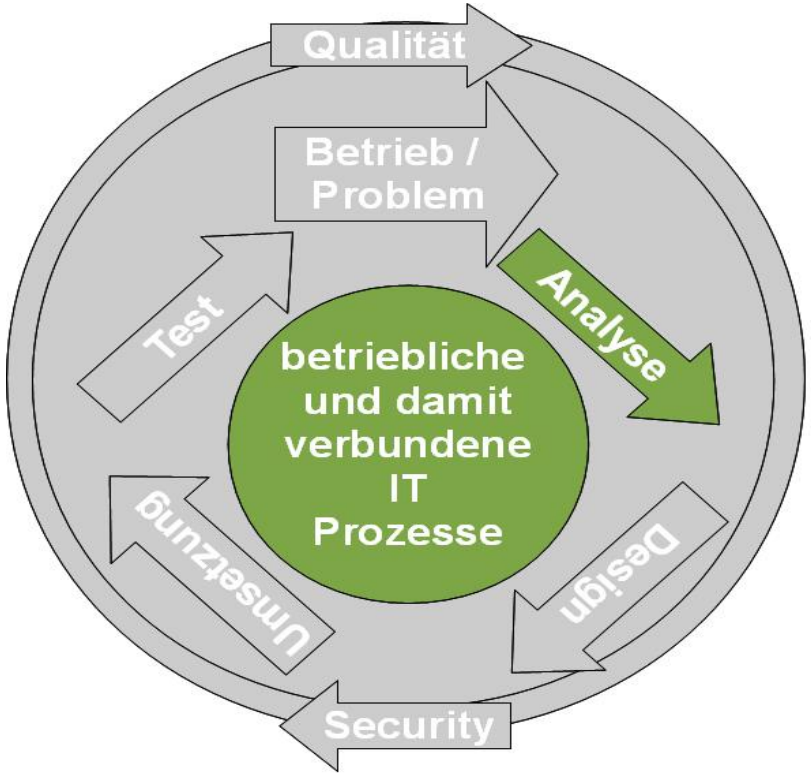
- **Kapitel 3**

[R10] Thomas A. Runkler: **Data Mining: Methoden und Algorithmen Intelligenter Datenanalyse,** Vieweg+Teubner. 2010.
Unibibliothek: http://www.ub.tu-dortmund.de/katalog/titel/1294605

- **Kapitel 8**

Einordnung
2.2: Data-Mining

- Geschäftsprozessmodellierung
- Process-Mining
  - Einführung: Process-Mining
  - Petrinetze
  - Data-Mining
  - Datenbeschaffung
  - Prozessextraktion
  - Konformanzanalyse
  - Mining: Zusätzliche Perspektiven
  - Betriebsunterstützung
  - Werkzeugunterstützung
  - Analysiere „Lasagne Prozesse“
  - Analysiere „Spaghetti Prozesse“
  - Kartographie und Navigation
  - Epilog
- Modellbasierte Entwicklung sicherer Software



Einleitung Data-Mining

- **Vorheriger Abschnitt:** Verifikation von Petrinetzen und Einschränkungen modellbasierter Analyse.
- **Dieser Abschnitt:** „Data-Mining“:
  - Datenbasierte Modellanalyse

Überblick Data-Mining

- Von Datensätzen zu Entscheidungsbäumen durch überwachtes / nicht überwachtes Lernen
- Konfusionsmatrix
- Cluster-Analyse
- Assoziationsregel-Lernen
- Sequence- und Episode-Mining
- Hidden-Markov-Modell und Validierung

Data-Mining vs. Process-Mining

Process-Mining: Data-Mining auf **Prozess-Daten.**

→ Kurzer Hintergrund zu Data-Mining.

Vergleich Data-Mining vs. Process-Mining:

- **Process-Mining**: Ende-zu-Ende Prozesse.
- **Data-Mining**: **Datenbasiert** und nicht prozessbasiert.
- **Qualität** von Data-Mining und Process-Mining bewerten: viele Ähnlichkeiten, aber auch Unterschiede.
- Process-Mining-Techniken können Vorteile aus Erfahrungen im Bereich des Data-Mining ziehen.

Datensatz 1

drinker	smoker	weight	age
yes	yes	120	44
no	no	70	96
yes	no	72	88
yes	yes	55	52
no	yes	94	56
no	no	62	93
...	...		

Daten über 860 kürzlich verstorbene Personen.
Studie über Auswirkung von Alkoholkonsum, Rauchen und Körpergewicht auf Lebenserwartung.

Fragen:
- Welchen Einfluss hat Rauchen und Trinken auf Körpergewicht?
- Trinken Leute, die rauchen, auch?
- Welche Faktoren beeinflussen Lebenserwartung am meisten?
- Gruppen mit ähnlichem Lebensstil identifizierbar?

Datensatz 2

linear algebra	logic	program-ming	operations research	workflow systems	...	duration	result
9	8	8	9	9	...	36	cum laude
7	6	-	8	8	...	42	passed
-	-	5	4	6	...	54	failed
8	6	6	6	5	...	38	passed
6	7	6	-	8	...	39	passed
9	9	9	9	8		38	cum laude
5	5	-	6				
...	...	...	...				

Daten über 420 Studenten, um Verbindungen zwischen Kursnoten und Gesamtleistung im Bachelor zu untersuchen.

Fragen:
- Haben Noten bestimmter Kurse eine hohe Korrelation?
- Welche Wahl treffen Studenten mit Auszeichnung (cum laude)?
- Welche Kurse verzögern den Abschluss signifikant?
- Warum brechen Studenten ab?
- Gruppen mit ähnlichem Lernverhalten identifizierbar?

Datensatz 3

cappuccino	latte	espresso	americano	ristretto	tea	muffin	bagel
1	0	0	0	0	0	1	0
0	2	0	0	0	0	1	1
0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	1	2	0
0	0	0	1	1	0	0	0
...	...	...	...	...	...	...	...

Daten über 240 Bestellungen in einem Café, aufgenommen von der Kasse.

Fragen:
- Welche Produkte werden häufig zusammen gekauft?
- Wann kaufen Kunden bestimmte Produkte?
- Gruppen typischer Kunden charakterisierbar?
- Wie fördert man Verkauf von Produkten mit hoher Gewinnmarge?

Variablen

- Datensatz besteht aus **Instanzen** (Individuen, Entitäten, Fälle, Objekte oder Aufzeichnungen).
- **Variablen**: als Attribute, Features oder Datenelemente bezeichnet.

Zwei Typen:

- Kategorielle Variablen:
  - **Ordinal** (hoch – mittel – niedrig,
    mit Auszeichnung – bestanden – durchgefallen) oder
  - **Nominal** (true – false, rot – pink – grün).
- Numerische Variablen (geordnet, können nicht einfach aufgezählt werden).

Überwachtes Lernen

- **Klassifizierte Daten** (Labeled Data): Jede Instanz durch Response-Variable gekennzeichnet.
- **Ziel**: Erkläre Response-Variable (abhängige Variable) in Form von Predictor-Variablen (unabhängige Variable).
- **Klassifikationstechniken** (z.B.: Lernen mit Entscheidungsbäumen): Setzen kategorielle Response-Variablen voraus.
  Ziel: Instanzen anhand Predictor-Variablen klassifizieren.
- **Regressionstechniken:** Benötigen numerische Response-Variablen.
  Ziel: Zu Daten passende Funktion mit wenigsten Fehlern finden.

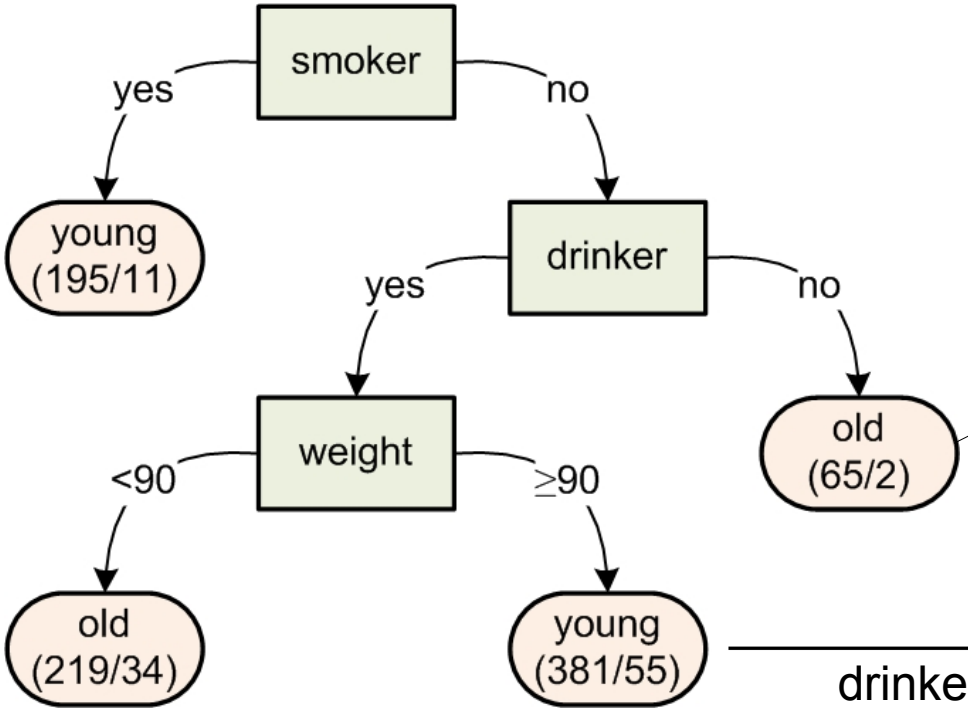
Nicht überwachtes Lernen

- **Nicht überwachtes Lernen** verwendet unlabeled data. Variablen nicht in Response- und Predictor-Variablen unterteilt.

Beispiele:

- Clustering (z.B.: k-means clustering und agglomerative hierarchical clustering).
- Pattern discovery (association rules).

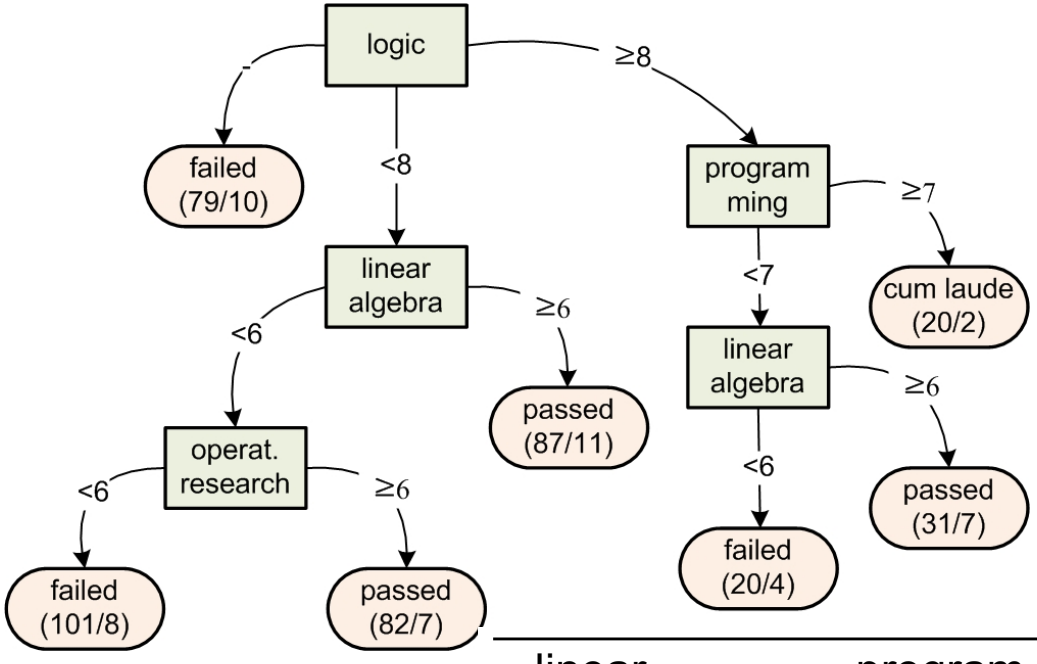
Entscheidungsbaum: Datensatz 1



65 als „old“ vorhergesagt,
davon 2 inkorrekt.

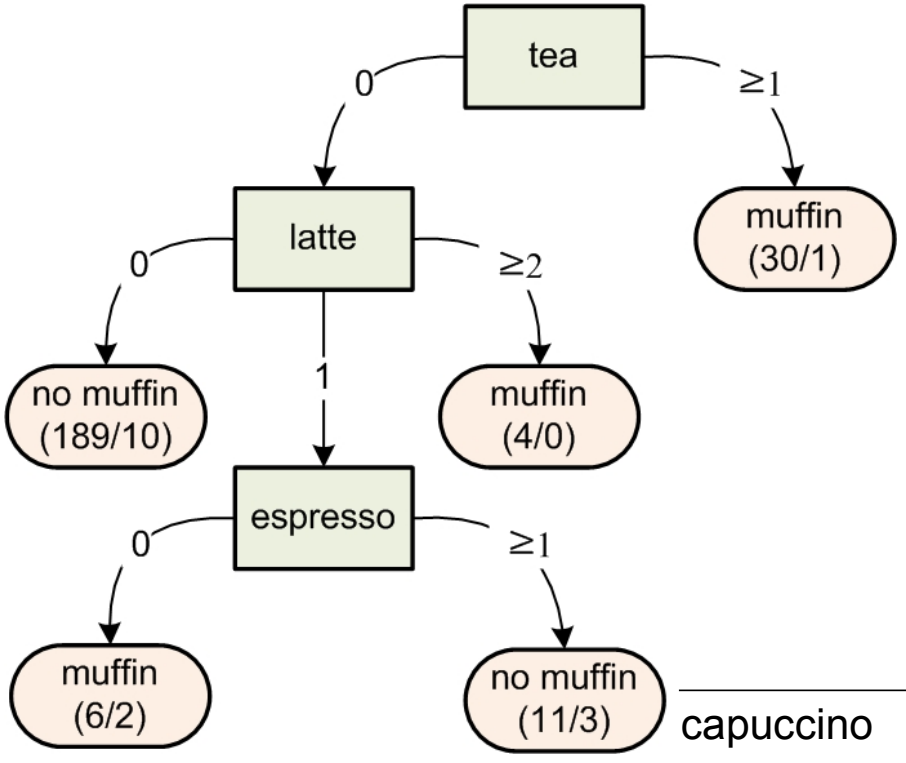
drinker	smoker	weight	age
yes	yes	120	44
no	no	70	96
yes	no	72	88
yes	yes	55	52
no	yes	94	56
no	no	62	93
...	...	...	...

Entscheidungsbaum: Datensatz 2



linear algebra	logic	program-ming	operations research	workflow system	...	duration	result
9	8	8	9	9	...	36	cum laude
7	6	-	8	8	…	42	passed
-	-	5	4	6	…	54	failed
8	6	6	6	5	…	38	passed
6	7	6	-	8	…	39	passed
9	9	9	9	8	…	38	cum laude
5	5	-	6	6	…	52	failed
...	...	...	...	...	...	...	...

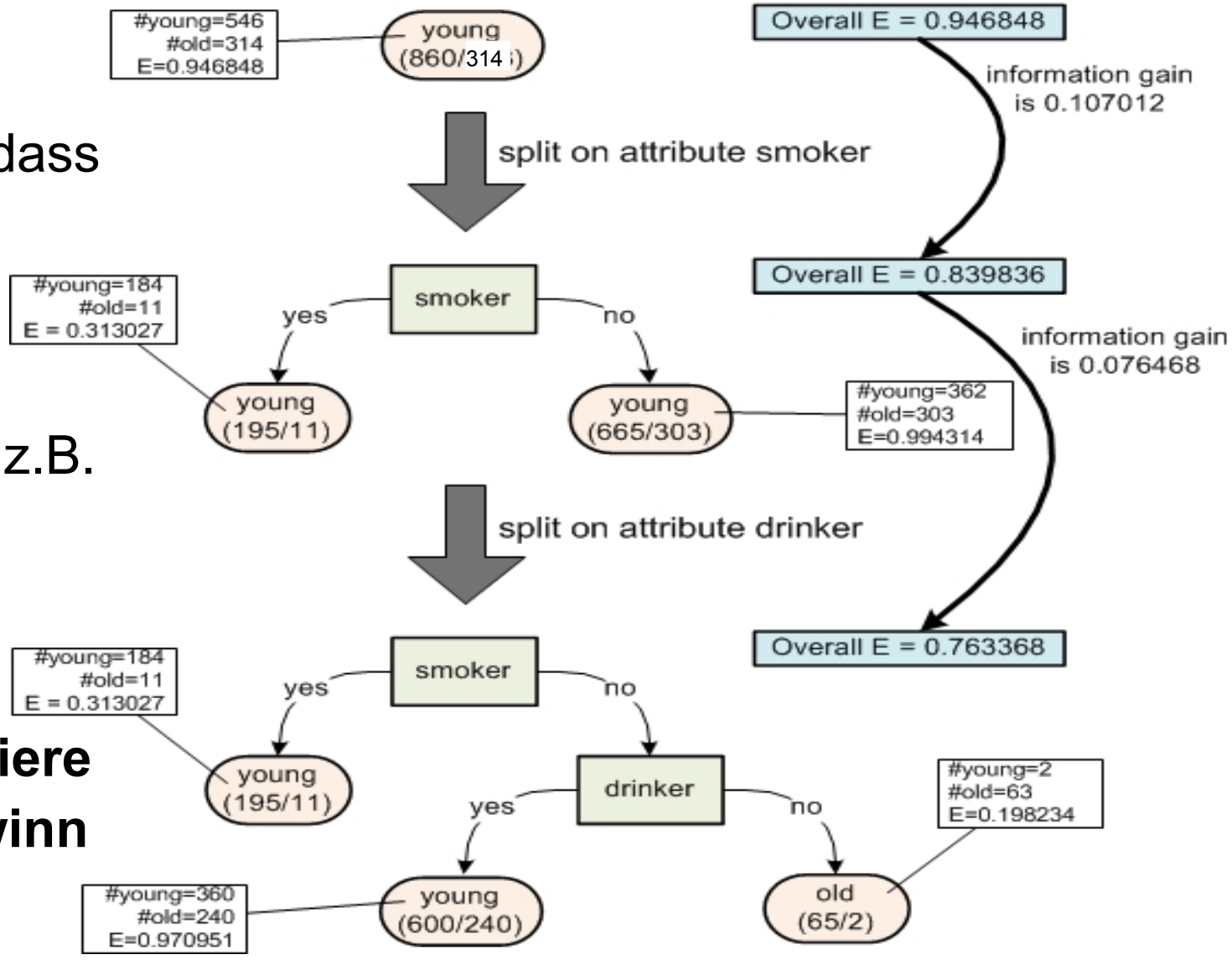
Entscheidungsbaum: Datensatz 3



capuccino	latte	espresso	americano	ristretto	tea	muffin	bagel
1	0	0	0	0	0	1	0
0	2	0	0	0	0	1	1
0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	1	2	0
0	0	0	1	1	0	0	0
...	...	...	...	...	...	...	...

Entscheidungsbaum generieren: Grundidee

- Teile Menge der Instanzen auf, sodass **Variation** in jeder Teilmenge **kleiner** wird.
- Variation messen z.B. durch **Entropie**.
- **Minimiere** durchschnittliche **Entropie**; **maximiere Informationsgewinn** pro Schritt.



Entropie

Entropie *E*: **informationstheoretisches Maß** für „Chaos“ in einer Multimenge:

- Benötigte Anzahl Bits zur **Kodierung** der Multimenge (*E* >= 0).
- Häufiges Ziel: Minimierung von *E* anstreben.

Annahme: Element v_i in Multimenge c_i-mal enthalten; Multimenge hat n Elemente.

Einzelentropie: $- \log_2(p_i)$, wobei $p_i = c_i/n$.

Gesamt-Entropie: gewichtetes Mittel der Einzel-Entropien: $E = -\sum_{i=1}^{k} (p_i \log_2(p_i))$

Beispiel 1: Alle Elemente haben denselben Wert ($p_1=1$) → $E = - \log_2 1 = 0$

Beispiel 2:

- 4 mögliche Elemente V = {a,b,c,d}
- Sequenz: *bacaabadabadcaba...*
- Relative Häufigkeiten: $p_1 = 0{,}5$; $p_2 = 0{,}25$; $p_3 = 0{,}125$; $p_4 = 0{,}125$
- *E* = 1,75

Iterative Dichotomiser 3 (ID3): Entscheidungsbaum generieren

Anwendung:

- Bei großer Datenmenge viele verschiedene Attribute von Bedeutung.
  → Entscheidungsbaum ohne große Berechnungen generieren.

Algorithmus:

- Iterativ
- Benutzt **Entropie** zur Bestimmung von Baum-Knoten.
- Abbruch, falls jedem Blattknoten eine Klassifikation zugeordnet.

Eingabe:

- Menge zu klassifizierender Objekte, Wurzelknoten, Menge noch zu vergebener Merkmale

Ausgabe:

- Struktur mit Tupeln (Entscheidungsbaum)

Iterative Dichotomiser 3 (ID3) Algorithmus: Informell

Aufruf von ID3 ausgehend von

- Menge {1,...,n} der Indizes aller zu klassifizierender Objekte,
- Wurzelknoten des Entscheidungsbaums,
- Menge {1,...,p} aller noch zu vergebenden Merkmale.

Am aktuellen Knoten gehören **alle Daten derselben Klasse → Abbruch!**

Sonst:

- Erwarteten **Informationsgewinn *gi*** für jedes zu vergebendes Merkmal berechnen.
- Merkmal j bestimmen, der höchsten Informationsgewinn liefert.
- Anhand Werte des **Gewinnermerkmals *j*** Datensatz in **disjunkte Teilmengen** zerlegen.
- Für jede **nichtleere Teilmenge** neuen Knoten anhängen.
- Für jeden angehängten Knoten rekursiv zugehörigen Teilbaum berechnen.

Ausgabe: Entscheidungsbaum bzw. Struktur mit Tupeln

Iterative Dichotomiser 3 (ID3) Algorithmus: Pseudocode

Gegeben: $X=\{x_1,\ldots,x_n\}\subset\{1,\ldots,v_1\}\times\ldots\times\{1,\ldots,v_p\}$, $y=\{y_1,\ldots,y_n\}\subset\{1,\ldots,c\}$

Aufrufen: $ID3(\{1,\ldots,n\},Wurzel,\{1,\ldots,p\})$

Prozedur ID3(*I,N,K*)

1. Wenn alle *y(I)* gleich dann abbrechen
2. Berechne Informationsgewinn $g_j((X(I),y(I)))=E(X)-\sum_{j\in y}(\frac{|X_j|}{|X|})E(X_j)\forall\, j\in K$

   X: Datensatz, E(X): Entropie im Datensatz, c: Klassifikation
   X_j: Untermenge von X, $|X|$: Mächtigkeit von X, $|X_j|$: Mächtigkeit von X_j
3. Bestimme Gewinnermerkmal $i=argmax\{g_j((X(I),y(I)))\}$
4. Zerlege *I* in v_i disjunkte Teilmengen

   $I_j=\{k\in I|x_k^{(i)}=k\}\; j=1,\ldots,v_i$
5. für *j* mit $I_j\neq\{\}$

   - Generiere neuen Knoten N_j und hänge ihn an *N*
   - Aufrufen: $ID3(I_j,N_j,I\setminus\{i\})$

Bedingte Entropie:

- Maß über den Wert einer Zufallsvariable, welche verbleibt, nachdem das Ergebnis einer anderen Zufallsvariable bekannt wird.

$$H(C|A_t) = -\sum_{i=1}^{k_t} p_i \sum_{j=1}^{k} \left(\frac{x_{i,j}}{x_i}\right) \log_2 \left(\frac{x_{i,j}}{x_i}\right)$$

Größe:

	giftig	essbar
klein	1	2
groß	0	2

$x_{klein} = 3 \qquad p_{klein} = \frac{3}{5} = 0.6$

$x_{groß} = 2 \qquad p_{groß} = \frac{2}{5} = 0.4$

$$H(C|A_{Größe}) = -\left[0.6\left(\frac{1}{3}\log_2\left(\frac{1}{3}\right) + \frac{2}{3}\log_2\left(\frac{2}{3}\right)\right) + 0.4\left(\frac{0}{2}\log_2\left(\frac{0}{2}\right) + \frac{2}{2}\log_2\frac{2}{2}\right)\right]$$

$$= -\left[0.4\left(-\log_2(3) + \frac{2}{3}\right) + 0\right] \approx \mathbf{0.4562}$$

Iterative Dichotomiser 3 (ID3)
Beispiel (2)

- **Punkte:**

	giftig	essbar
ja	1	1
nein	0	3

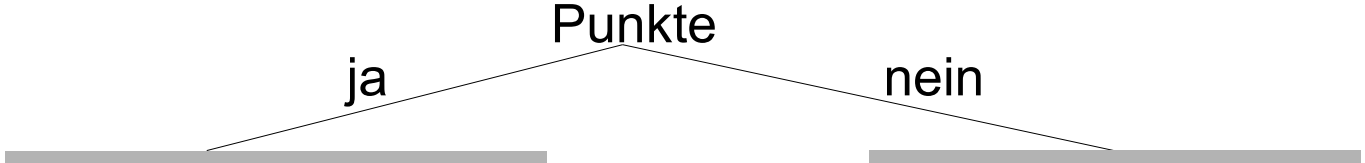
$x_{ja} = 2$ $\quad p_{ja} = \frac{2}{5} = 0.4$

$x_{nein} = 3$ $\quad p_{nein} = \frac{3}{5} = 0.6$

$$H(C|A_{Punkte}) = -[0.4(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}) + 0.6(\frac{0}{3}\log_2\frac{0}{3} + \frac{3}{3}\log_2\frac{3}{3})]$$
$$-[0.4(-1)+0] = \mathbf{0.4}$$

→ **Bedingte Entropie minimal** für das Attribut Punkte.

- Entscheidungsbaum nach **Rekursionsebene I:**



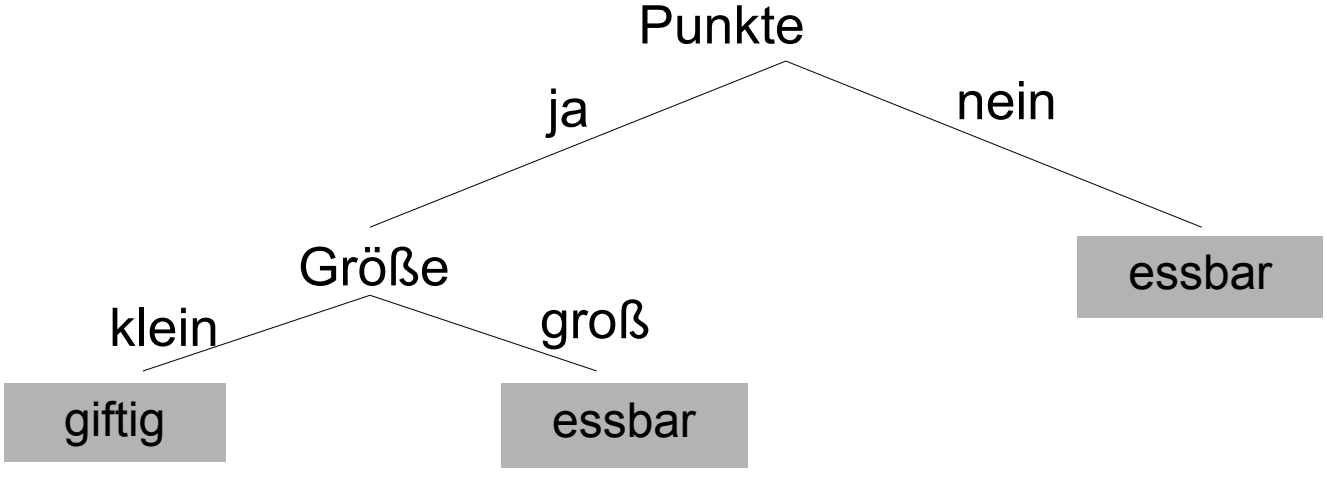
Größe	**Klasse**
klein	giftig
groß	essbar

Größe	**Klasse**
klein	essbar
klein	essbar
groß	essbar

Rekursionsstufe II:

- Objektsammlung M_{nein} besteht aus 3 Beispielen, die alle in die Klasse der essbaren Pilze gehören. → M_{nein} trivial
- Objektsammlung M_{ja} trivial, der Entscheidungsbaum besteht nur aus Blättern mit Label giftig bzw. essbar.

Entscheidungsbaum nach Rekursionsebene II:

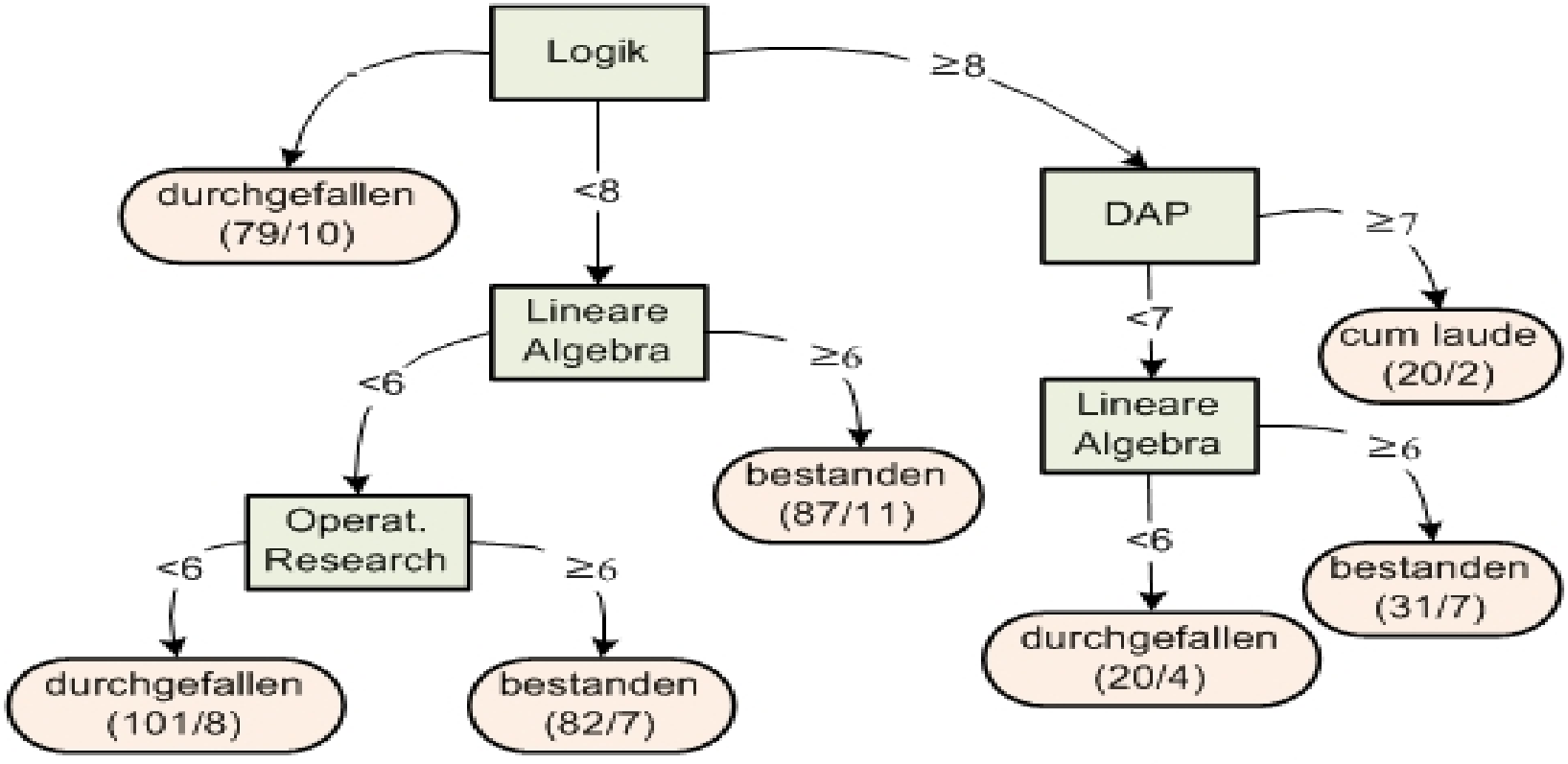


http://www2.informatik.uni-hamburg.de/wsv/teaching/praktika/GwvPRAK_WiSe10/aufg05_ID3_script.pdf

Überblick Data-Mining

- Von Datensätzen zu Entscheidungsbäumen durch überwachtes / nicht überwachtes Lernen
- Konfusionsmatrix
- Cluster-Analyse
- Assoziationsregel-Lernen
- Sequence- und Episode-Mining
- Hidden-Markov-Modell und Validierung

Konfusionsmatrix



reale Bewertung		*vorhergesagte Bewertung*		
		durchge-fallen	bestanden	cum laude
	durchge-fallen	178	22	0
	bestanden	21	175	2
	cum laude	1	3	18

Konfusionsmatrix: Metriken

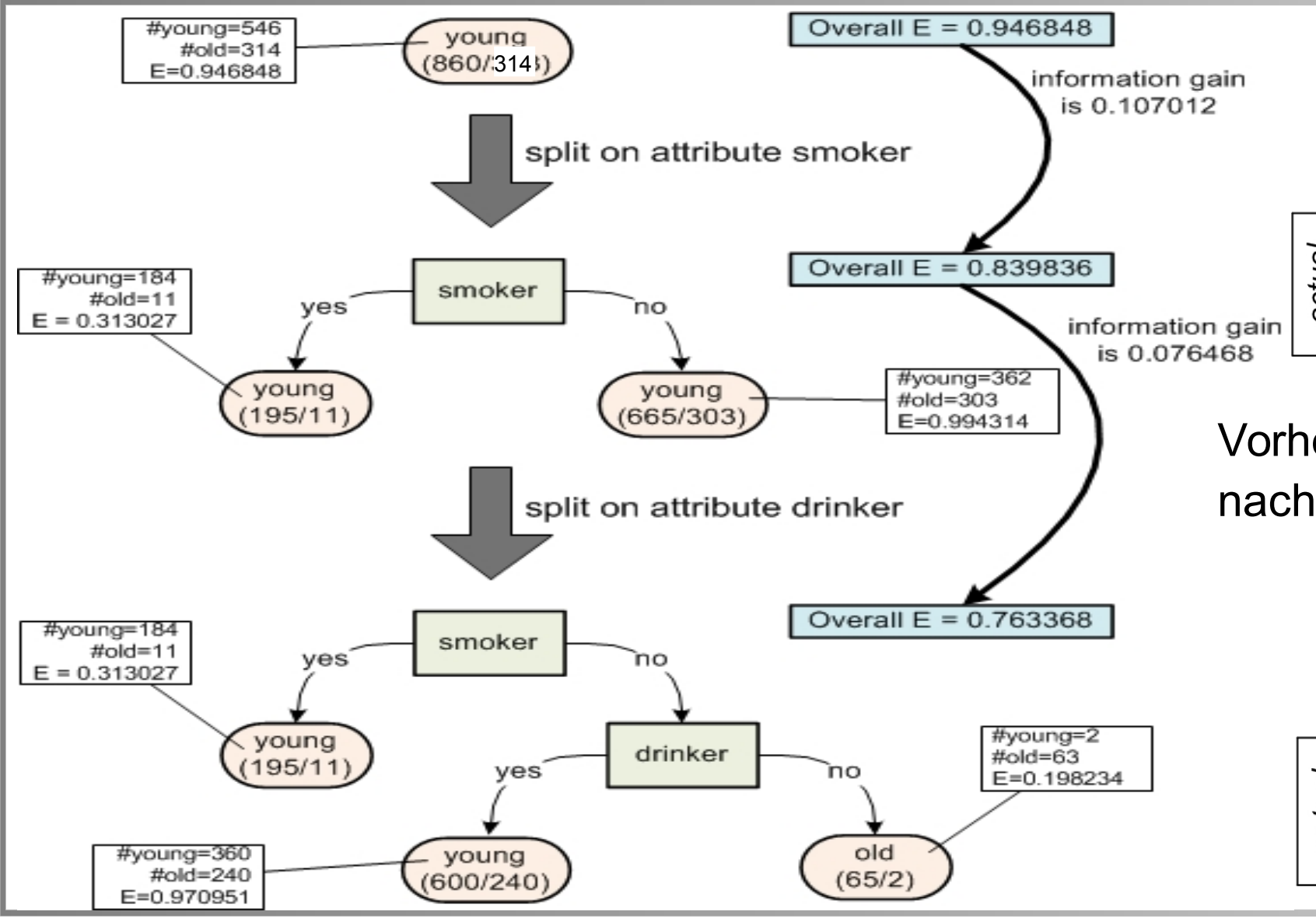
Bsp. für 2 Klassen + und -

		vorhergesagte Klasse +	*vorhergesagte Klasse* -	
reale Klasse	+	*tp*	*fn*	*p*
	-	*fp*	*tn*	*n*
		p'	*n'*	*N*

Name	*Formel*
Error	*(fp+fn)/N*
accuracy	*(tp+tn)/N*
tp-Rate	*tp/p*
fp-Rate	*fp/n*
precision	*tp/p'*
recall	*tp/p* (= tp-Rate!)

tp: Anzahl true positives; korrekterweise als positiv klassifiziert.
fn: Anzahl false negatives; als negativ klassifiziert, aber positiv.
fp: Anzahl false positives; als positiv klassifiziert, aber negativ.
tn: Anzahl true negatives; korrekterweise als negativ klassifiziert.

Konfusionsmatrix: Beispiel



		predicted class: young	predicted class: old
actual class	young	546	0
	old	314	0

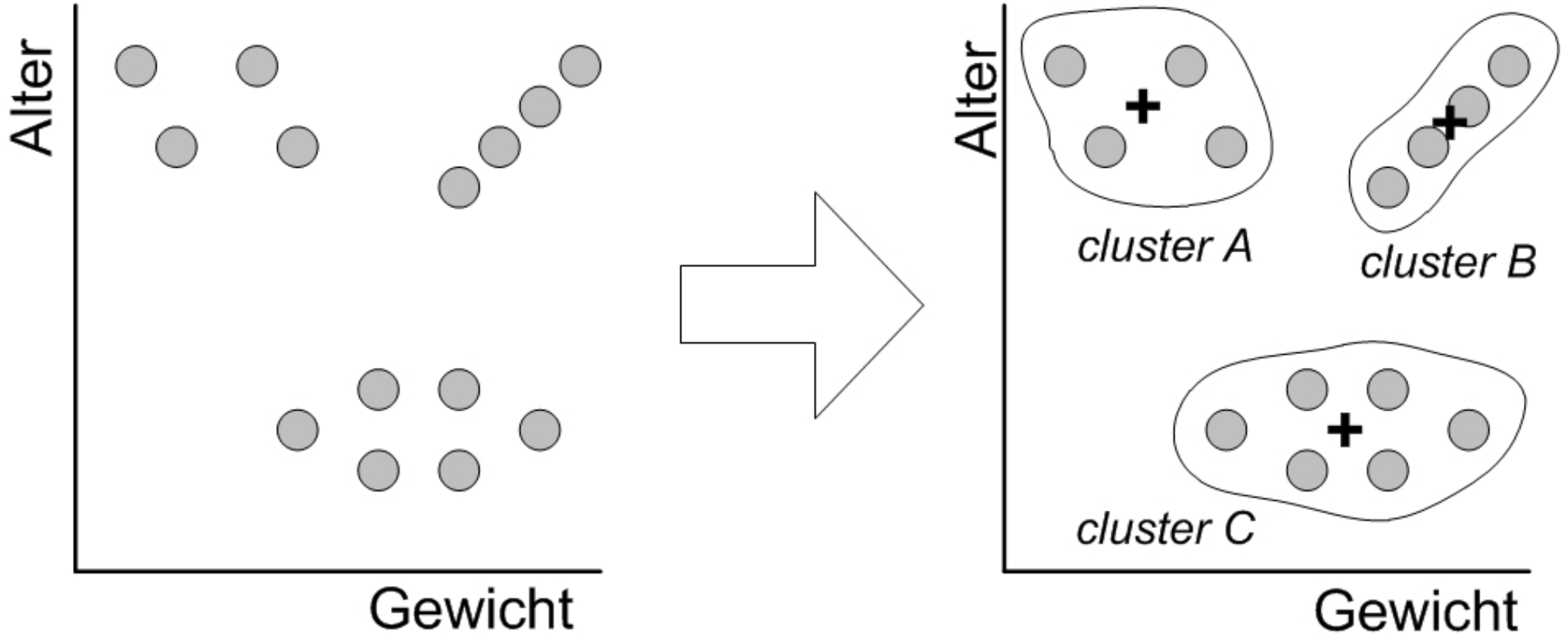
Vorhersage vor bzw. nach Aufsplitten.

		predicted class: young	predicted class: old
actual class	young	544	2
	old	251	63

Überblick Data-Mining

- Von Datensätzen zu Entscheidungsbäumen durch überwachtes / nicht überwachtes Lernen
- Konfusionsmatrix
- Cluster-Analyse
- Assoziationsregel-Lernen
- Sequence- und Episode-Mining
- Hidden-Markov-Modell und Validierung

Cluster-Analyse



+: Schwerpunkt eines Clusters

K-Means-Cluster-Analyse mittels Lloyd Algorithmus: Informell

Von **Stuart P. Lloyd** 1982 vorgestellt (benannt: k-Means).

- Anzahl *K* zu ermittelnder Cluster vorher festlegen.
- Start (*i=0*): Positionen der **Clusterschwerpunkte** zufällig initialisieren.
- Objekte **nächstgelegenen Schwerpunkten** zuordnen. (*i=1*)
- Bei jeder Iteration *i* Schwerpunkt und nächstliegende Kandidaten neu berechnen.
- Dies Wiederholen bis **Summe quadratischer Distanz** einzelner Objekte zu ihrem jeweiligen Clusterschwerpunkt über alle Cluster ein **Minimum** erreicht.

→ Mathematische Darstellung:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} \| \overrightarrow{x_n} - \overrightarrow{\mu_k} \|^2$$

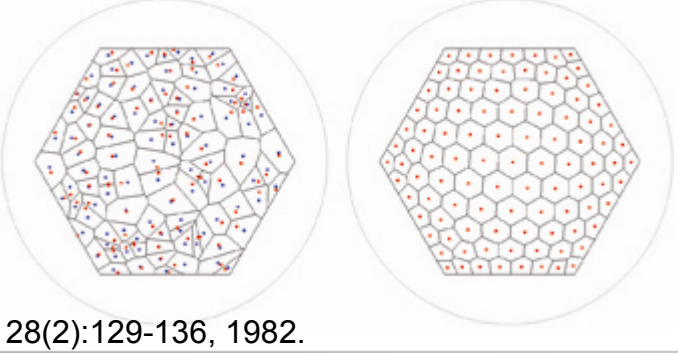
- $\overrightarrow{x_n}$ Datensätze und $\overrightarrow{\mu_k}$ Schwerpunkte der Cluster.
- **Entscheidungskriterium** für Cluster-Zugehörigkeit der Testobjekte: Abstände der Testvektoren von Clusterschwerpunkten.

k-Means-Cluster-Analyse mittels Lloyd Algorithmus: Pseudocode

k-MEANS(P, k):

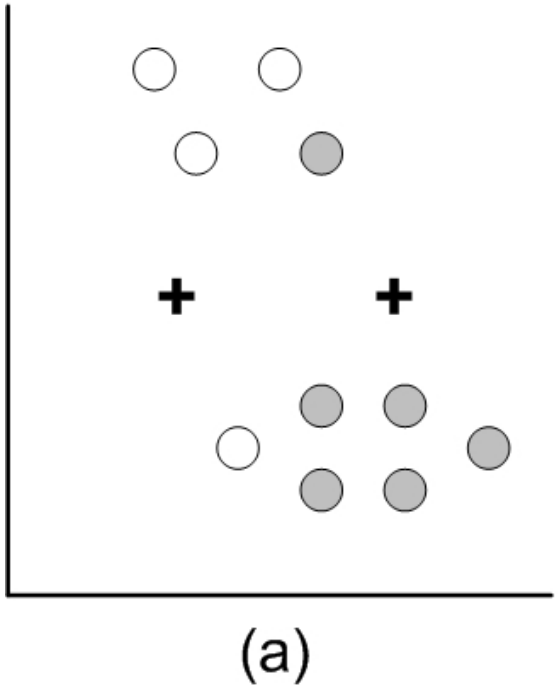
1: Wähle Punkte $C = \{c_1, \ldots, c_k\}$ zufällig gleichverteilt
2: **repeat**
3: Assoziiere jeden Punkt aus P mit dem Zentrumspunkt c_j mit dem geringsten Abstand, um eine Partitionierung $P_1, P_2, \ldots, P_k$ zu erhalten
4: Berechne für jede Teilmenge P_j den Zentroid und verwende diese Menge von Zentroiden als neue Menge von Zentrumspunkten
5: **until** Menge der Zentrumspunkte ändert sich nicht mehr

- Beginn mit **zufälligen Zentrumspunkten.**
- Zur **Verringerung der Gesamtfehler** der **Approximation** neue Punkte in 3. und 4. wählen.



Stuart P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory,* 28(2):129-136, 1982.

k-Means-Clustering: Anzahl *k* von Clustern a-priori festgelegt.

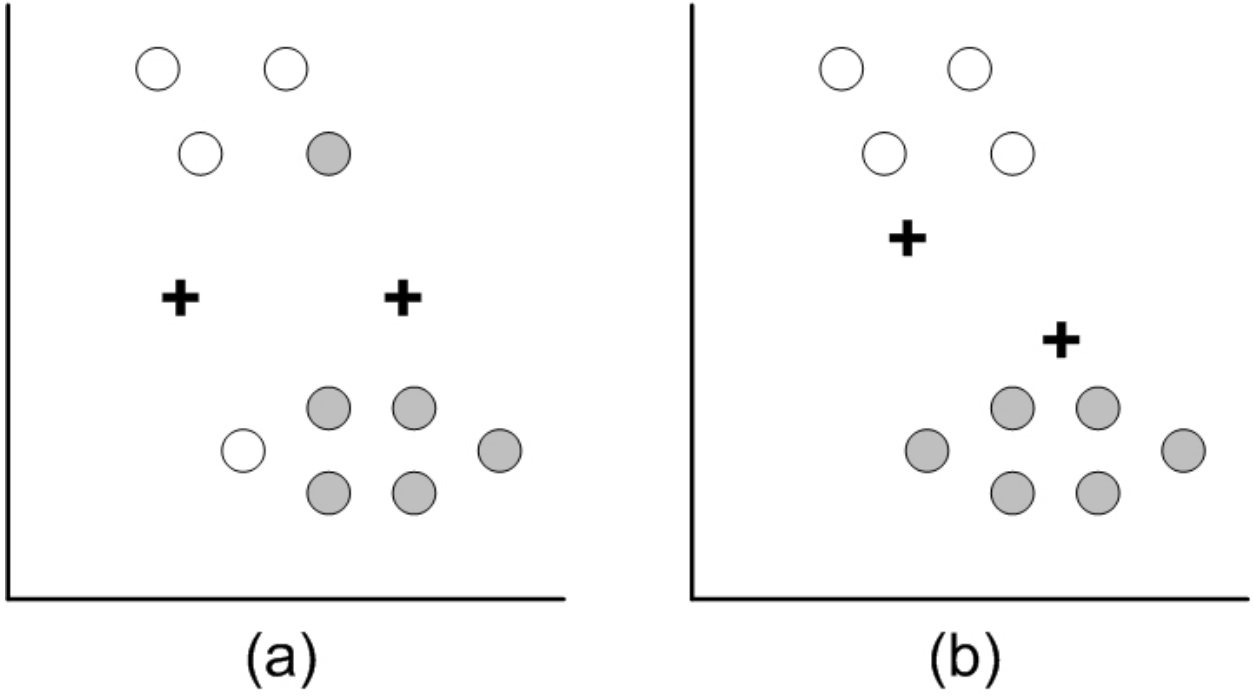


k-Means-Algorithmus veranschaulicht für *k*=2:

(a) „Schwerpunkte“ zufällig setzen; Punkte nächstgelegenen Schwerpunkten zuordnen.

k-Means-Cluster-Analyse
Beispiel

k-Means-Clustering: Anzahl *k* von Clustern a-priori festgelegt.



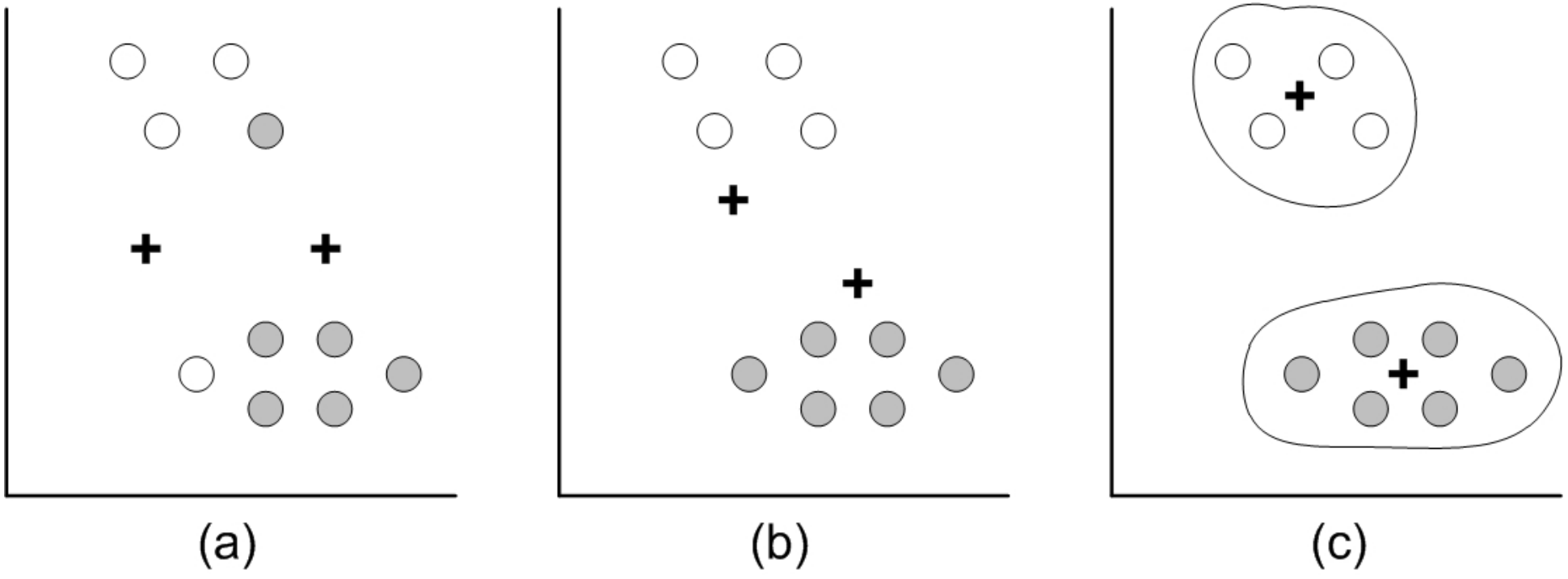
k-Means-Algorithmus veranschaulicht für *k*=2:

(a) „Schwerpunkte“ zufällig setzen; Punkte nächstgelegenen Schwerpunkten zuordnen.

(b) Schwerpunkte neu berechnen und Punkte neu zuordnen.

k-Means-Cluster-Analyse
Beispiel

k-Means-Clustering: Anzahl *k* von Clustern a-priori festgelegt.



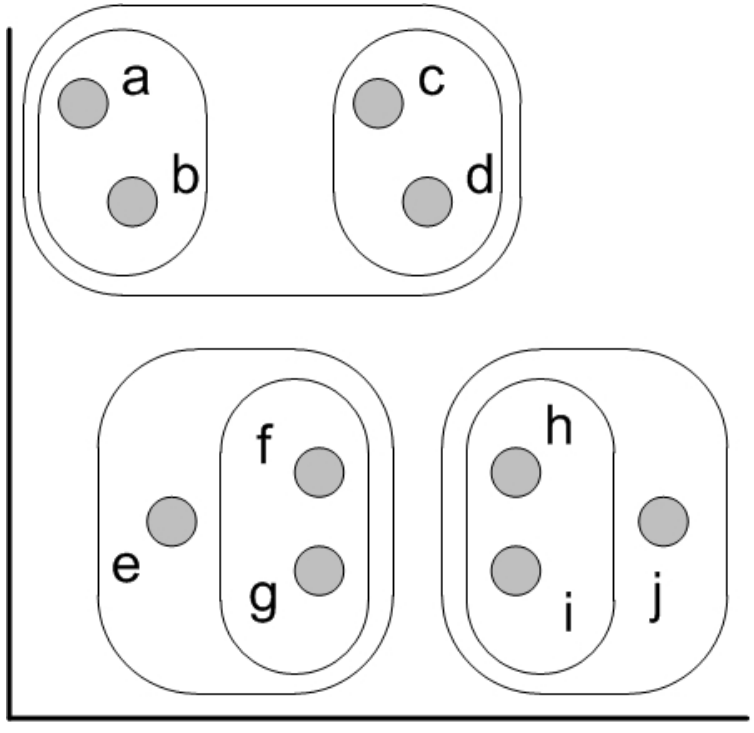
k-Means-Algorithmus veranschaulicht für *k*=2:

(a) „Schwerpunkte“ zufällig setzen; Punkte nächstgelegenen Schwerpunkten zuordnen.

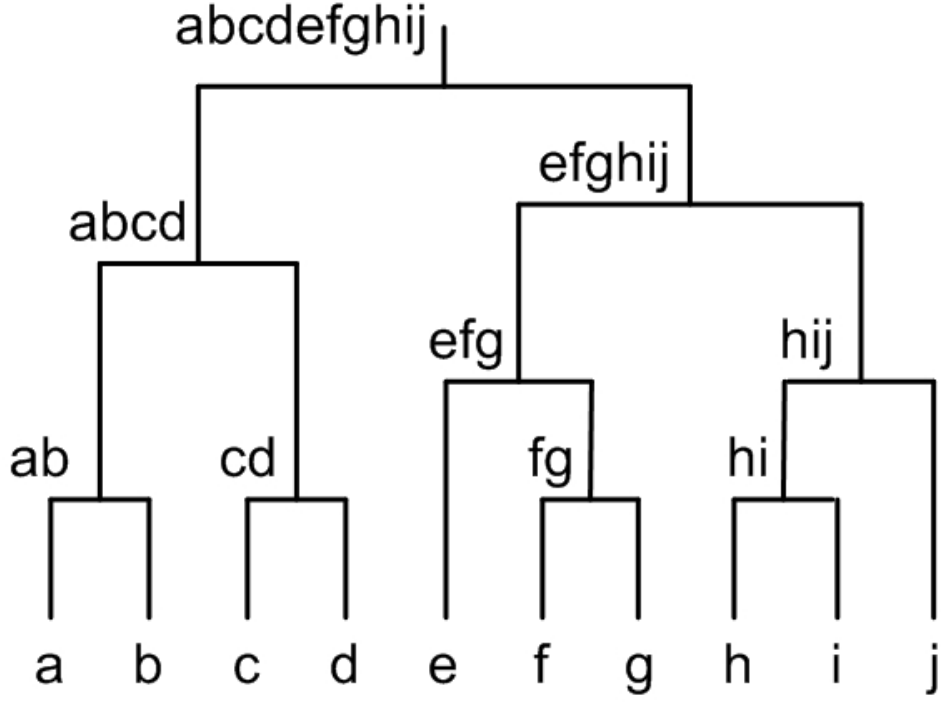
(b) Schwerpunkte neu berechnen und Punkte neu zuordnen.

(c) Fixpunkt erreicht.

Agglomerative Hierarchische Cluster-Analyse



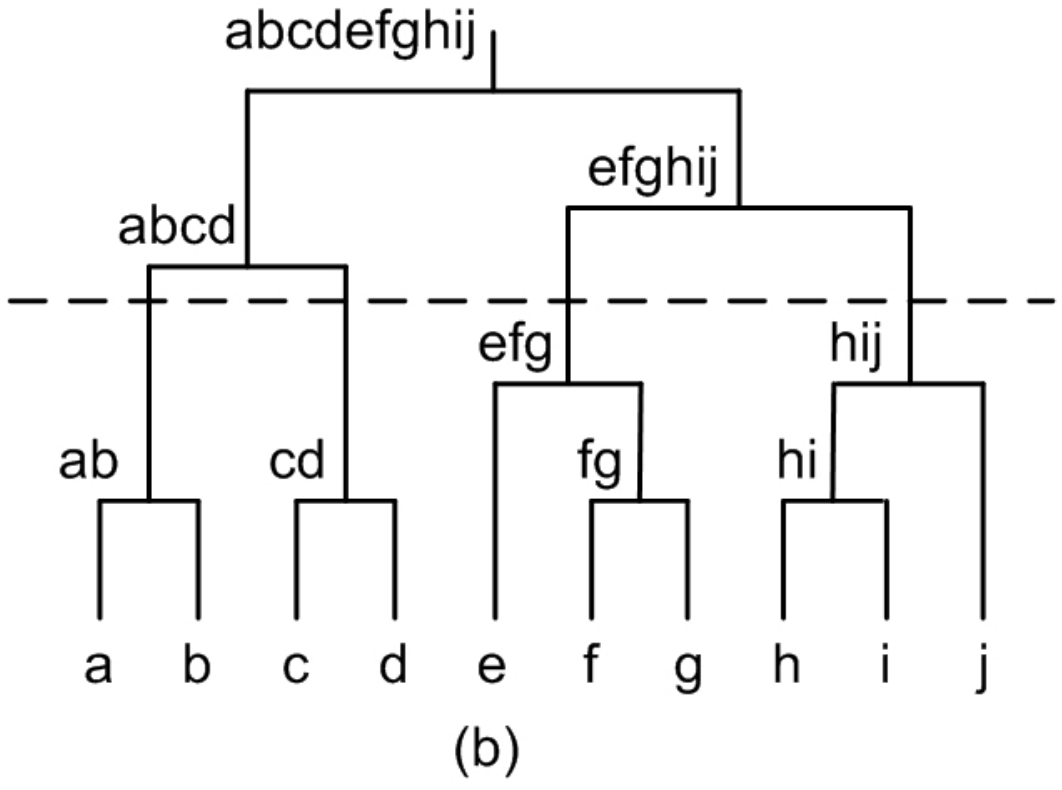
(a)



(b)

(a) Agglomerative hierarchische Cluster-Analyse: **Bottom-up**-Vorgehen.
(b) Visualisierung durch Dendrogramm.

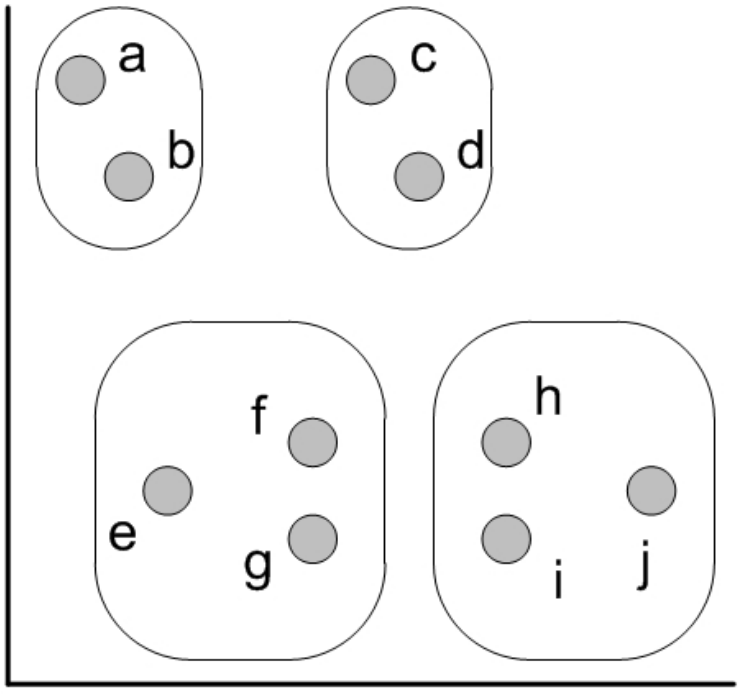
Ebenen bei Agglomerativer Hierarchischer Cluster-Analyse



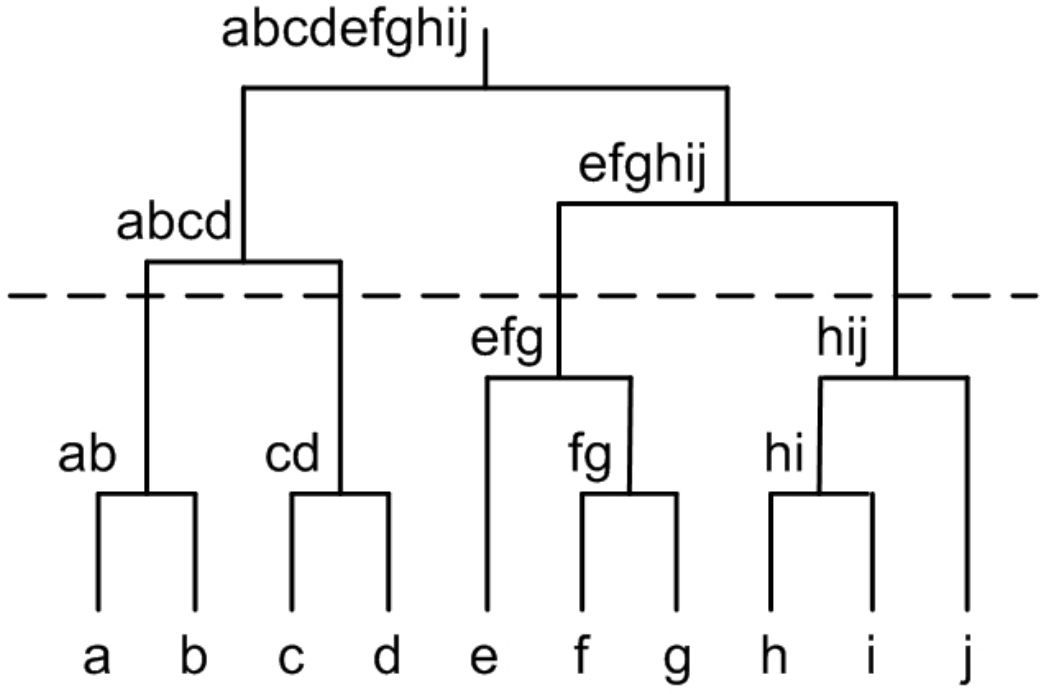
(b)

(b) **Horizontale Linie** in Dendrogramm korrespondiert zu konkretem Clustering auf bestimmter **Abstraktionsebene**.

Ebenen bei Agglomerativer Hierarchischer Cluster-Analyse



(a)



(b)

(b) **Horizontale Linie** in Dendrogramm korrespondiert zu konkretem Clustering auf bestimmter **Abstraktionsebene**.
(a) Aus dem Dendrogramm resultierende Cluster.

Überblick Data-Mining

- Von Datensätzen zu Entscheidungsbäumen durch überwachtes / nicht überwachtes Lernen
- Konfusionsmatrix
- Cluster-Analyse
- Assoziationsregel-Lernen
- Sequence- und Episode-Mining
- Hidden-Markov-Modell und Validierung

Ziel: Regeln der Form “IF X THEN Y” lernen: $X \Rightarrow Y$

Definiere Maße für **Relevanz / Gültigkeit / Aussagekraft** der Regel:

$$support(X \Rightarrow Y) = N_{X \wedge Y} / N$$

$$confidence(X \Rightarrow Y) = N_{X \wedge Y} / N_X$$

$$lift(X \Rightarrow Y) = \frac{N_{X \wedge Y} / N}{(N_X / N)\ (N_Y / N)} = \frac{N_{X \wedge Y}\ N}{N_X\ N_Y}$$

(N_x: Anzahl Datensätze, die alle Eigenschaften in X erfüllen).

Welche Bedeutung könnten hohe / niedrige Werte haben ?

Assoziationsregel-Lernen: Maße und Bedeutung

$$support(X \Rightarrow Y) = N_{X \wedge Y} / N$$
$$confidence(X \Rightarrow Y) = N_{X \wedge Y} / N_X$$
$$lift(X \Rightarrow Y) = \frac{N_{X \wedge Y} / N}{(N_X / N)\ (N_Y / N)} = \frac{N_{X \wedge Y}\ N}{N_X\ N_Y}$$

- **Support**: So hoch wie möglich (oft niedrig). Niedriger Support: relativ geringe Fallzahl, evtl. „zufällig“ geltende Regel.
- **Confidence**: Immer <=1 (da X∧Y => X). Möglichst nahe an 1.
- **Lift**:
  - **Hoch**: positive Korrelation.
  - **Niedrig**: negative Korrelation.
  - **Nahe 1**: Unabhängigkeit (bei probabilistisch unabhängigen Ereignissen X, Y gilt p(X∧Y)=p(X)*p(Y)).

Beispiel: Einkaufswagen-Analyse

cappuccino	latte	espresso	americano	ristretto	tea	muffin	bagel
1	0	0	0	0	0	1	0
0	2	0	0	0	0	1	1
0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	1	2	0
0	0	0	1	1	0	0	0
...	...	...	...	...	...	...	...

Zwei Beispiel-Hypothesen:

$$tea \wedge latte \Rightarrow muffin$$

→ Leute, die *tea* und *latte* bestellen, bestellen auch *muffins*.

$$tea \Rightarrow muffin \wedge bagel$$

Beispiel: tea $\wedge$ latte => muffin

$tea \wedge latte \Rightarrow muffin$, d.h.: $X = tea \wedge latte$ und $Y = muffin$

$$support(X \Rightarrow Y) = N_{X \wedge Y}/N = N_{tea \wedge latte \wedge muffin}/N = 15/240 = 0.0625$$

$$confidence(X \Rightarrow Y) = N_{X \wedge Y}/N_X = N_{tea \wedge latte \wedge muffin}/N_{tea \wedge latte} = 15/20 = 0.75$$

$$lift(X \Rightarrow Y) = \frac{N_{X \wedge Y}\, N}{N_X\, N_Y} = \frac{N_{tea \wedge latte \wedge muffin}\, N}{N_{tea \wedge latte}\, N_{muffin}} = \frac{15 \times 240}{20 \times 40} = 4.5$$

Brute-force-Algorithmus

Definition: Für gegebenen *support*-Schwellwert *minsup*: Menge *Z* heißt ***frequent item-set***, wenn $N_z / N >= minsup$.

Assoziationsregeln wie folgt **generierbar**:

1) Generiere ***frequent item-sets***: alle Mengen *Z* sodass N_z / N größer als gegebener Schwellwert für *support* und |Z| >=2.
2) Für jedes *frequent item-set Z* betrachte Partition in nicht-leere Teilmengen *X*, *Y*.
   Behalte Regeln *X* => *Y*, für die ***confidence*** gegebenen Schwellwert überschreitet oder gleich ist.

Beobachtung

Beobachtung:

Unter welcher Voraussetzung ist **Teilmenge** einer ***frequent item-set*** ebenfalls eine ***frequent item-set*** ?

Beobachtung

Beobachtung: Jede nicht-leere **Teilmenge** einer ***frequent item-set*** ist ebenfalls ***frequent***: Durch **Teilmengen-**Bildung kann sich Menge der zu erfüllenden **Eigenschaften** allenfalls **verringern** (nicht vergrößern), daher kann sich der ***support*** allenfalls **vergrößern** (nicht verringern), da es einfacher wird, alle Eigenschaften zu erfüllen.

1. If an item-set is *frequent* (i.e., an item-set with a support above the threshold), then all of its non-empty subsets are also frequent. Formally, for any pair of non-empty item-sets X, Y: if $Y \subseteq X$ and $N_X/N \geq minsup$, then $N_Y/N \geq minsup$.
2. If, for any k, I_k is the set of all frequent item-sets with cardinality k and $I_l = \emptyset$ for some l, then $I_k = \emptyset$ for all $k \geq l$.

→ **Maximale *frequent item-sets*** ausgehend von **1-elementigen** *frequent item-sets* generierbar.

Optimierte Generierung von *frequent item-sets* mit o.g. Idee

1. Create I_1. This is the set of singleton frequent item-sets, i.e., item-sets with a support above the threshold *minsup* containing just one element.
2. $k := 1$
3. If $I_k = \emptyset$, then output $\bigcup_{i=1}^{k} I_i$ and end. If $I_k \neq \emptyset$, continue with the next step.
4. Create C_{k+1} from I_k. C_{k+1} is the candidate set containing item-sets of cardinality $k+1$. Note that one only needs to consider elements that are the union of two item-sets A and B in I_k such that $|A \cap B| =$ k-1 and $|A \cup B| = k+1$. (wg. o.g. Beobachtung)
5. For each candidate frequent item-set $c \in C_{k+1}$: examine all subsets of c with k elements; delete c from C_{k+1} if any of the subsets is not a member of I_k. (wg. o.g. Beobachtung)
6. For each item-set c in the pruned candidate frequent item-set C_{k+1}, check whether c is indeed frequent. If so, add c to I_{k+1}. Otherwise, discard c.
7. $k := k+1$ and return to Step 3.

Überblick Data-Mining

- Von Datensätzen zu Entscheidungsbäumen durch überwachtes / nicht überwachtes Lernen
- Konfusionsmatrix
- Cluster-Analyse
- Assoziationsregel-Lernen
- Sequence- und Episode-Mining
- Hidden-Markov-Modell und Validierung

Sequence-Mining

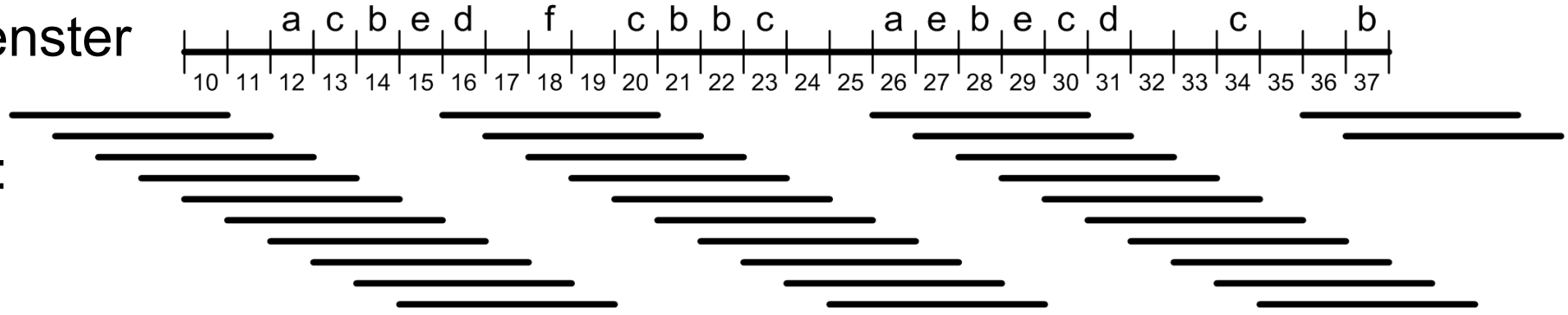
customer	seq. number	timestamp	items
Wil	1	02-01-2011:09.02	{*cappuccino*}
	2	03-01-2011:10.06	{*espresso,muffin*}
	3	05-01-2011:15.12	{*americano,cappuccino*}
	4	06-01-2011:11.18	{*espresso,muffin*}
	5	07-01-2011:14.24	{*cappuccino*}
	6	07-01-2011:14.24	{*americano,cappuccino*}
Mary	1	30-12-2010:11.32	{*tea*}
	2	30-12-2010:12.12	{*cappuccino*}
	3	30-12-2010:14.16	{*espresso,muffin*}
	4	05-01-2011:11.22	{*bagel,tea*}
Bill	1	30-12-2010:14.32	{*cappuccino*}
	2	30-12-2010:15.06	{*cappuccino*}
	3	30-12-2010:16.34	{*bagel,espresso,muffin*}
	4	06-01-2011:09.18	{*ristretto*}
	5	06-01-2011:12.18	{*cappuccino*}
...	...	...	...

$$X = \langle \{cappuccino\}, \{espresso\} \rangle$$

$$Y = \langle \{cappuccino\}, \{espresso\}, \{latte, muffin\} \rangle$$

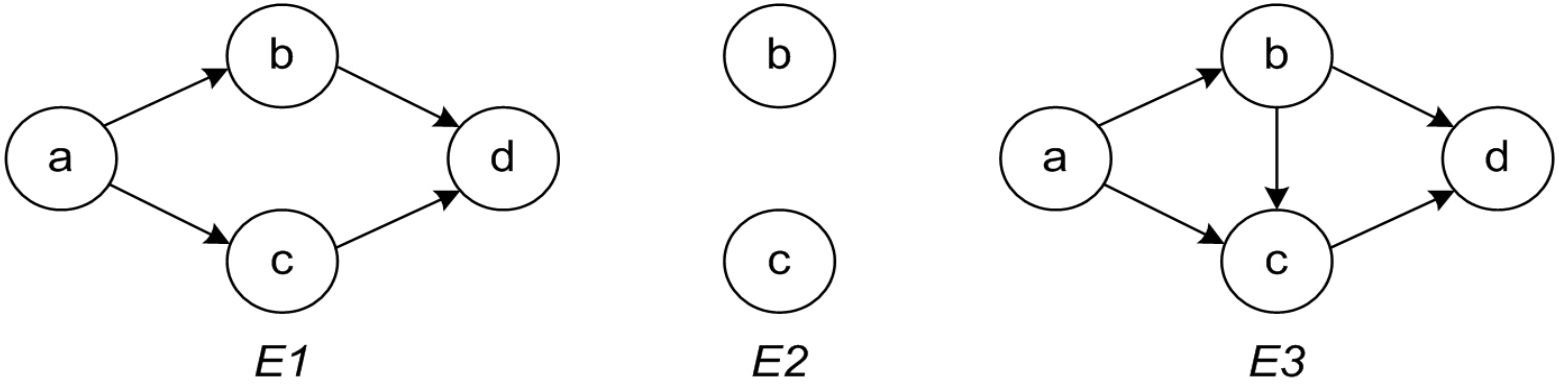
Episode-Mining

32 Zeitfenster der Länge 5:



„**Episode**“: Prozessmodell-Fragment.

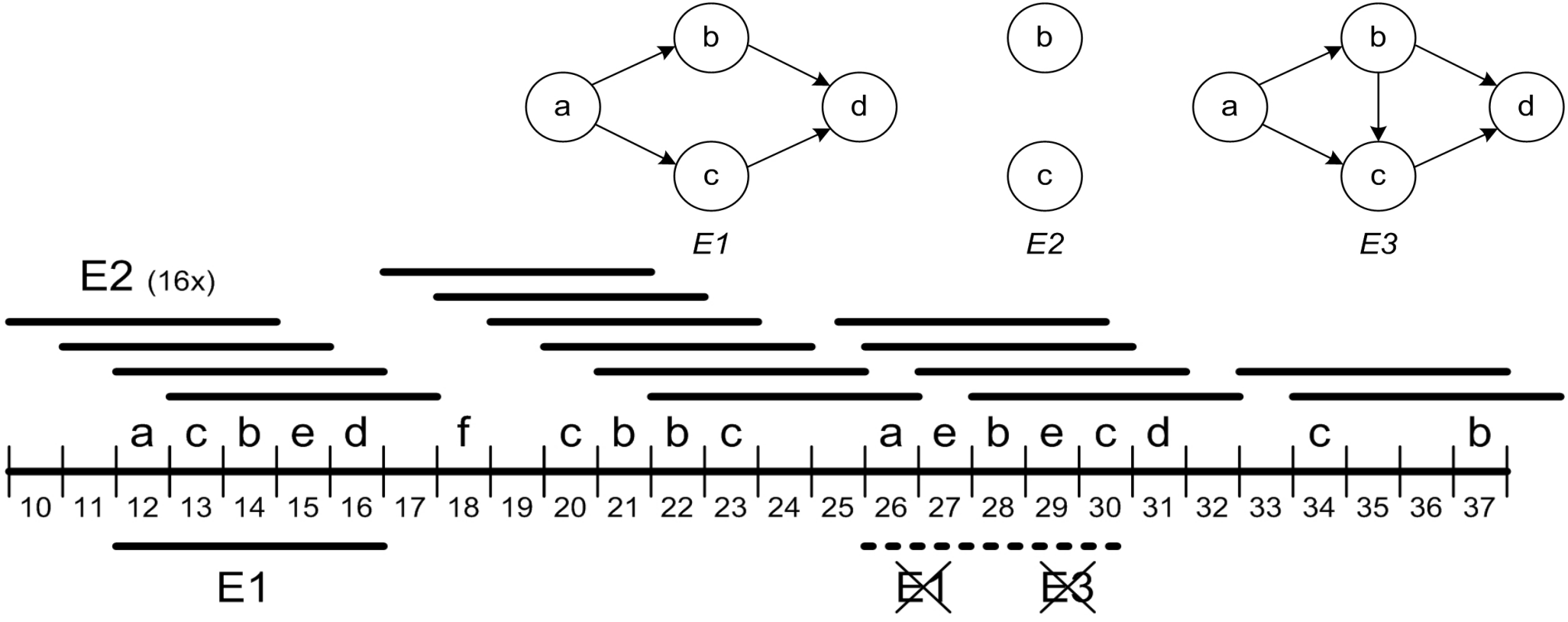
3 Beispiele:



(weitere sind möglich, z.B. unter Verwendung von e,f).

Wie oft treten diese Episoden hier auf (d.h. wie oft gibt es vollständigen Ablauf eines Modells in Zeitfenster der Länge 5) ?

Auftritte von Episoden



16 Auftritte von E2 (b, c in beliebiger Reihenfolge).

Davon 1 Auftritt von E1: *E2* => *E1* hat *confidence* 1/16.

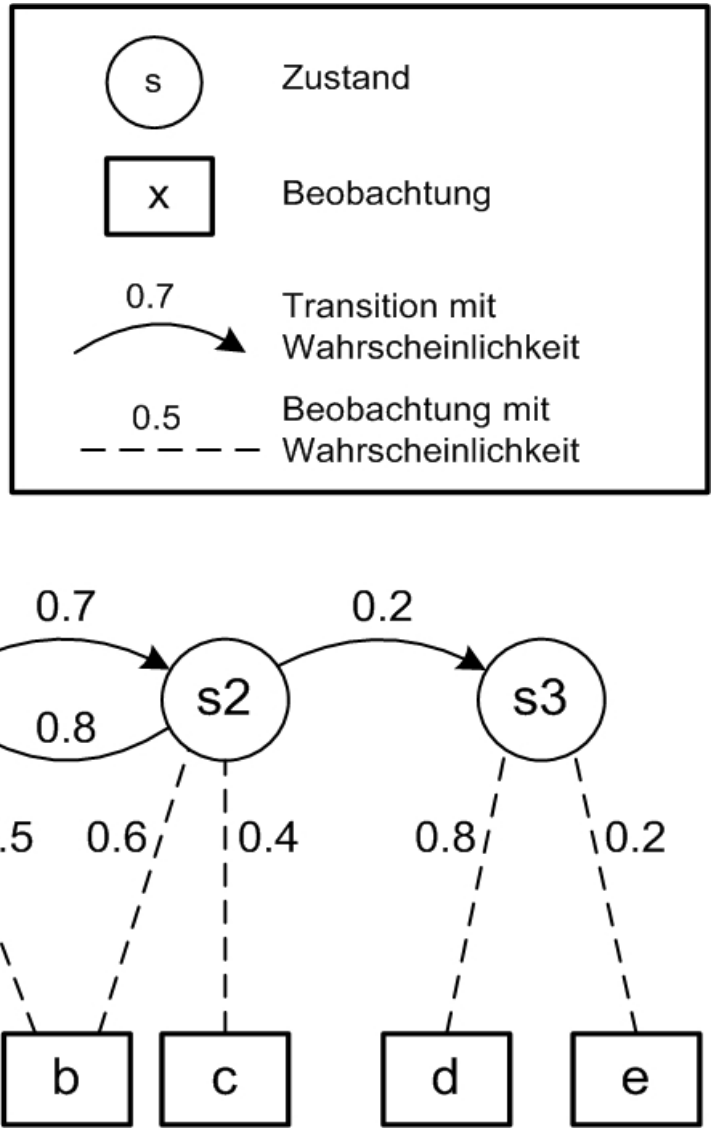
Weitere Auftritte von E1 und E3 bei Fenstergröße 6.

Überblick Data-Mining

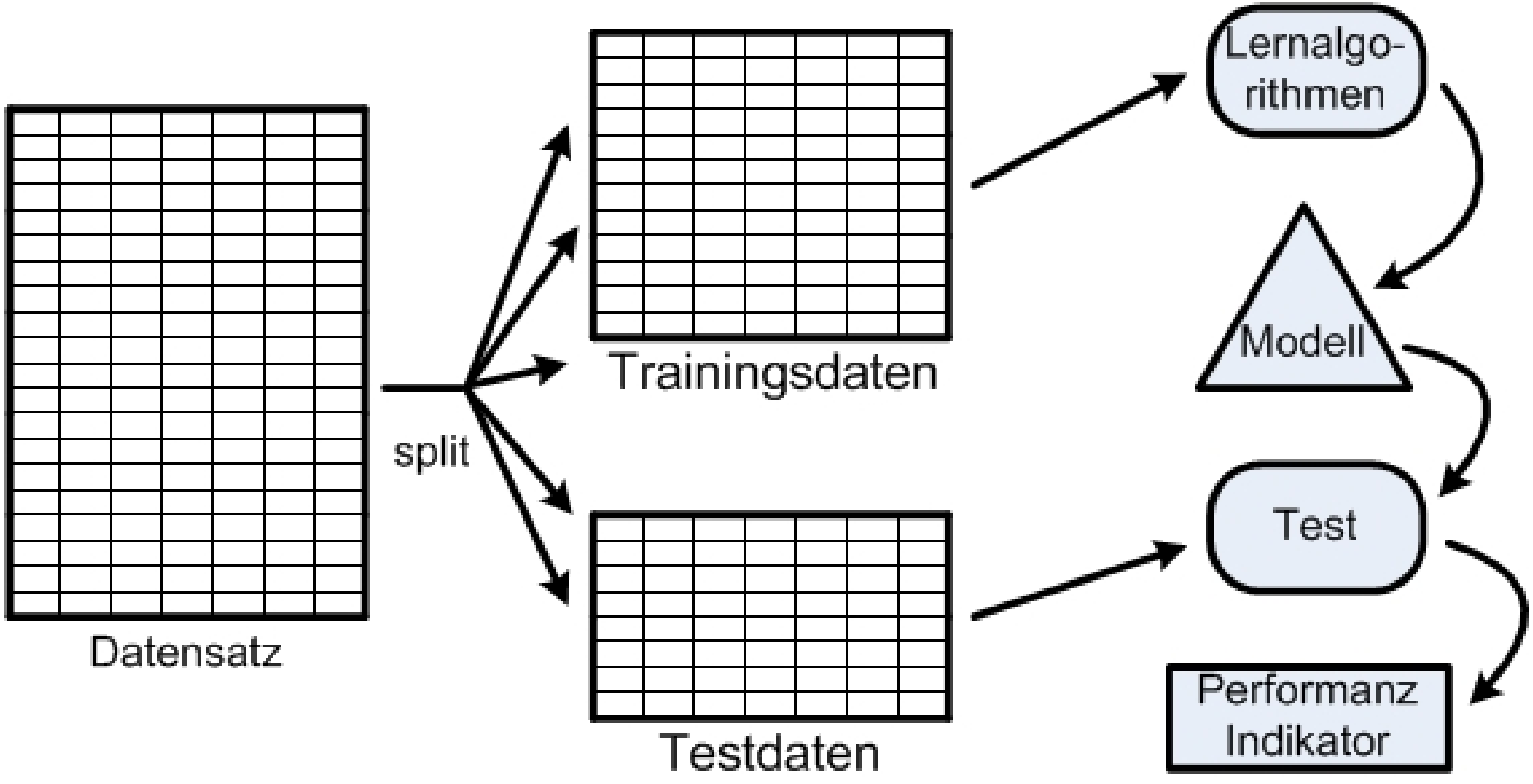
- Von Datensätzen zu Entscheidungsbäumen durch überwachtes / nicht überwachtes Lernen
- Konfusionsmatrix
- Cluster-Analyse
- Assoziationsregel-Lernen
- Sequence- und Episode-Mining
- Hidden-Markov-Modell und Validierung

Hidden-Markov-Modell

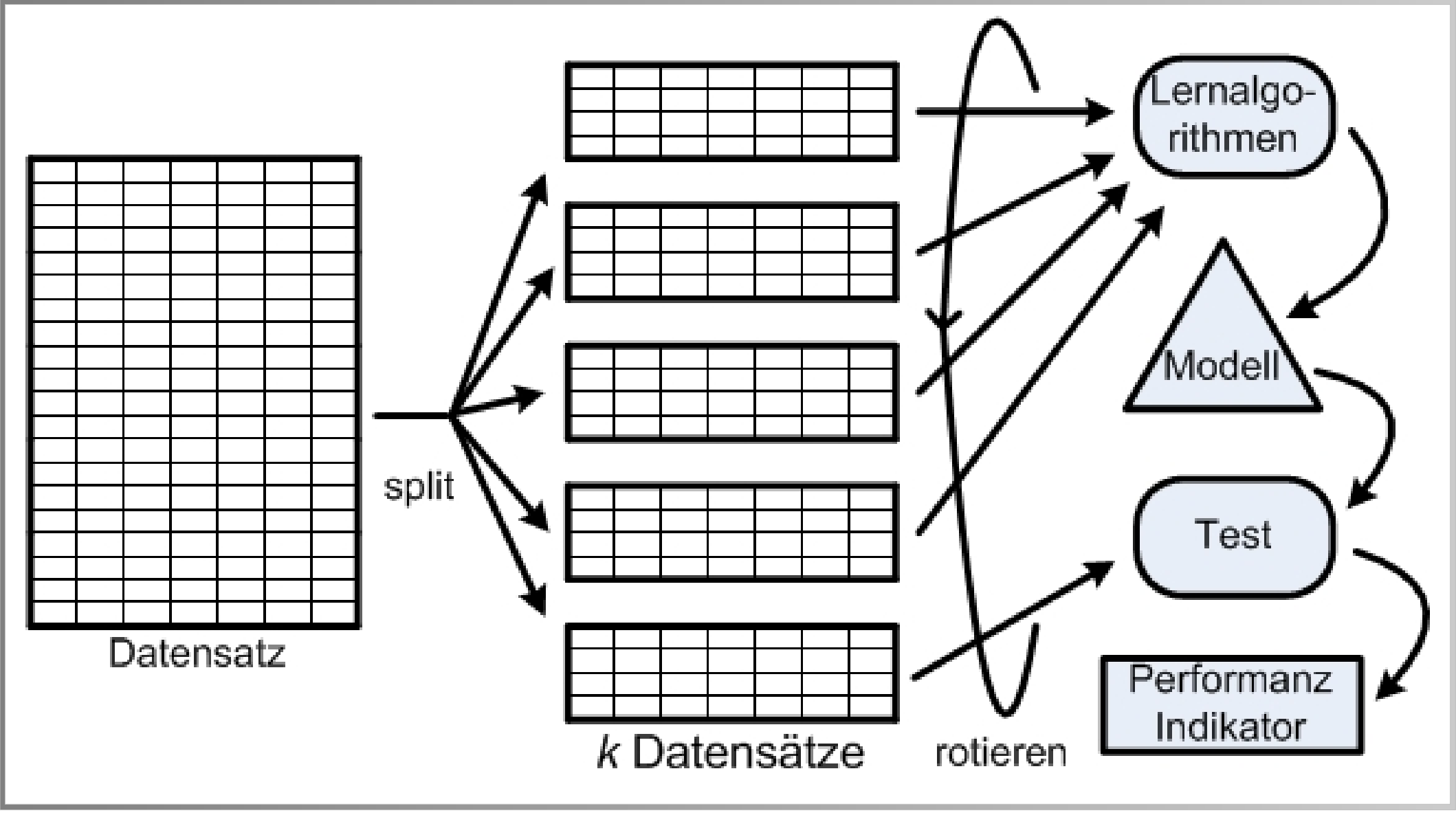
1) **Wahrscheinlichkeit** der Sequenz berechnen (gegeben Beobachtungssequenz und Hidden-Markov-Modell).
2) Gegeben Beobachtungssequenz und Hidden-Markov-Modell, wahrscheinlichsten „**hidden path**“ im Modell berechnen (= interne Zustandsfolge; kann zur z.T. beobachtet werden).
3) Bei gegebener Beobachtungs-sequenz **Hidden-Markov-Modell** ableiten, welches mit max. Wahrscheinlichkeit Sequenzen erzeugt.



Qualitätsbewertung von Lernalgorithmen: Cross-Validierung



Verlässlichere Qualitätsbewertung: k-fache Cross-Validierung



Zusammenfassung

In diesem Abschnitt:

- Von Datensätzen zu Entscheidungsbäumen durch überwachtes / nicht überwachtes Lernen.
- Konfusionsmatrix.
- Cluster-Analyse.
- Assoziationsregel-Lernen.
- Sequence- und Episode-Mining.
- Hidden-Markov-Modell und Validierung.

Im nächsten Abschnitt:

- Datenbeschaffung.