Technische Universität Kaiserslautern
Fachbereich Informatik
Lehrgebiet Datenverwaltungssysteme

Integriertes Seminar
Datenbanken und Informationssysteme
Sommersemester 2005
Thema: Data Streams

Data Mining auf Datenströmen

Andreas M. Weiner
amw@amweiner.de

22. Juli 2005

Überblick

1. Was ist Data Mining?
2. Klassische Data-Mining-Verfahren
3. Data Mining auf Datenströmen
4. Fazit und Ausblick

1. Was ist Data Mining?

1.1 Motivation

„I never waste memory on things that can easily be stored and retrieved from elsewhere.“

Albert Einstein (1879–1955)

1.2 Data Mining

Definition: Unter Data Mining versteht man Verfahren zum Aufspüren von neuen und interessanten Mustern und Regeln, sowie von Beziehungen zwischen Datensätzen in großen Datenmengen.

1.3 Der Knowledge-Discovery-Prozess

Data Mining ist Teil eines sechsstufigen Knowledge-Discovery-Prozesses:

(1) Datenauswahl

(2) Datenbereinigung

(3) Datenanreicherung

(4) Datentransformation

(5) Data Mining

(6) Berichterstattung und Reportgenerierung

2. Klassische Data-Mining-Verfahren

2.1 Assoziationsregeln

Zielsetzung beim Association Rule Mining:

- Finden von interessanten Mustern
- Aufspüren von Regelmäßigkeiten
- Aufdeckung von Abhängigkeiten

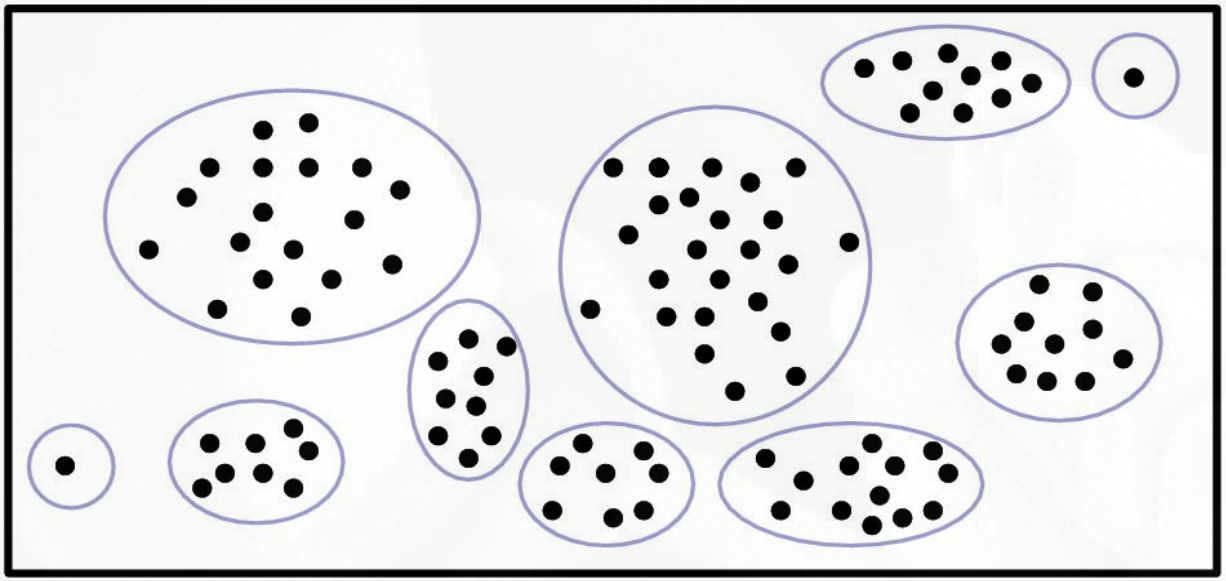
Beispiel für eine Assoziationsregel:

If kauft(X, „Buddenbrooks“) then kauft(X, „Der Zauberberg“)

[support=5%, confidence=65%]

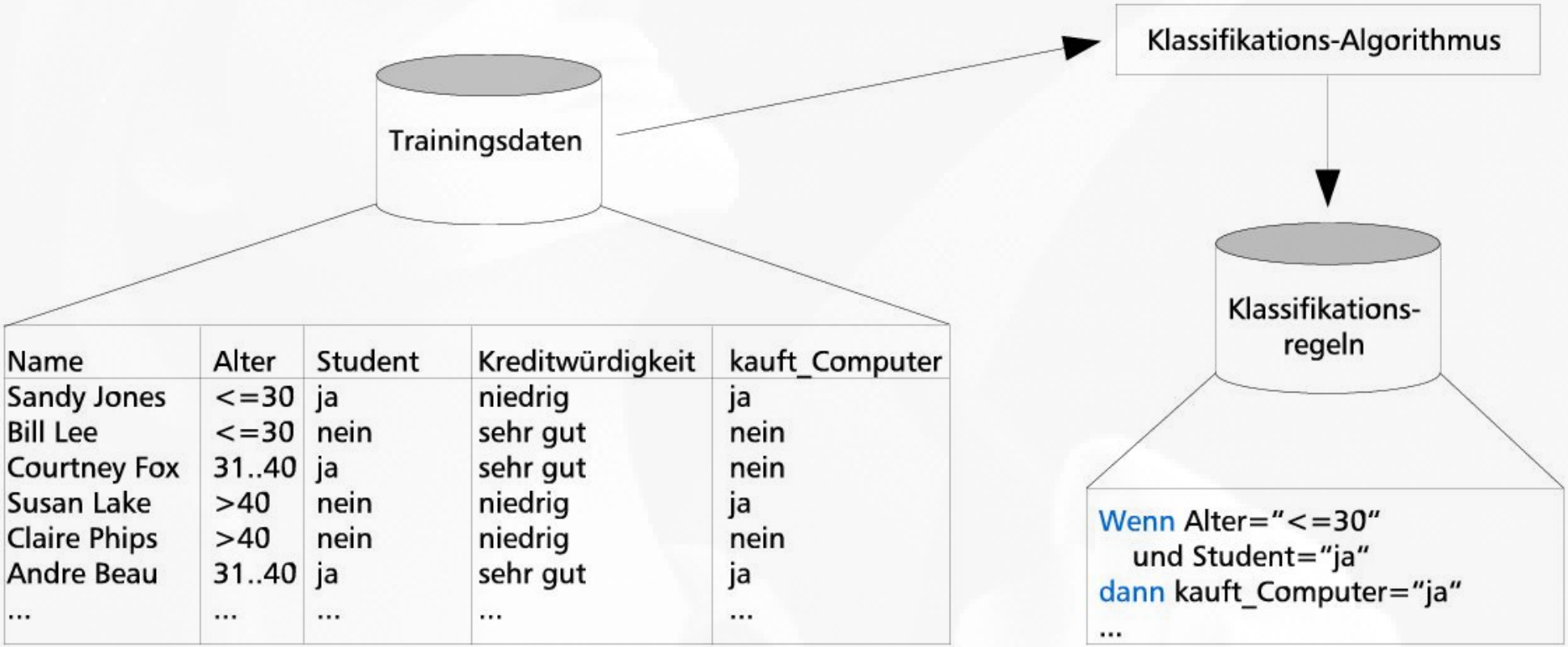
2.2 Cluster-Analyse

Definition: Unter einem Cluster versteht man eine Menge von Objekten, die eine *gewisse Ähnlichkeit* zu Elementen des gleichen Clusters aufweisen und verschieden von Elementen anderer Cluster sind. Ein Cluster wird durch ein kanonisches Element – dem *Cluster-Zentrum* – bestimmt.



2.3 Klassifikation (1)

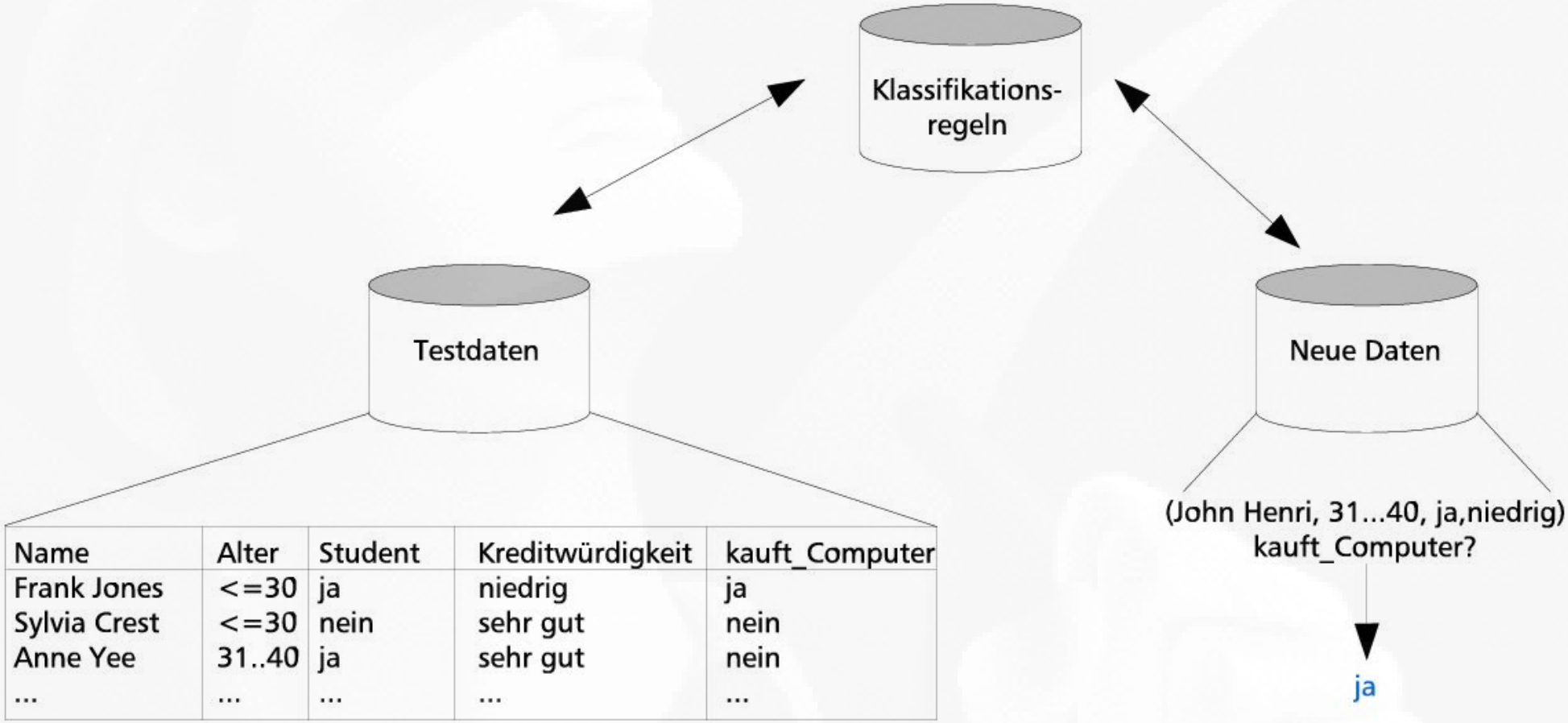
1. Lernen eines Klassifikators für das Konzept *„X kauf Computer“*



Name	Alter	Student	Kreditwürdigkeit	kauft_Computer
Sandy Jones	<=30	ja	niedrig	ja
Bill Lee	<=30	nein	sehr gut	nein
Courtney Fox	31..40	ja	sehr gut	nein
Susan Lake	>40	nein	niedrig	ja
Claire Phips	>40	nein	niedrig	nein
Andre Beau	31..40	ja	sehr gut	ja
...	...	...	...	...

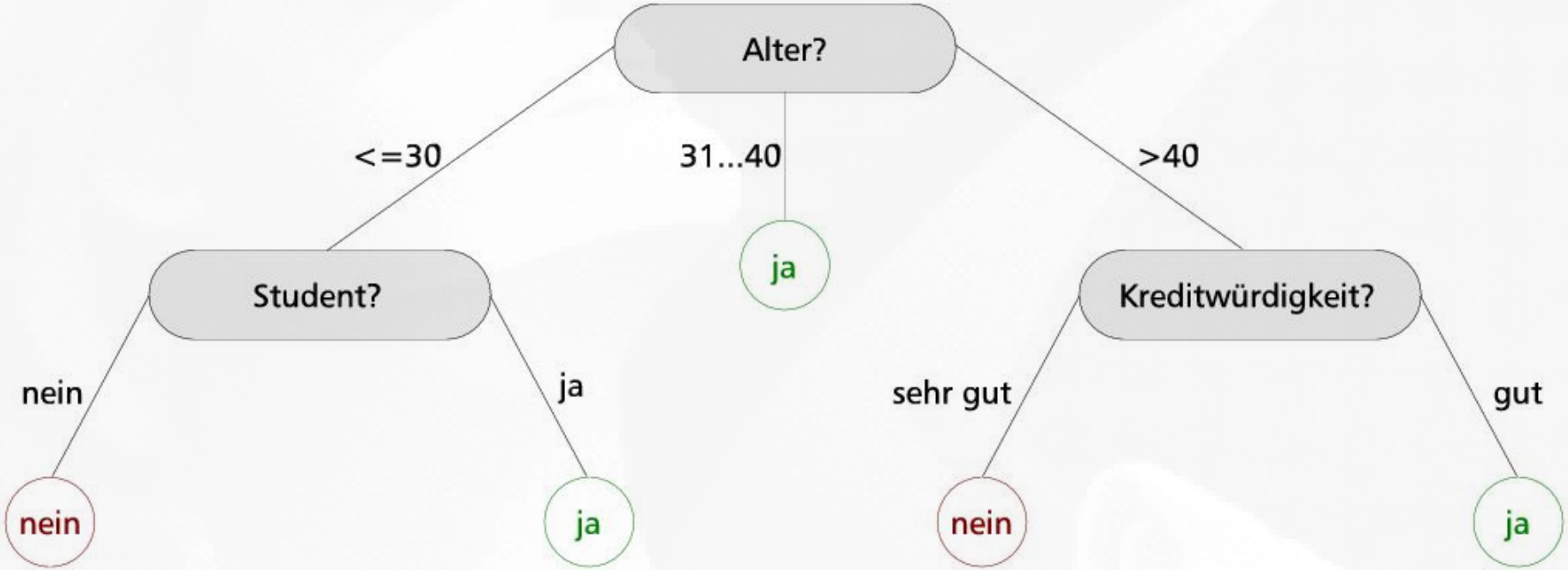
2.3 Klassifikation (2)

2. Klassifikation neuer Datensätze



Name	Alter	Student	Kreditwürdigkeit	kauft_Computer
Frank Jones	<=30	ja	niedrig	ja
Sylvia Crest	<=30	nein	sehr gut	nein
Anne Yee	31..40	ja	sehr gut	nein
...	...	...	...	...

2.3 Klassifikation (3)



Entscheidungsbaum für das Konzept *„X kauft Computer“*

3. Data Mining auf Datenströmen

3.1 Data Mining auf Datenströmen

Designkriterien für Data-Mining-Algorithmen

- Nur einmaliger Scan der Daten
- Verarbeitung in konstanter Zeit
- Geringer Arbeitsspeicherbedarf
- Zu jedem Zeitpunkt muss ein vernünftiges Modell vorliegen
- Erzeugung eines Modells, welches äquivalent zu einem Modell ist, das durch einen klassischen Data-Mining-Algorithmus erzeugt wurde
- Aktuelles Modell zu jeder Zeit

3.2 Der STREAM-Algorithmus (1)

Der STREAM-Algorithmus...

- ist ein divide-and-conquer-Algorithmus im klassischen Sinne
- verwendet ein randomisiertes Approximationsverfahren
- gibt die k gefundenen Cluster-Zentren aus

Algorithmus 6 : STREAM

```
1 foreach chunk X_i in the stream do
2     if a sample of size ≥ (1/ε) log(k/δ) contains fewer than k distinct points then
3         X_i ← weighted representation;
4     end
5     Cluster X_i using LSEARCH;
6     X' ← ik centers obtained from chunks 1 through i iterations of the stream, where each
      center c obtained by clustering X_i is weighted by the number of points in X_i assigned to
      c;
7     Output the k centers obtained by clustering X' using LSEARCH;
8 end
```

3.2 Der STREAM-Algorithmus (2)

Algorithmus 9 : LSEARCH(N, $d(\cdot,\cdot)$, k, ϵ, ϵ', ϵ'')

Input : N is a data set of size n; the metric $d(\cdot,\cdot)$; k the number of medians to create; $\epsilon, \epsilon', \epsilon'' \in \mathbb{R}$

Output : a solution (F', g')

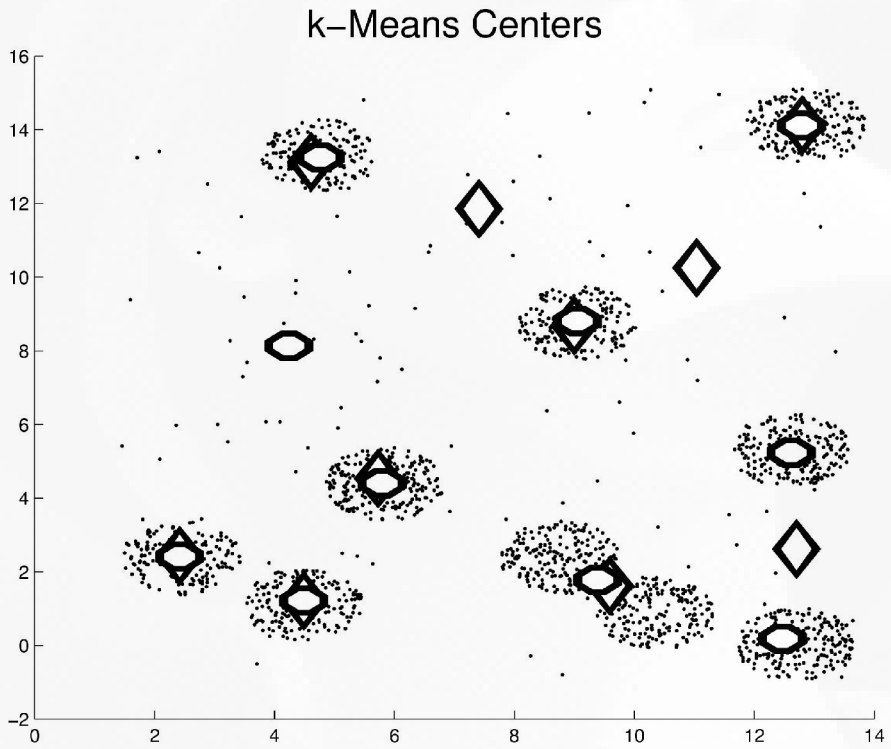
1 $z_{min} = 0$;
`/* for` x_0 `an arbitrary point in` N `*/`
2 $z_{max} = \sum_{x \in N} d(x, x_0)$;
3 $z = \frac{z_{max}+z_{min}}{2}$;
4 (I, a) =`InitialSolution(`N`, `z`)`;
5 Randomly pick $\Theta(\frac{1}{p} \log k)$ points as feasible medians;
6 **while** $\#$ *medians* $\neq k \wedge z_{min} < (1-\epsilon'') \cdot z_{max}$ **do**
7 Let (F, g) be the current solution;
8 Run `FL(`N`, `d`,`z`, `ϵ`, `(F, g)`)` to obtain a new solution (F', g');
9 **if** $k \leq |F'| \leq 2k$ **then** then exit loop;
10 **if** $|F'| > 2k$ **then**
11 $z_{min} = z$;
12 $z = \frac{z_{max}+z_{min}}{2}$;
13 **end**
14 **if** $|F'| < k$ **then**
15 $z_{max} = z$;
16 $z = \frac{z_{max}+z_{min}}{2}$;
17 **end**
18 **end**
19 **return** *our solution* (F', g');

3.2 Der STREAM-Algorithmus (3)

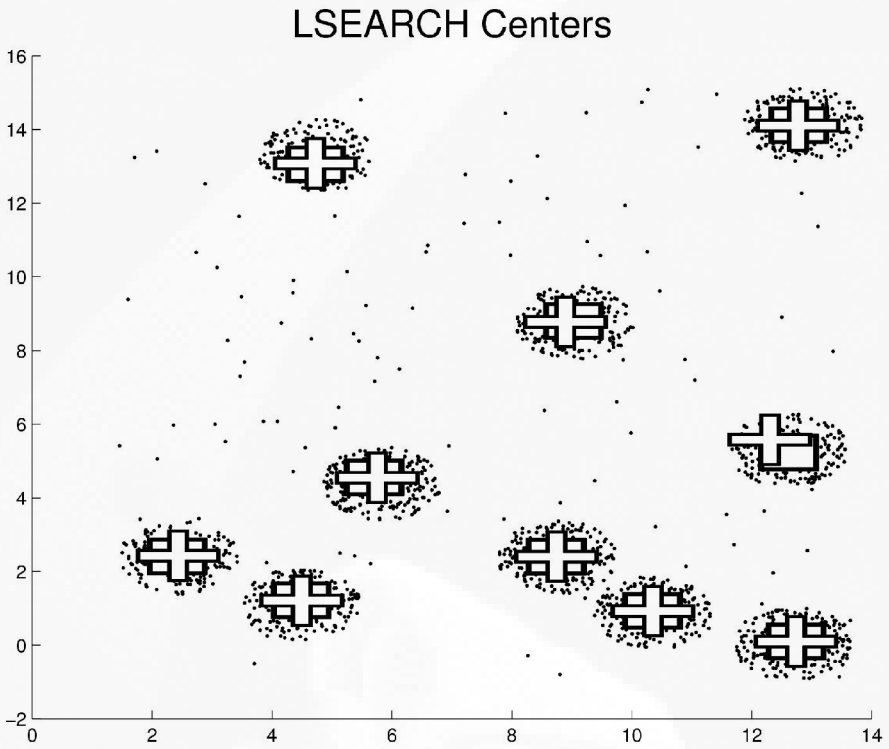
Gütekriterien

- STREAM startet zunächst mit mehr als k Cluster-Zentren und verbessert diese nach und nach auf genau k.
- beschränkter Suchraum, der potentiell „gute“ Cluster-Zentren enthält
- $\Theta(1/p \log k)$ beliebige Elemente als Cluster-Zentren sind ausreichend
- Laufzeit $O(nm + nk \log k)$

3.2 Der STREAM-Algorithmus (4)



(a)



(b)

3.3 Der VFDT-Algorithmus (1)

Idee des VFDT (Very Fast Decision Tree learner)-Algorithmus:

- Baue einen Entscheidungsbaum auf
- Verwende nur einen Teil der Trainingsdaten für die Test-Auswahl
- Schätze die Anzahl der benötigten Trainingsdaten ab
- Garantiert „gute“ Auswahl des Testattributs
- Wähle die ersten Datensätze des Datenstroms als Test der Wurzel aus und setze dieses Verfahren rekursiv auf den Kindern fort

3.3 Der VFDT-Algorithmus (2)

Wieviele Trainingsdaten werden konkret benötigt?

Man betrachte eine reele Zufallsvariable r mit einer oberen Schranke $R=1$.

Führt man nun n unabhängige Zufallsexperimente durch und berechnet man deren Mittelwert $\bar{r}$, dann folgt mit der *Hoeffding-Schranke*, dass mit einer Wahrscheinlichkeit von $1-\delta$ der echte Mittelwert der Zufallsvariablen mindestens $\bar{r}-\epsilon$ ist, mit

$$\epsilon=\sqrt{\frac{R^2 \ln(1/\delta)}{2n}}\,.$$

3.3 Der VFDT-Algorithmus (3)

Beispiel

$\delta = 0{,}05$

$\epsilon = 0{,}01$

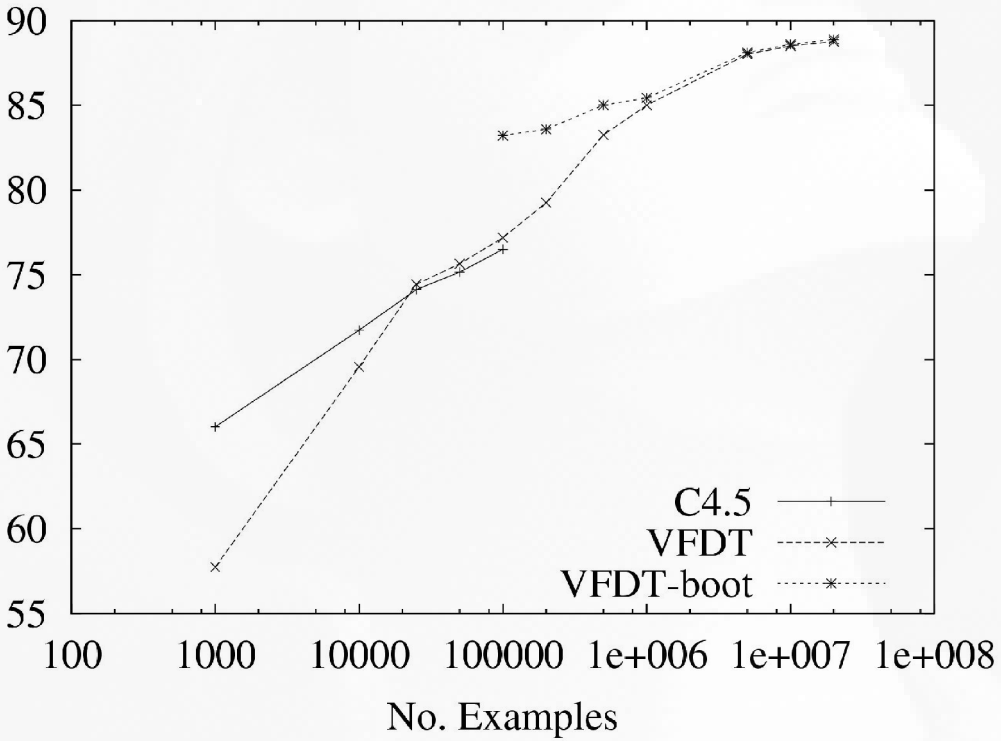
$\Rightarrow n \approx 14.979$ *Trainingsdatensätze*

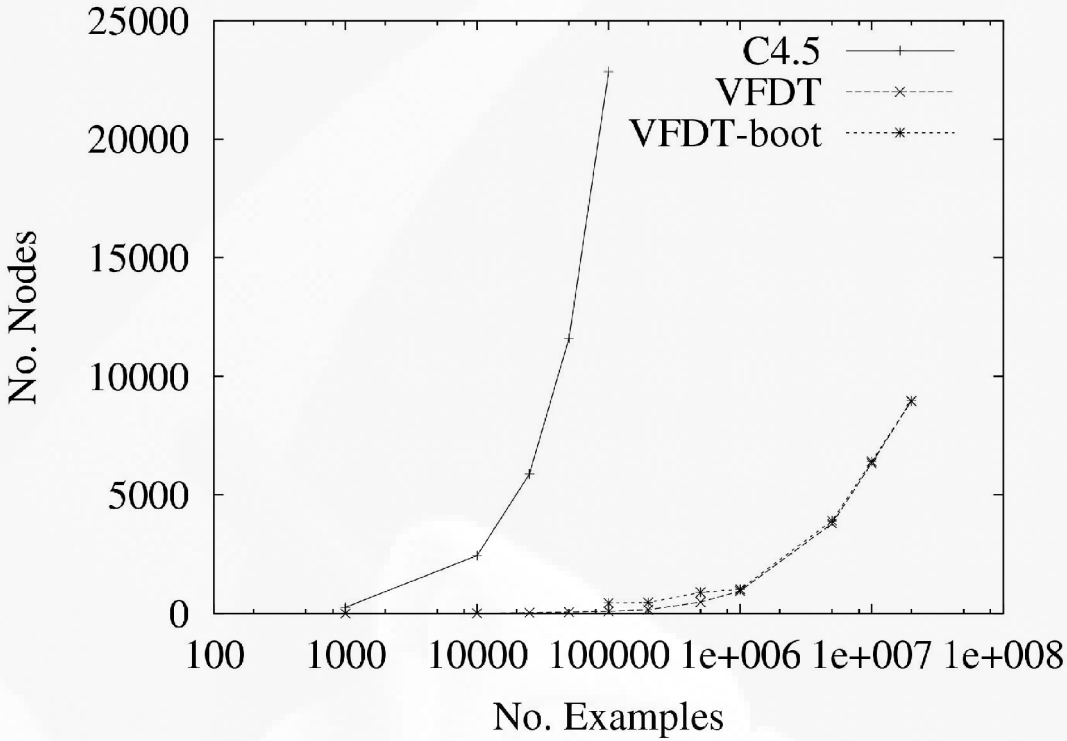
$\delta = 0{,}01$

$\epsilon = 0{,}005$

$\Rightarrow n \approx 92.104$ *Trainingsdatensätze*

3.3 Der VFDT-Algorithmus (4)





3.4 Der FP-Stream-Algorithmus (1)

Ziele des FP-Stream-Algorithmus

- Aufspüren von häufig vorkommenden Mustern (engl. *frequent patterns)*
- Ableitung von Assoziationsregeln

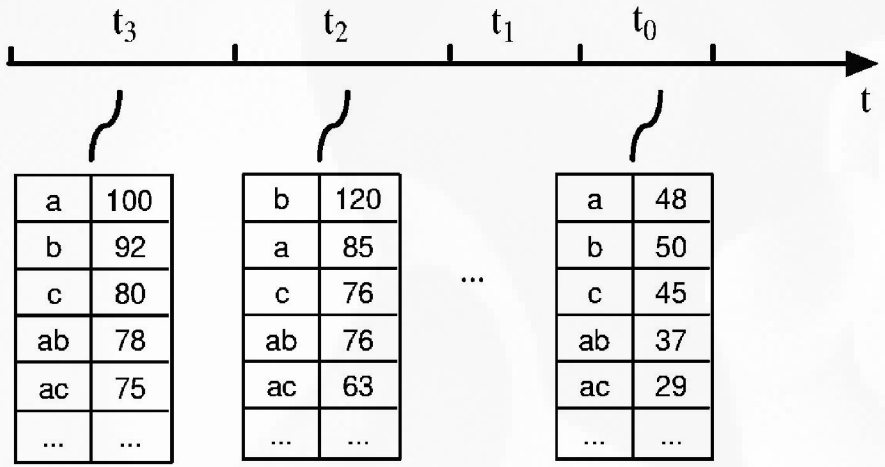
Ein potentiell unendlicher Datenstrom verlangt die...

- Aufbewahrung aller vorkommenden Muster
- Berücksichtigung unterschiedlicher Zeitgranularitäten

3.4 Der FP-Stream-Algorithmus (2)

- Gekippte Zeitfenster (engl. *tilted-time-windows*) dienen der Verwaltung unterschiedlicher Zeitgranularitäten

- Zuordnung von Mustern zu den gekippten Zeitfenstern



3.4 Der FP-Stream-Algorithmus (3)

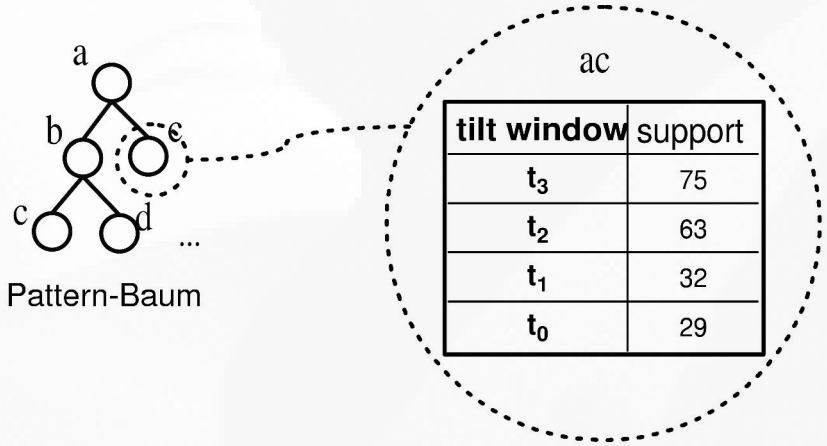
- Verwaltung von gekippten Zeitfenstern trivial
- Reduzierung der benötigten Zeitfenster durch logarithmische Partitionierung

Beispiel (*Speicherung der Daten eines Jahres in Zeitfenstern*)

- herkömmliche Partitionierung: $365*24*4=35.040$ Zeitfenster
- logarithmische Partitionierung: $\log_2(365*24*4)+1\approx17$ Zeitfenster

3.4 Der FP-Stream-Algorithmus (4)

- Der Pattern-Baum dient der effizienten Verwaltung der Muster



Der FP-Stream-Algorithmus…

- unterteilt den Datenstrom in Stapel (engl. *batches*)
- bewahrt nur einen Teil der Stapel auf
- bestimmt eine approximative Auftrittshäufigkeit

4. Fazit und Ausblick

4. Fazit und Ausblick

Weiterentwicklung des VFDT-Algorithmus

- Verarbeitung von nummerischen Attributen (Jin und Agrawal)
- Verbesserung der Vorhersage (Jin und Agrawal)
- Reduzierung der Sample-Größe (Jin und Agrawal)
- On-Demand-Algorithmus (Aggarwal et al.)

MAIDS (**M**ining **A**larming **I**ncidents from **D**ata **S**treams) unterstützt

- Cluster-Analyse
- Klassifikation
- Aufspüren von Assoziationsregeln
- Visualisierung

Vielen Dank für Ihre Aufmerksamkeit!