Ludwig Maximilians Universität München
Institut für Informatik
Lehr- und Forschungseinheit für Datenbanksysteme

Skript zur Vorlesung

Knowledge Discovery in Databases II

im Sommersemester 2007

Kapitel 3: Projected, Corelation und Subspace Clustering

Skript übernommen aus KDD © 2003 Johannes Aßfalg, Christian Böhm, Karsten Borgwardt, Martin Ester, Eshref Januzaj, Karin Kailing, Peer Kröger, Jörg Sander und Matthias Schubert

http://www.dbs.ifi.lmu.de/Lehre/KDD

Inhalt dieses Kapitels

3.1 Einführung

Lösung: Dimensionsreduktion

- Daten-Vorverarbeitung: Reduktion der Dimensionalität durch Selektion der (für das Clustering) relevanten Features
- Verwende Datensatz mit reduzierter Dimensionalität zum Clustern

Allgemein:

- Gegeben: n Datenpunkte (Featurevektoren) $DB = \{x_1,...,x_n\}$ mit Dimensionalität d
- Gesucht: Transformation der Datenpunkte in (d-k)-dimensionale Featurevektoren, so dass der dabei gemachte Fehler möglichst klein ist

Einfacher Ansatz:

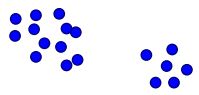
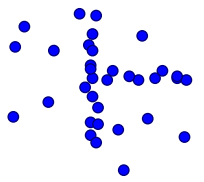
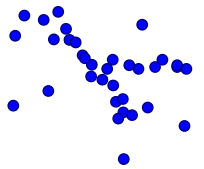
- Abgeleitete Attribute (Summe, Durchschnitt, etc.) statt urspr. Attribute
  - erfordert Expertenwissen
  - kaum automatisierbar
  (+) teilweise gute Ergebnisse (hängt vom Experten ab)

3.1 Einführung

2. Grundproblem für das Clustering:

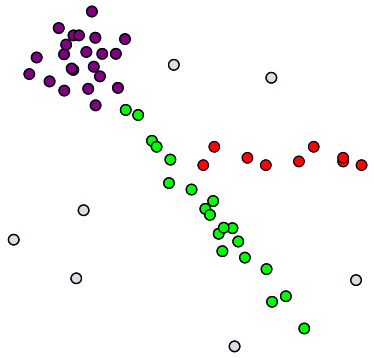
Cluster in verschiedenen Unterräumen

- *Für zwei Klassen können zwei verschiedene Mengen an Attributen relevant/irrelevant sein*

Clustering: Gruppierung der DB-Objekte in Teilmengen, so dass Intra-Cluster-Ähnlichkeit maximiert und Inter-Cluster-Ähnlichkeit minimiert wird	
Merkmals-Relevanz: Einzelne Merkmale unterliegen starkem Rauschen und verschlechtern das Clustering-Ergebnis	
Merkmals-Korrelation: Redundanz durch Korrelationen zwischen den einzelnen Merkmalen	

3.1 Einführung

- Charakteristik von Merkmals-Relevanz und Merkmals-Korrelation kann individuell für jeden Cluster sein
- Hinweis auf unterschiedliche stochastische Prozesse
- Ermittlung von Clustern mit einheitlicher Charakteristik

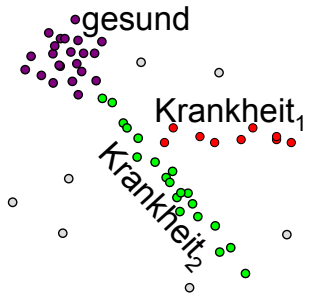


- Cluster mit Subspace-Präferenz
- Korrelations-Cluster
- Konventioneller Cluster
- Rausch-Objekte

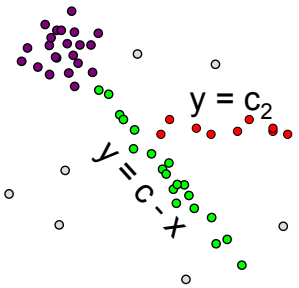
3.1 Einführung

Anwendungen:

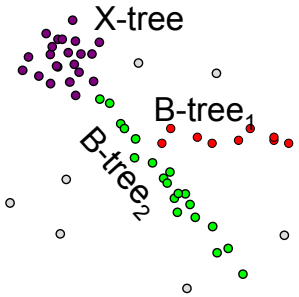
Modellierung



Kompression

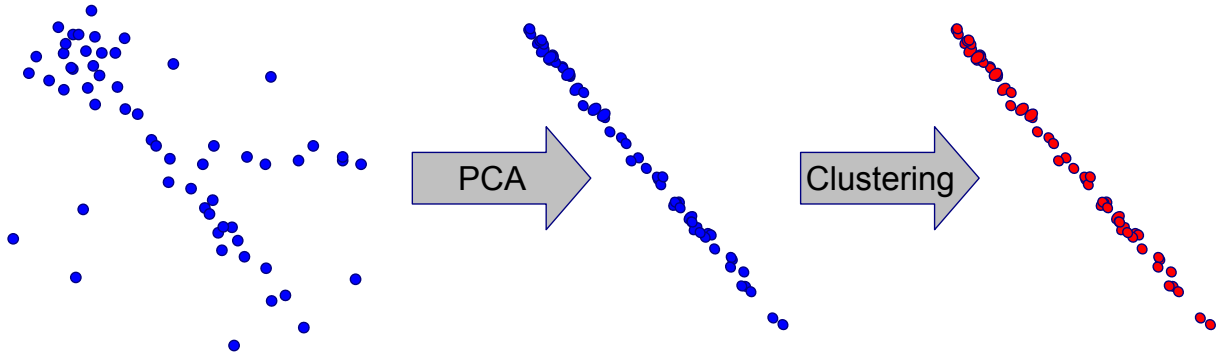


Indexierung



3.1 Einführung

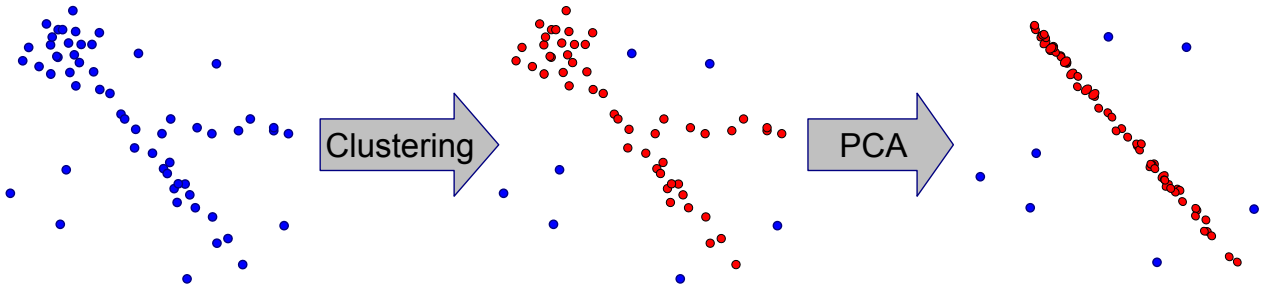
Erst Dimensionsreduktion, dann Cluster-Analyse



3.1 Einführung

Erst Dimensionsreduktion, dann Cluster-Analyse

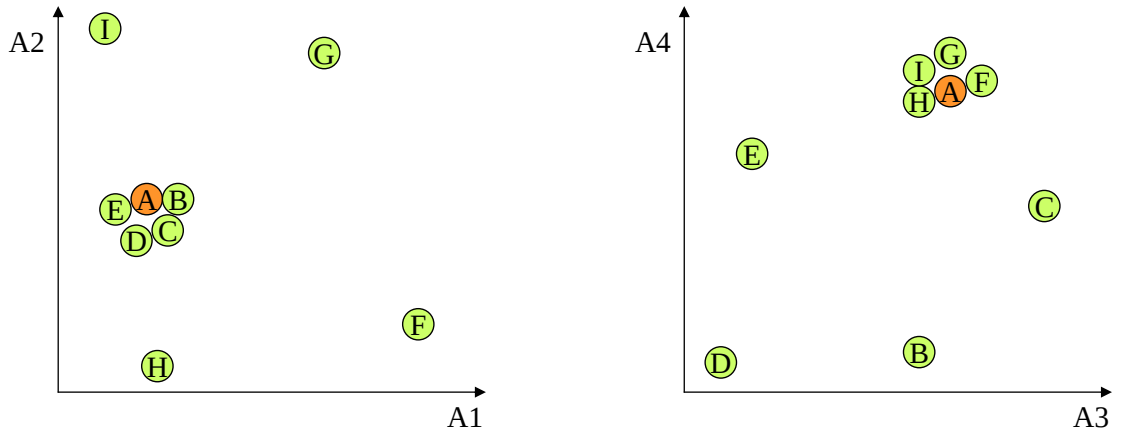
Erst Cluster-Analyse, dann Dimensionsreduktion



Integriere Merkmalsanalyse in den Clustering-Prozess

3. Grundproblem für das Clustering:
Überlappende Cluster

- *Bei manchen Anwendungen:*
  Objekte können in unterschiedlichen Teilräumen unterschiedlich clustern



Übersicht: Clustering hochdimensionaler Daten

Dimensionsreduktion → Meist unbrauchbar bei lokaler Merkmalsrelevanz

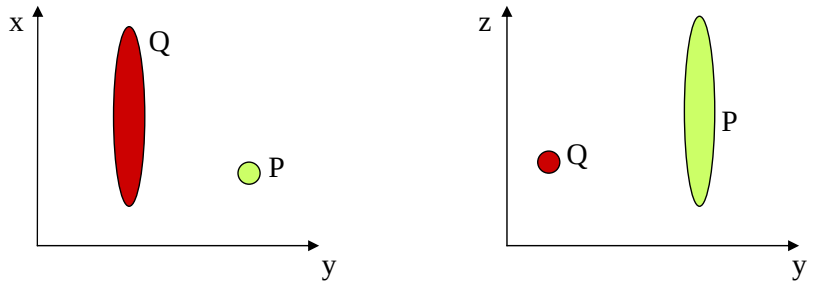
3.1 Projected Clustering
3.2 Correlation-Clustering
3.3 Subspace Clustering

→ Integration von Varianz- und Covarianz-Analyse in Clustering-Verfahren

3.2 Projected Clustering

Projected Clustering:

- Integration von Varianzanalyse in den Clustering-Prozess
- Jeder Cluster enthält Punkte mit geringer Varianz in einer beliebigen Projektion des Datenraums
- Verfahren:
  - PROCLUS (basiert auf k-medoid Verfahren)
  - PreDeCon (basiert auf DBSCAN)



3.2.1 PROCLUS

PROCLUS [Aggarwal & Procopiuc 1999]

- Erweitert *k*-Medoid Verfahren CLARANS um Varianzanalyse für die gefundenen Cluster
- Verwendet Manhattan Distanz (L_1) geteilt durch die Unterraumdimension, um Objekt-Distanzen aus verschiedenen Unterräumen „objektiv“ zu vergleichen
- Benötigt Anzahl der Cluster *k* und durchschnittliche Dimension der Cluster *l* als Eingabe

Phase 1 Initialisierung: Bestimmen der initialen Medoide durch „Greedy“-Methode auf DB-Sample

- Beginne mit einem zufälligen Medoid
- Wähle den weitest entferntesten Punkt zu allen bisherigen Medoiden als neuen Medoid bis *k* Medoide ausgewählt sind.

Phase 2 Iteration: Optimierung der Medoide und der Cluster-Unterräume

- Medoid-Optimierung wie bei k-Medoid (PAM, CLARANS)

- Unterraumoptimierung (Varianzanalyse):
  d_i = minimale Distanz des Medoiden m_i zu allen anderen Medoiden
  L_i = {Punkte, die zu m_i einen kleineren Abstand als d_i haben}
  $X_{i,j}$ = durchschnittliche Distanz aller Punkte aus L_i zu m_i in Dimension j
  Je kleiner $X_{i,j}$ desto näher liegen die Punkte in $L_{i,j}$ entlang Dimension j bei m_i

  Wähle Unterräume S_i, so dass insgesamt $k \cdot l$ Dimensionen zu den Medoiden zugeordnet wurden (Greedy-Verfahren)

  Zuordnung der Punkte zu den m_i unter Berücksichtigung der ausgewählten S_i

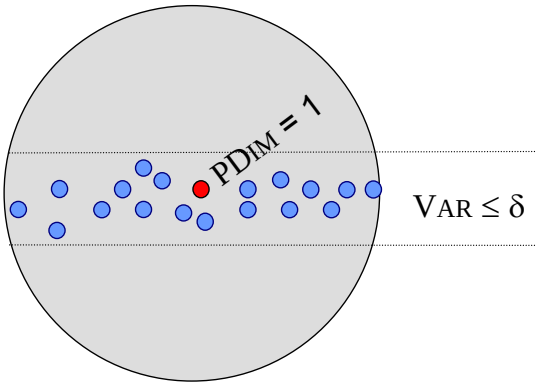
Diskussion

- Nachteile lokal optimierender Verfahren wie *k*-Medoids
  - Konvergiert evtl. nur gegen ein lokales Minimum
  - Eingabeparameter *k* schwer zu bestimmen
  - Anfällig gegen Rauschen
  - Berechnet nur konvexe Cluster
  - Inputparameter *l* schwer zu bestimmen
- Idee: Integration von Varianz- und Covarianz-Analyse in dichtebasierte Clustering-Verfahren
- Wie?
  - Ermittlung der Charakteristik von Merkmals-Relevanz und Merkmals-Korrelation aus der Nachbarschaft eines jeden Punktes
  - Vereinigung von Punkten mit ähnlicher Charakteristik zu einem Cluster

PreDeCon [Böhm, Kailing, Kriegel, Kröger 2004]

Intuition: Wenn ein Punkt zu einem Subspace-Cluster gehört, ist das in der lokalen Nachbarschaft in der Varianz sichtbar.

- Euklidische Bereichsanfrage mit Radius ε
- Ermittlung der Varianz in allen Dimensionen separat
- Ermittlung der Anzahl der Dimensionen mit Varianz ≤ δ



Subspace preference dimensionality (PDIM):
Dimensionen mit VAR > δ

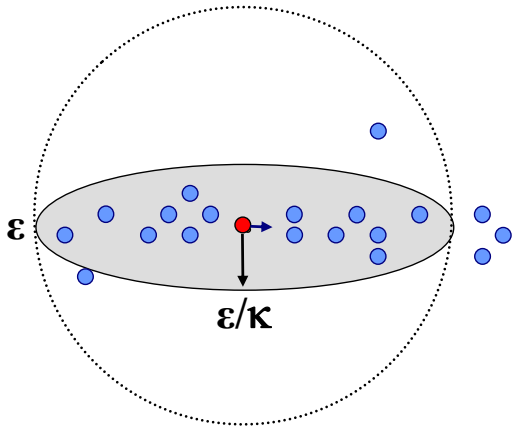
Parameter λ: Finde Cluster mit PDIM ≤ λ

Verwende gewichtete Euklidische Distanz fürs Clustering:

$$dist_P(P,Q) = \sqrt{\sum_i w_i \cdot (p_i - q_i)}$$

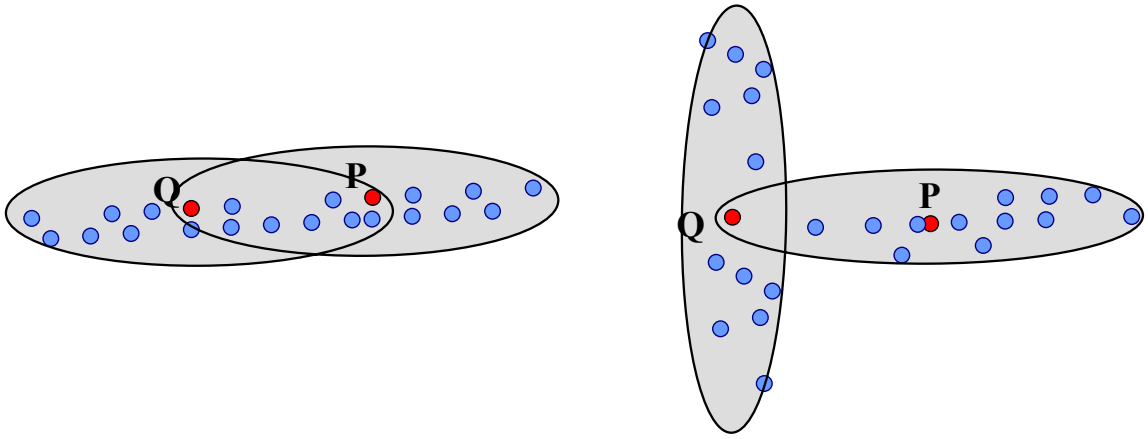
Bestimmung des *subspace preference* Gewichts *w*:

$$w_i = \begin{cases} 1 & if \quad \text{VAR}_i > \delta \\ \kappa & if \quad \text{VAR}_i \leq \delta \end{cases}$$



κ >> 1 kontrolliert die tolerierte Abweichung („thickness“) der λ-dimensional *subspace preference* Linie oder Ebene

3.2.2 PreDeCon



Subspace Preference ε-Nachbarschaft:

$$N_\varepsilon^{\overline{w}_P}(P) = \left\{ X \in DB \,\middle|\, \max\left\{ dist_P(P,X), dist_X(X,P) \right\} \le \varepsilon \right\}$$

→ Garantiert Reihenfolgeunabhängigkeit

3.2.2 PreDeCon

Preference weighted Kernpunkt:

$$\text{CORE}_{\varepsilon,\mu}^{\lambda,\delta}(O) \Leftrightarrow \text{PDIM}(N_\varepsilon(O)) \le \lambda \wedge \left| N_\varepsilon^{\overline{w}_O}(O) \right| \ge \mu$$

Direkte Erreichbarkeit bzgl. subspace preference:

$$\text{DIRREACH}_{\varepsilon,\mu}^{\lambda,\delta}(Q,P) \Leftrightarrow$$

(1) $\text{CORE}_{\varepsilon,\mu}^{\lambda,\delta}(Q)$

(2) $\text{PDIM}(N_\varepsilon(P)) \le \lambda$

(3) $P \in N_\varepsilon^{\overline{w}_Q}(Q)$

Subspace Preference Cluster:

C ist ein *Subspace Preference Cluster* wenn

(1) Alle Punkte in C verbunden bzgl. subspace preferences sind (symmetrisch-transitive Hülle der direkten Erreichbarkeit)

(2) C ist maximal bzgl. der Erreichbarkeit

3.2.2 PreDeCon

Algorithmus PreDeCon (ε, μ , λ, δ)

seedlist = $\varnothing$

while unprocessed object exists **do**

  if seedlist is empty **then**

    insert some unproc. object *O* with $\text{PDIM}(O) \leq \lambda \wedge \left|N_\varepsilon^{\overline{w}_o}(O)\right| \geq \mu$

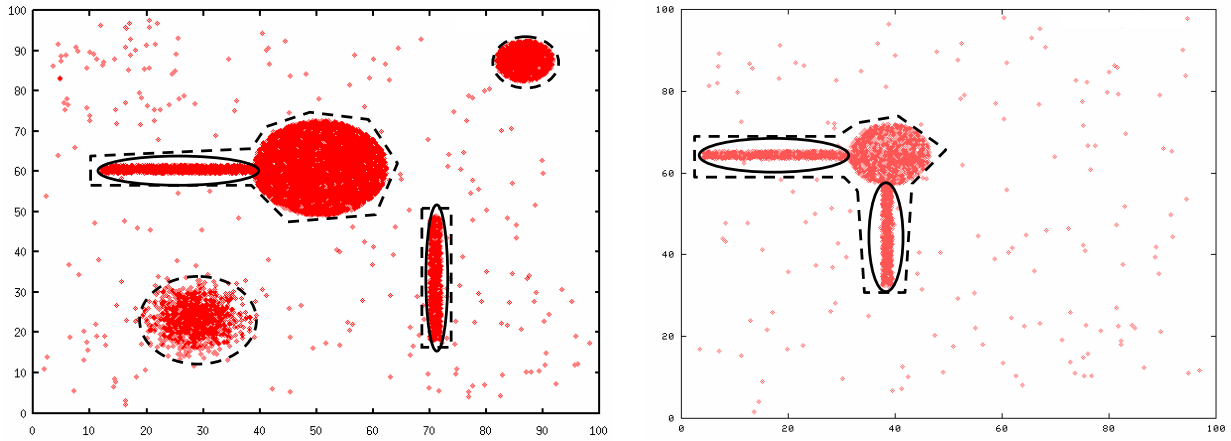
  take any element *O* out of seed list

  if $\left|N_\varepsilon^{\overline{w}_o}(O)\right| \geq \mu$ **then**

    insert all elements of $N_\varepsilon^{\overline{w}_o}(O)$ into seed list

Laufzeit: linear in *d* konstant in λ quadratisch in *n*

3.2.2 PreDeCon



Gene Expressions Daten: [Tavazoie et al., Nature Genetics, 99]

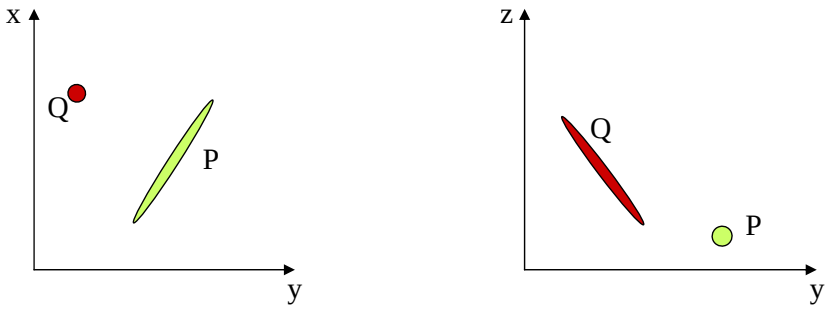
- 2800 genes, 17 time slots, Aufgabe: Finde Cluster mit funktionell verwandten Genen
- PreDeCon findet 12 Cluster mit ca. 10-14 Genen, z.B.:
  - Mehrere Gene die in die Chromatin Modellierung und Instandsetzung involviert sind (z.B. NHP10, DPB4, IES3, TAF9); IES3 und NHP10 sind sogar direkte Interaktionspartner
  - >30 Gene die strukturelle Komponenten der Ribosomen codieren (z.B. CDC33, TEF4, EFB1, und NHP2)
  - Mehrere Gene der Glykolyse (z.B. CDC19, TPI1, TDH2, FBA1, and GPM1).

Vergleichspartner (DBSCAN, PROCLUS, DOC) fanden keine signifikanten funktionellen Zusammenhänge

3.3 Correlation Clustering

Correlation Clustering

- Integration von Covarianzanalyse in den Clustering-Prozess
- Jeder Cluster enthält Punkte einheitlicher, beliebig dimensionaler Korrelation (Covarianz)
- Verfahren:
  - ORCLUS (basiert auf k-medoid Verfahren)
  - 4C (basiert auf DBSCAN)
  - HICO (basiert auf OPTICS)



3.3.1 ORCLUS

ORCLUS [Aggarwal & Yu 2000]

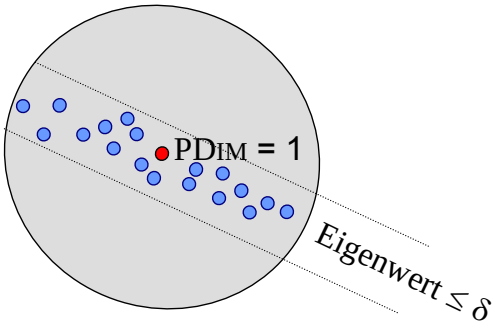
- Ähnlich wie PROCLUS aber Covarianzanalyse
- Berechnet Unterräume, die nicht achsenparallel sind
- Verwendet *k*-Means statt *k*-Medoid Ansatz
- Verwendet Korrelationsmatrix, um für jeden Cluster die Eigenvektoren mit den kleinsten Eigenwerten zu berechnen (Covarianz des Clusters)
- Eingabe: *k* Anzahl der Cluster, *l* durchschnittliche Dimensionalität der Cluster-Unterräume
- Ergebnis: *k* Cluster (C_i) mit zugeordneten Eigenvektoren (S_i)
- Probleme:
  - Nachteile von *k*-Means
  - Input-Parameter *l*
  - Laufzeit: O ($l^3 + l \cdot n \cdot d + l^2 + d^3$)

4C [Böhm, Kailing, Kröger, Zimek 2004]

Intuition: Integration der Korrelationsanalyse in dichtebasiertes Clustering.
4C = Compution Correlation Connected Clusters

Idee:

- Bestimmung der Nachbarschaftspunkte (euklidisch)
- Ermittlung und Zerlegung (PCA) der Kovarianz-Matrix



PCA zerlegt Kovarianzmatrix M_P in:

$$M_P = V\ E\ V^T$$

V: Eigenvektor-Matrix (Haupt-Ausdehungs-Richtungen)

E: Eigenwert-Matrix (Varianzen)

Quadratische Form als Distanzfunktion:

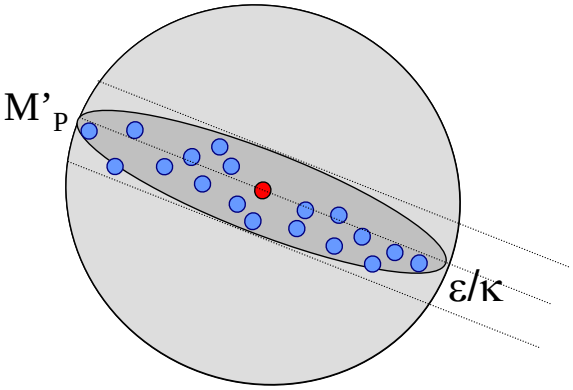
$$dist_{M_P}(P,Q) = \sqrt{(P-Q)*M_P*(P-Q)^T}$$

Korrelations Ähnlichkeits Matrix:

$$M'_P = V\ E'\ V^T$$

Anpassung der Eigenwerte:

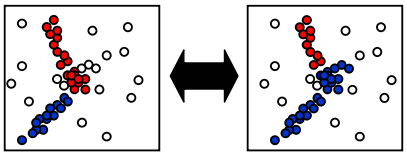
$$e'_i = \begin{cases} 1 & if \quad e_i > \delta \\ \kappa & if \quad e_i \leq \delta \end{cases}$$

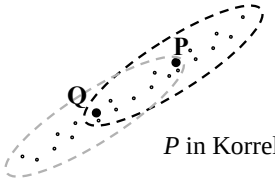


κ steuert die tolerierte Abweichung („thickness“) der λ-dimensionalen Korrelations-Linie oder -Ebene)

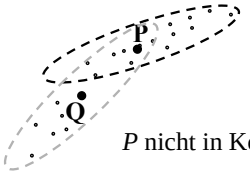
3.3.2 4C

- Problem der Reihenfolge-Unabhängigkeit:
  - Punkte müssen sich gegenseitig finden
  - *P* Element der Korrelations-ε-Nachbarschaft von *Q* gdw. $dist_{M'}(P,Q) \leq \varepsilon$ **und** $dist_{M'}(Q,P) \leq \varepsilon$





P in Korrelations-ε-Nachbarschaft von *Q*



P nicht in Korrelations-ε-Nachbarschaft von *Q*

- Definitionen zu:
  - Correlation Core Object
  - Direct Correlation Reachability
  - Correlation Reachability
  - Correlation-Connected Set

  analog wie DBSCAN/PreDeCon

3.3.2 4C

Algorithmus 4C (*DB*, ε, μ , λ, δ)

// assumption: each point in *DB* is marked
// as unclassified

for each unclassified $O \in DB$ **do**

compute $N_\varepsilon(O)$;

if $|N_\varepsilon(O)| \geq \mu$ **then**

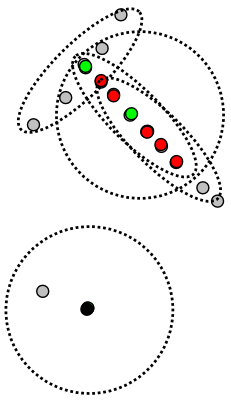
if CorDim($N_\varepsilon(O)$) ≤ λ **then**

if $|N_\varepsilon^{M'_O}(O)| \geq \mu$ **then**

expand a new cluster**;**

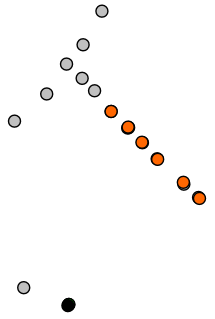
In all other cases: mark *O* as noise;

μ = 3



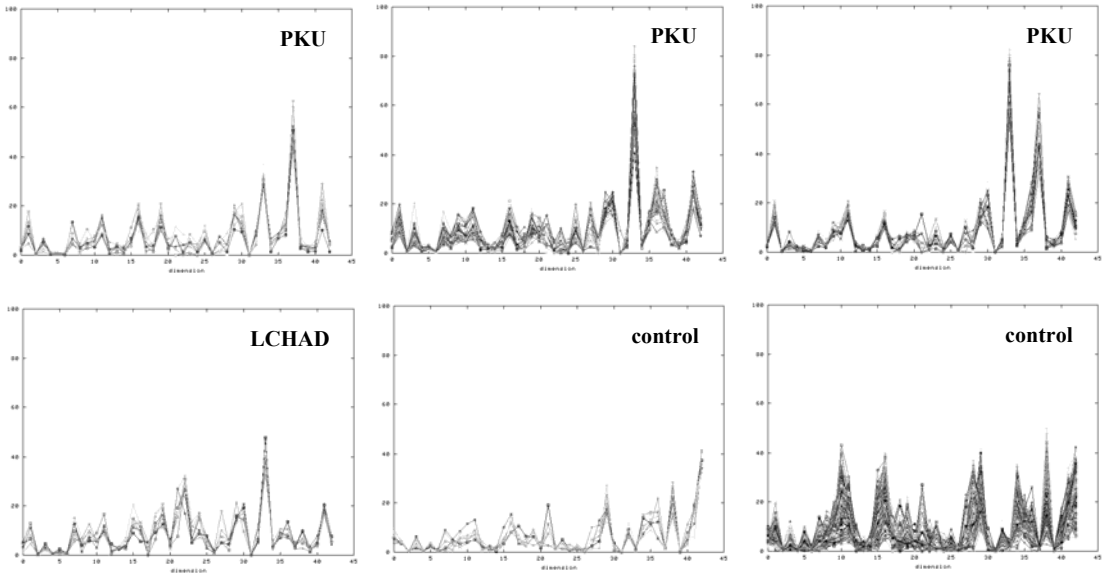
3.3.2 4C

//expand cluster
generate new clusterID;
Insert all *X* with Dir*CorReach*(O,X) into queue Φ;
while $\Phi \neq \varnothing$ **do**
  Q = first point in Φ;
  compute $N_{\varepsilon}^{M'_Q}(Q)$;
  for each *X* with Dir*CorReach*(Q,X) **do**
    if *X* is unclassified or noise **then**
      assign current clusterID to *X*;
    if *X* is unclassified **then**
      insert *X* into Φ;
    remove *Q* from Φ;



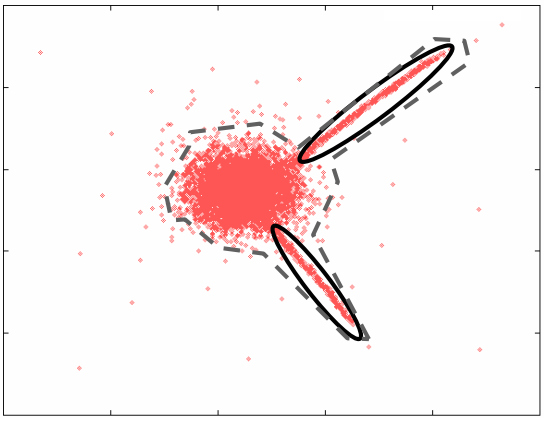
3.3.2 4C

Ergebnisse auf Metabolome Daten



3.3.2 4C

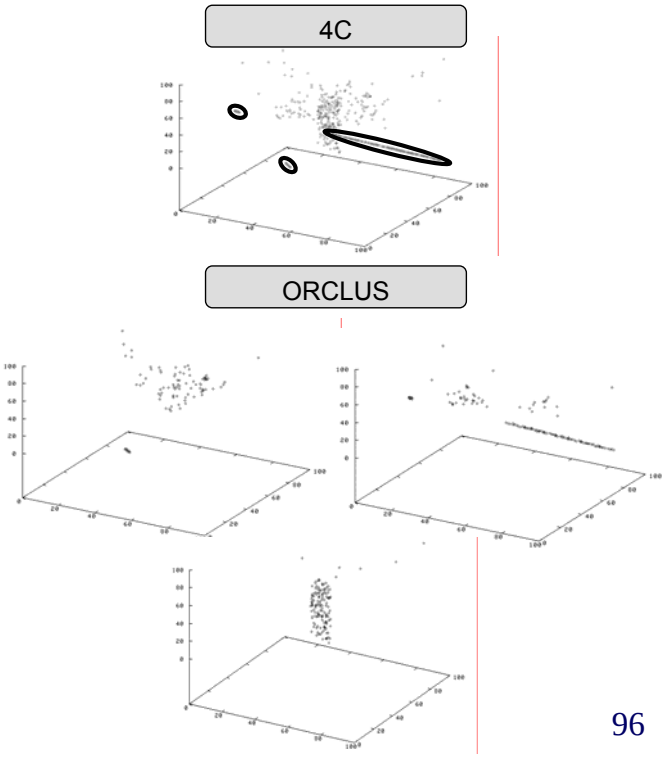
Vergleich mit DBSCAN



Cluster found by DBSCAN

Clusters found by 4C

Vergleich mit ORCLUS



3.3.2 4C

Komplexität ohne Indexunterstrützung:

- Für jeden (Kern-) Punkt ist das zugeordnete Ähnlichkeitsmaß (die modifizierte Kovarianzmatrix) zu ermitteln:
  - Ermittlung der Kovarianzmatrix: $O(nd^2)$
  - Eigenwert-Zerlegung der Kovarianzmatrix: $O(d^3)$
- DBSCAN wertet je eine Bereichsanfrage pro Punkt aus:
  - Auswertung mit modifizierter Kovarianzmatrix: $O(nd^2)$
- Gesamtkomplexität: $O(n^2d^2+d^3n)$

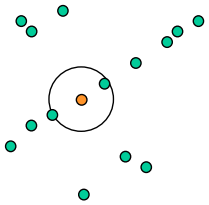
Komplexität mit Indexunterstützung:

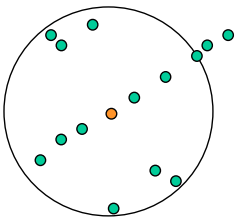
- Bereichsanfrage reduziert sich auf $O(d^2 \log n)$
- Gesamt-Komplexität: $O(d^2n \log n + d^3n)$

3.3.2 4C

Diskussion

- Finden lokaler Korrelations-Cluster
- Kritisch: Parameter ε bestimmt, wieviele und welche Punkte aus der Nachbarschaft für die Bestimmung der Kovarianzmatrix verwendet werden => Kovarianzmatrix (und damit die PCA) nicht stabil





- Verbesserung [Achtert,Böhm,Kröger,Zimek 2006]
  - *k*-nächste Nachbarn statt Range-Query
  - Dadurch zumindest Anzahl der Punkte festgelegt
  - z.B. $k = 10d$

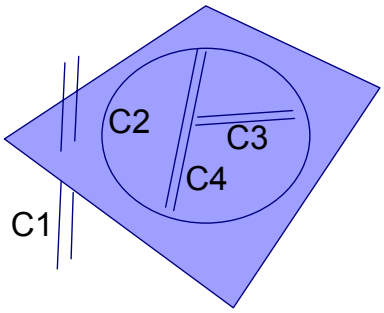
3.3.3 HICO

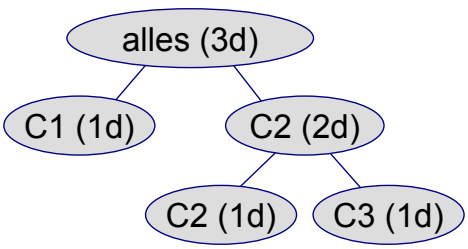
HICO [Achtert, Böhm, Kröger, Zimek 2006]

Motivation:

Korrelations-Cluster können Hierarchie bilden:

- Mehrere Cluster in je einem Korrelations-Subspace
- In diesem Teilraum können weitere Korrelations-Cluster mit noch niedrigerer Dimension sein



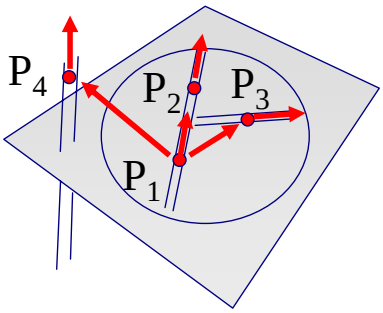


Idee von HICO (Hierarchical Correlation Clustering):

- Wieder Ermittlung von Eigenwerten und Eigenvektoren aus der Nachbarschaft eines Punktes (euklidische k-Nächste-Nachbarn)
- Integration in OPTICS durch ein geeignetes, korrelationsbasiertes Distanzmaß:
  - Distanz zwischen 2 Punkten soll niedrig sein, wenn sich die Punkte auf einer gemeinsamen Korrelationslinie befinden
  - Distanz soll höher sein, wenn es sich um eine 2d-Korrelationsebene handelt
  - usw.

  Distanz: Dimension des „aufgespannten“ Raumes aus:
  - Eigenvektoren, aus den Kovarianz-Matrizen der beiden Punkte
  - Differenzvektor der beiden Punkte

Beispiel:



$dist(P_1,P_2) = 1$
$dist(P_1,P_3) = 2$
$dist(P_1,P_4) = 3$

Dimension des aufgespannten Raums:

- Nicht im algebraischen Sinn
- Sondern in dem Sinn:
  Hinreichende Anzahl von Dimensionen ist hinreichend „flach“, d.h. kleine Varianz

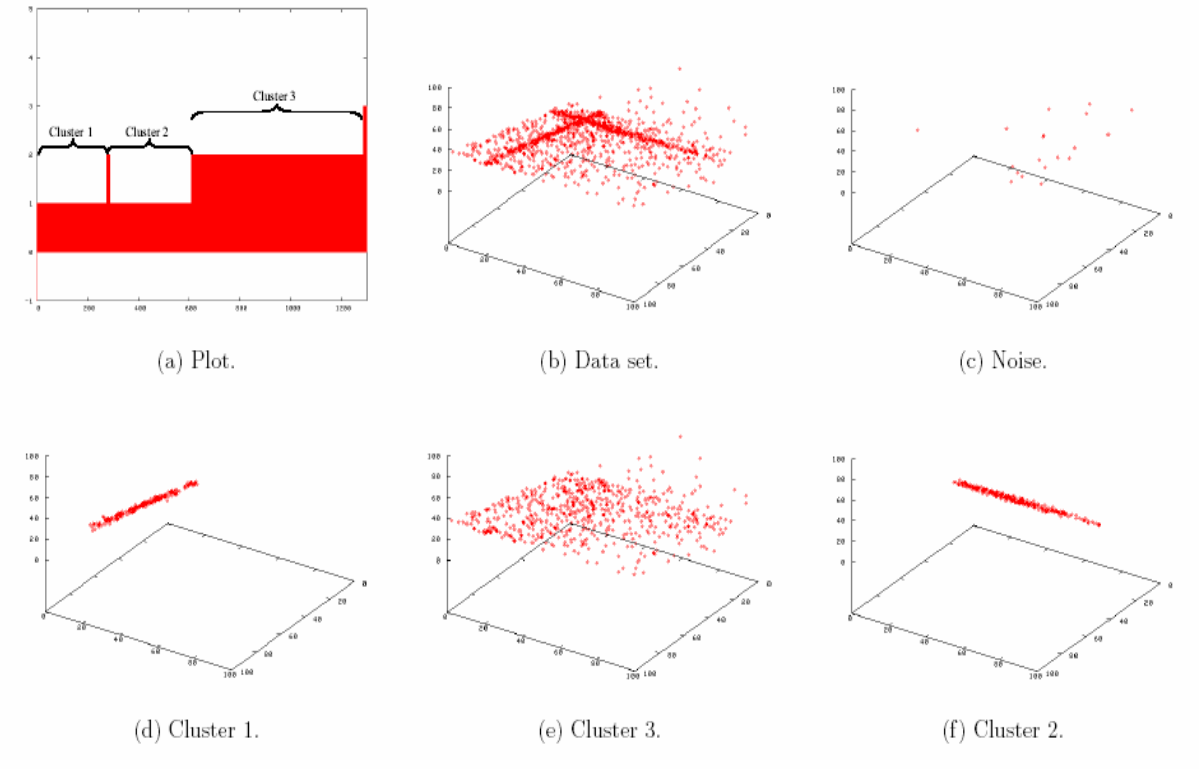
Distanz-Ermittlung mit Schmidt´scher Orthonormierung

3.3.3 HICO

Distanzermittlung:

- Sei M_P die Menge aller Eigenvektoren von P mit Eigenwerten > δ
- Für jeden Eigenvektor q von Q mit Eigenwert > δ:
  - Teste, ob q von der durch M_P aufgespannten Hyper-Ebene einen Abstand > δ hat
  - Wenn ja, nimm q zu M_P hinzu und orthonormiere M_P
- Teste den Differenzvektor (P–Q), ob er von M_P einen Abstand > δ hat
- Korrelations-Distanzen sind ganzzahlig
- Deshalb haben viele Punkte exakt gleiche Distanz
- Beim Ordnen der Distanzen in der Seed-Liste deshalb zwei Ordnungs-Kriterien:
  - Korrelations-Distanz (erstes Ordnungs-Kriterium)
  - Euklidische Distanz (bei *gleicher* Korrelations-Distanz)
- Hierdurch innerhalb einer Korrelations-Linie:
  Ermittlung der konventionellen Cluster-Struktur

3.3.3 HICO



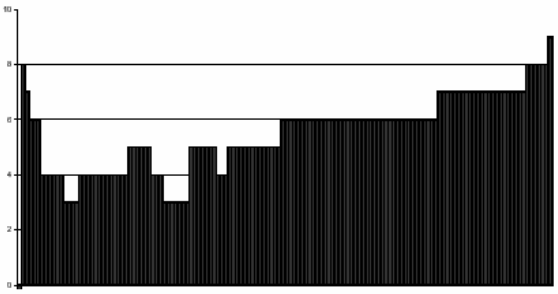
(a) Plot. (b) Data set. (c) Noise.

(d) Cluster 1. (e) Cluster 3. (f) Cluster 2.

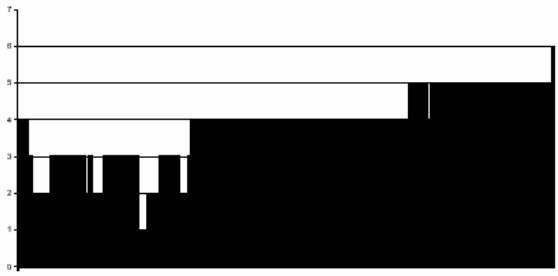
3.3.3 HICO

(a) "NHL" data set ($k = 30$).

(b) "Soccer" data set ($k = 25$).



(c) "COIL" data set ($k = 25$).

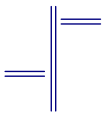


(d) "El Nino" data set ($k = 40$).

Projected Clustering und Correlation Clustering

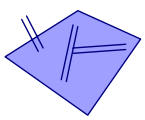
Überblick und Fazit:

- PROCLUS/PreDeCon:
  - Achsenparallele Subspace-Cluster



- ORCLUS/4C:
  - Beliebig orientierte Korrelations-Cluster

- HICO:
  - Hierarchien von Korrelations-Clustern



- Dichtebasierte Methoden effizienter und effektiver als partitionierende Verfahren (vgl. Clustering Kap. 2)
- Allgemeine Vorteile:
  - Determiniertes Ergebnis
  - Robust gegenüber Rauschen
  - Quadratisch in *n*, maximal kubisch in *d*
  - Parametrisierung einfacher (besonders HICO)

3.4 Subspace Clustering

Beobachtung:

- Projected Clustering Methoden finden Cluster in verschiedenen Unterräumen
- ABER: Punkte können in verschiedenen Unterräumen in verschiedenen Clustern liegen („überlappende Cluster“)

Idee des Subspace Clusterings:

- Berechne Clustering für mehrere Unterräume
- Vollständige Suche ist nicht effizient (O (2^d) mögliche Unterräume)
- Daher: Finde alle (möglichst viele) Unterräume, in denen Cluster liegen

Bemerkung:

Terminologie nicht immer einheitlich. Hier:

- Projected Clustering = keine überlappenden Cluster
- Subspace Clustering = überlappende Cluster

3.4.1 CLIQUE

CLIQUE [Agrawal, Gehrke, Gunopulos, Raghavan 1998]

1. Identifikation von Unterräumen mit Clustern
2. Identifikation von Clustern

- *Cluster*: „dichtes Gebiet“ im Datenraum
- Dichte-Grenzwert τ

  Region ist *dicht*, wenn sie mehr als τ Punkte enthält

  Region = Gitterzelle
- Gitterbasierter Ansatz

  jede Dimension wird in ξ Intervalle aufgeteilt

  Cluster ist Vereinigung von verbundenen dichten Regionen

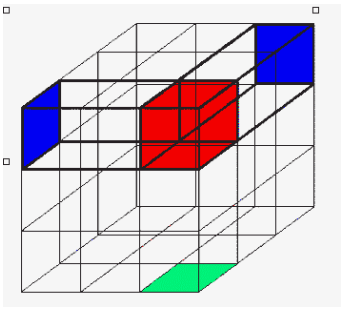
Identifikation von Unterräumen mit Clustern

- Aufgabe: Entdecken dichter Regionen
- Greedy-Algorithmus (Bottom-Up), ähnlich wie *Apriori*-Algorithmus bei der Warenkorbanalyse:
  - beginne mit der leeren Menge
  - nehme jeweils eine Dimension dazu
- Grundlage dieses Algorithmus: *Monotonie-Eigenschaft*

  wenn eine Region *R* im *k*-dimensionalen Raum dicht ist, dann ist auch jede Projektion von *R* in einen (*k*-1)-dimensionalen Unterraum dicht
- Umkehrung:

  Wenn eine (*k*-1)-dimensionale Region *R* nicht dicht ist, sind alle *k*-dimensionalen Regionen, die *R* als Projektion besitzen, nicht dicht.

Beispiel



2-dim. dichte Regionen

3-dim. Kandidaten-Region

2-dim. Region, die geprüft werden muß

- auch die Regionen, die in Unterräumen dicht sind, müssen noch auf der DB gezählt werden (enthalten sie wirklich mehr als τ Punkte?)
- heuristische Reduktion der Anzahl der Kandidaten-Regionen

Identifikation von Clustern

- Aufgabe: Finden maximaler Mengen verbundener dichter Regionen
- Gegeben: alle dichten Regionen in demselben *k*-dimensionalen Unterraum
- „depth-first“-Suche in folgendem Graphen (Suchraum)

  Knoten: dichte Regionen

  Kanten: gemeinsame Hyperflächen / Dimensionen der beiden dichten Regionen
- Laufzeitkomplexität

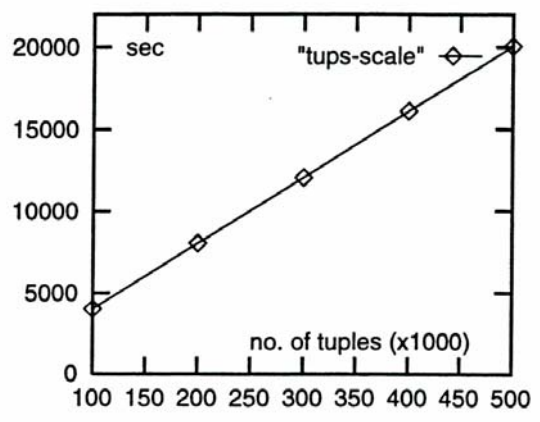
  dichte Regionen im Hauptspeicher (z.B. Hashbaum)

  für jede dichte Region 2 *k* Nachbarn zu prüfen

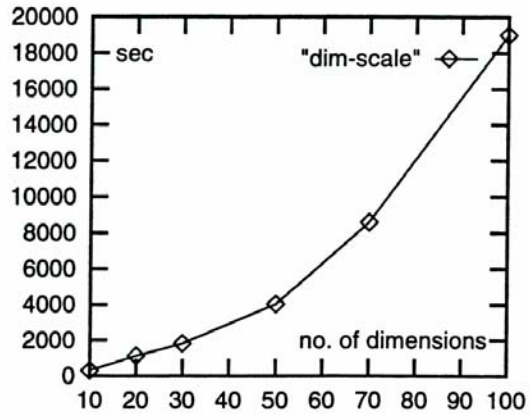
  $\Rightarrow$ Zahl der Zugriffe zur Datenstruktur: O ($2 \cdot k \cdot n$)

Experimentelle Untersuchung

Laufzeit in Abhängigkeit von *n*



Laufzeit in Abhängigkeit von *d*



Laufzeitkomplexität von CLIQUE

linear in *n*, superlinear in *d*

Diskussion

+ automatische Entdeckung von Unterräumen mit Clustern
+ automatische Entdeckung von Clustern
+ keine Annahme über die Verteilung der Daten
+ Unabhängigkeit von der Reihenfolge der Daten
+ gute Skalierbarkeit mit der Anzahl *n* der Datensätze

- Genauigkeit des Ergebnisses hängt vom Parameter ξ ab
- braucht eine Heuristik, um den Suchraum aller Teilmengen der Dimensionen einzuschränken

  $\Rightarrow$ findet u.U. nicht alle Unterräume mit Clustern

Erweiterungen von CLIQUE

- *ENCLUS* [Cheng, Fu & Zhang 1999]
  - Unterschied zu CLIQUE:
    Anderes Dichtekriterium für Regionen: Entropie $H(X)$ fürMenge von Regionen
    $H(X) = -\Sigma_{x \in X} d(x) \cdot \log d(x)$ (d(x) Anteil der Datenpunkte in Region x)
  - Entropie verhält sich ebenfalls monoton

- *MAFIA* [Goil, Nagesh & Choudhary 1999]
  - Adaptives Gitter $\Rightarrow$ weniger Regionen variabler Größe
  - Finde Regionen, die um Faktor α dichter sind als der erwartete Durchschnitt (relativ zum Volumen)
  - Keine Monotonie-Eigenschaft, daher Brute-Force Navigation durch den Suchraum aller möglichen Unterräume

Dichte-verbundenes Subspace Clustering

[Kailing, Kriegel, Kröger 2004]

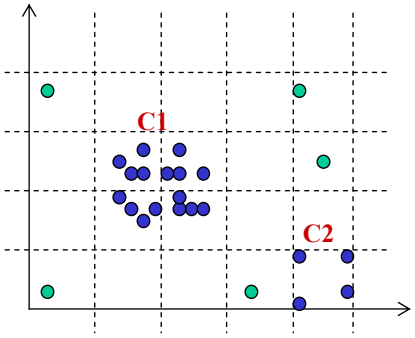
Motivation:

- Nachteil der gitterbasierten Ansätze

Wahl von ξ und τ

Cluster für $\tau = 4$
(ist C2 Cluster?)

Für $\tau > 4$: keine Cluster
(insb. C1 geht verloren!)



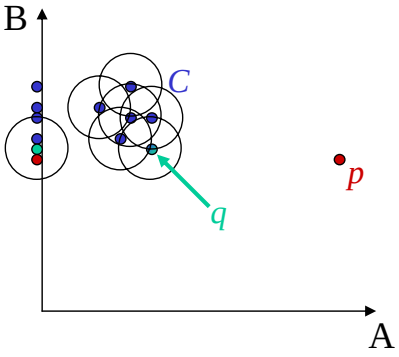
$\Rightarrow$ Verwende dichte-verbundenes Clustering (DBSCAN)

SUBCLU

- Berechne dichte-verbundene Subspace Cluster
- Vorteile:
  - Clusterbegriff mathematisch sauber formuliert
  - Zuordnung der (Kern-) Punkte zum Cluster eindeutig
  - Erkennen von Clustern unterschiedlicher Größe und Form
- Gesucht:
  - Effiziente Strategie, um die dichte-verbundenen Cluster in allen Unterräumen (bzgl. ε und *MinPts*) zu berechnen
  - Nutze Greedy-Ansatz wie bei CLIQUE: generiere bottom-up alle Subspace Cluster
  - Dazu notwendig: Monotoniekriterium für dichte-verbundene Cluster

Monotonie dichte-verbundener Cluster

- Gilt leider nicht:
  - Sei *C* ein dichte-verbundener Cluster im Unterraum *S*
  - Sei $T \subset S$ ein Unterraum von *S*
  - *C* muss nicht mehr maximal bzgl. Dichte-Erreichbarkeit sein
  - Es kann Punkte geben, die nicht in *C* sind, aber im Unterraum *T* dichte-erreichbar von einem Objekt in *C* sind



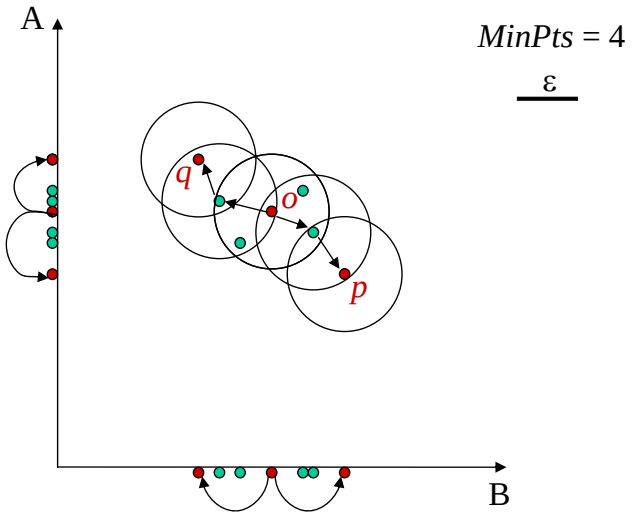
C ist ein dichte-verbundener Cluster im Unterraum {A,B}

$p \notin C$ und $q \in C$

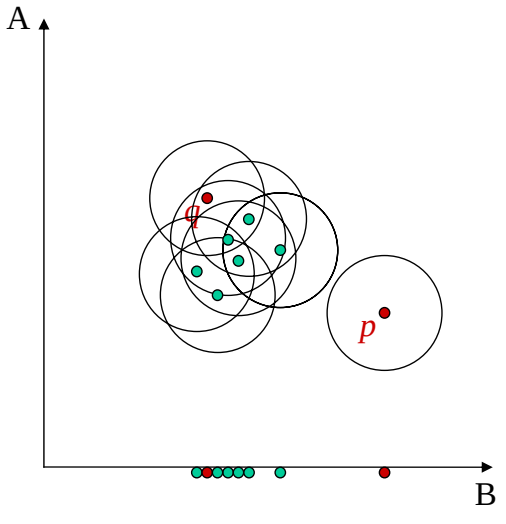
Im Unterraum {B} ist *p* (direkt) dichte-erreichbar von $q \in C$

Monotonie dichte-verbundener Mengen

Wenn *C* eine dichte-verbundene Menge im Unterraum *S* ist, so ist *C* auch eine dichte-verbunden Menge in allen Teilräumen $T \subset S$



p und *q* dichte-verbunden in {A,B}, {A} und {B}



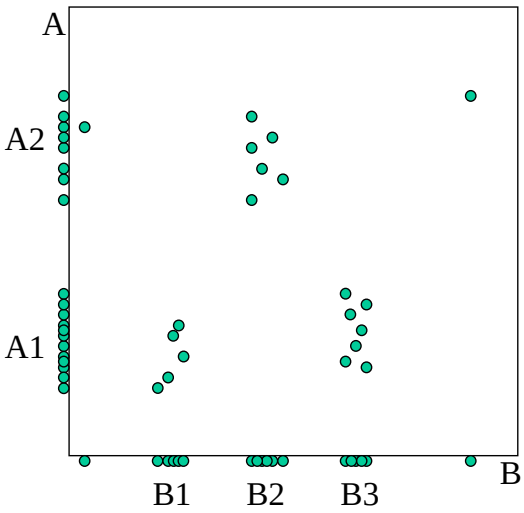
p und *q* nicht dichte-verbunden in {B} und {A,B}

Algorithmus

- Generiere alle 1-dimensionalen dichte-verbundenen Cluster
- Für jeden *k*-dimensionalen Cluster muss nun geprüft werden, ob er in einem (*k*+1)-dimensionalen Oberraum noch vorhanden ist:
  - Gegeben:
    - S*k*: Menge der *k*-dimensionale Unterräume in denen Cluster existieren
    - C*S*: Menge der Cluster im Unterraum *S*
    - C*k*: Menge aller Mengen von Cluster in *k*-dimensionalen Unterräumen

      C*k* = {C*S* | *S* ist *k*-dimensionaler Unterraum}
  - Vorgehen:
    - Bestimme (*k*+1)-dimensionale Kandidatenunterräume *Cand* aus S*k*
    - Für einen beliebigen *k*-dimensionalen Unterraum *U*⊂*Cand* :

      Bestimme für alle *k*-dimensionalen Cluster *c* in *U* (c∈C*U*) die (*k*+1)-dimensionalen Fortsetzungen durch die Funktion DBSCAN(*c*, *U*, ε, *MinPts*)



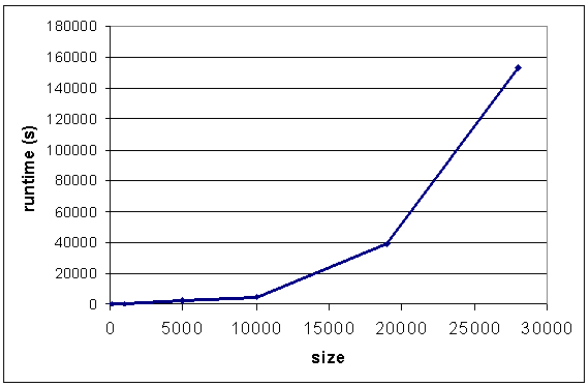
Funktion DBSCAN(*D*, *U*, ε, *MinPts*) berechnet alle dichte-verbundenen Cluster bzgl. ε und *MinPts* einer Datenmenge *D* im Unterraum *U*

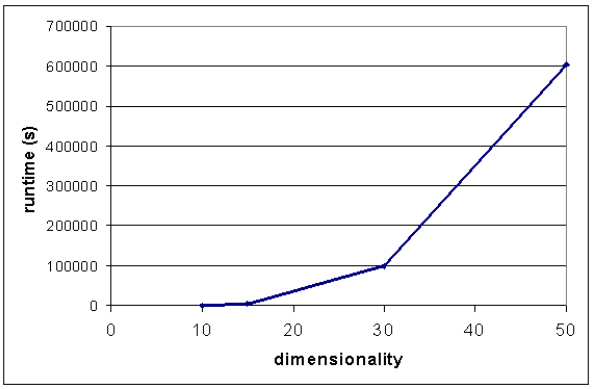
S1 = {{A}, {B}}
C{A} = {A1, A2}
C{B} = {B1, B2, B3}
C1 = {C{A}, C{B}}

- Heuristische Optimierungsmöglichkeit:
  - DBSCAN(*c*, *U*, ε, *MinPts*) nicht für zufälligen *U* ⊂ *Cand* aufrufen, sondern für den Unterraum *U*, in dem die Gesamtanzahl der Punkte in den Clustern (also der Punkte in C*U*) am geringsten ist (im Beispiel: *U* = {B})
  - Dadurch wird die Anzahl der Range-Queries beim DBSCAN-Lauf minimiert (im Beispiel um 2)

3.4.2 SUBCLU

Experimente





Skalierbarkeit: superlinear in Anzahl der Dimensionen und Anzahl der Objekte

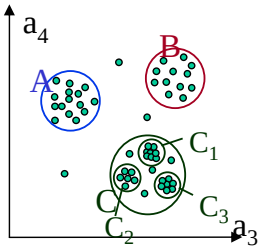
ABER: Findet mehr Cluster als CLIQUE 

3.4.3 RIS

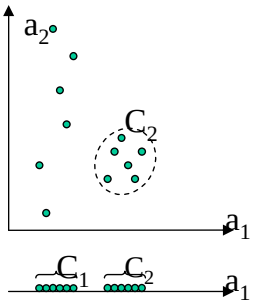
RIS (Ranking Interesting Subspaces) [Kailing, Kriegel, Kröger, Wanka 2003]

Probleme von SUBCLU:

- Verschiedene Cluster in einem Unterraum können verschieden dicht sein



- Cluster aus verschiedenen Unterräumen können verschieden dicht sein



3.4.3 RIS

Idee von RIS:

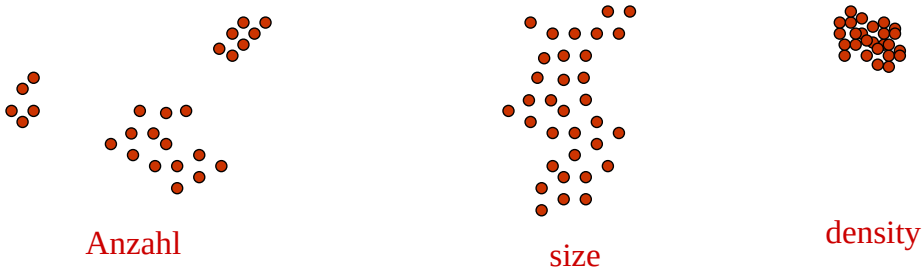
- Berechne nicht mehr direkt die Subspace Cluster
- Sondern: berechne nur die Unterräume, die interessante Cluster enthalten
  - Was sind interessante Cluster/Unterräume?
  - Qualitätskriterium für Unterräume
- RIS gibt eine Liste von Unterräumen aus, sortiert nach Qualität
- Die eigentlichen Cluster können durch ein beliebiges Cluster-Verfahren für die interessanten Unterräume erzeugt werden

3.4.3 RIS

Interessante Unterräume:

- Cluster enthalten mindestens einen Kernpunkt

  $\Rightarrow$ Unterraum, der keinen Kernpunkt enthält, kann nicht interessant sein
- Anzahl der Kernpunkte ist proportional zur
  - Anzahl der verschiedenen Cluster und/oder
  - Größe der Cluster und/oder
  - Dichte der Cluster



Algorithmus RIS:

1. Berechne für jeden Punkt *p* der Datenbank die Unterräume, in denen *p* noch Kernpunkt ist
   ⇒ Berechnet alle relevanten Unterräume

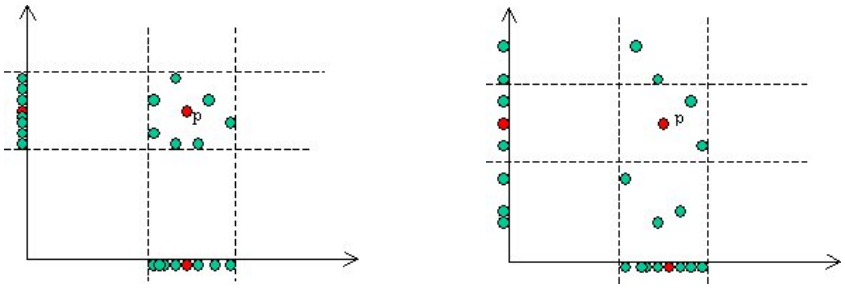
2. Sammle für jeden berechneten Unterraum statistische Informationen um über die „Interessantheit“ des Unterraumes entscheiden zu können
   ⇒ Qualität der Unterräume (z.B. Anzahl der Kernpunkte)
   ⇒ Sortierung der Unterräume nach „Interessantheit“ möglich

3. Entferne Unterräume, die redundante Informationen enthalten
   ⇒ Cluster in einem Unterraum S sind in allen Unterräumen T ⊆ S enthalten

Schritt 1

Suche Unterräume, die mindestens einen Kernpunkt enthalten:

- Monotonie der Kernpunkteigenschaft:
  Wenn *p* ein Kernpunkt in Featureraum *S* ist, dann ist *p* auch ein Kernpunkt in allen Unterräumen $T \subseteq S$



Wenn *p* in *T* kein Kernpunkt ist, kann *p* auch in allen $S \supset T$ kein Kernpunkt sein.
⇒ Suchstrategie von CLIQUE und SUBCLU wieder verwendbar

Schritt 2

Qualität der gefundenen Unterräume:

- count[S] = Summe (der Anzahl) aller Punkte, die in der ε-Nachbarschaft aller Kernpunkte eines Unterraumes S liegen
- NaiveQuality(S) = count[S] – Kernpunkte(S)
  - Anzahl der erwarteten Punkte in einer ε-Nachbarschaft sinkt mit steigender Dimension
  - NaiveQuality favorisiert niedrig dimensionale Unterräume
- Skalierung in Abhängigkeit der Dimensionalität:

$$\text{Quality}(S) = \frac{\text{count}[S] - \text{Kernpunkte}(S)}{n(n-1)(\frac{2\epsilon}{\text{Attr.bereich}})^{\dim(S)}}$$

  - Periodische Randbedingungen um Punkte, die am Rand des Datenraumes liegen, nicht zu benachteilen

Schritt 3

Entfernen redundanter Unterräume:

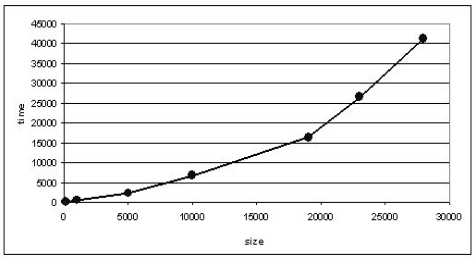
- „Überflüssige“ Unterräume:
  - Cluster im Raum S haben eine Projektion in Unterräumen von S
  - Durch die Hinzunahme von irrelevanten Dimensionen muss ein Cluster zunächst noch nicht verschwinden
- Pruning-Schritte:
  - Abwärts-Pruning:

    Wenn es einen (k-1)-dimensionalen Unterraum S mit einer höheren Qualität als ein k-dimensionaler Unterraum T ($T \subset S$) gibt, lösche T.
  - Aufwärts-Pruning:

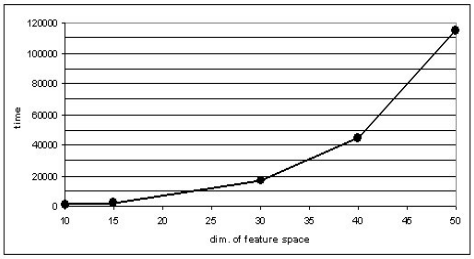
    Wenn der Count-Wert eines echten (k-1)-dimensionaler Unterraumes von S „besonders stark“ vom Mittelwert der Count-Werte aller echten (k-1)-dimensionalen Unterräume von S abweicht, lösche S

Experimentelle Untersuchung

Laufzeit in Abhängigkeit von *n*



Laufzeit in Abhängigkeit von *d*



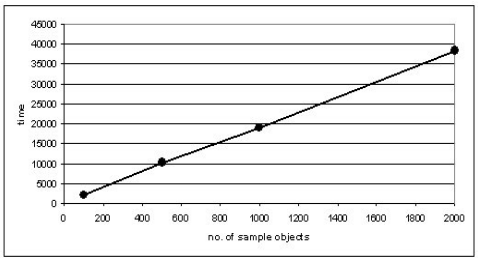
Skaliert superlinear in *n* und *d*

⇒ Random Sampling

auch bei kleinen Samplegrößen

hohe Qualität

Laufzeit in Abhängigkeit der Samplegröße



Diskussion

Vorteile:

- Findet alle Unterräume, in denen interessante Cluster vorhanden sind
- Erzeugen von Subspace Clustern unterschiedlicher Dichte möglich (z.B. indem man in den gefundenen Unterräumen mit OPTICS „clustert“)

Nachteile:

- Problem, das Cluster in verschieden dimensionalen Unterräumen meist unterschiedlich dicht sind, ist immer noch nicht gelöst
- Trotz Dimensions-Anpassung des Qualitätskriteriums:

  ε begrenzt die Dimension der gefunden Unterräume nach oben:

  je kleiner ε desto niedriger dimensional die Unterräume, die gefunden werden

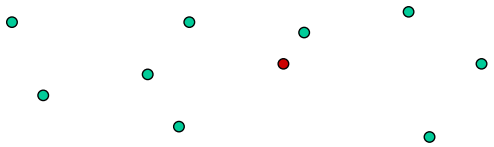
3.4.4 SURFING

SURFING *(Subspaces Relevant for Clustering)* [Baumgartner, Kailing, Kriegel, Kröger, Plant 2004]

Idee: Berechne interessante Unterräume

- Unabhängigkeit von einem globalen Dichteparameter für verschiedene Cluster und verschiedene Unterräume
- ohne die dichte-basierte Vorstellung von Clustern komplett aufzugeben
- OPTICS:
  - Unabhängig von einem globalen Dichteparameter
  - Dichte-basiertes Cluster-Modell
  - Kerndistanz (Distanz zum *k*-nächsten Nachbarn) und Erreichbarkeitsdistanz als Maß für lokale Dichte
  - Je kleiner Kerndistanz, desto dichter sind die Punkte lokal
  - Je größer Kerndistanz, desto weniger dicht sind die Punkte lokal

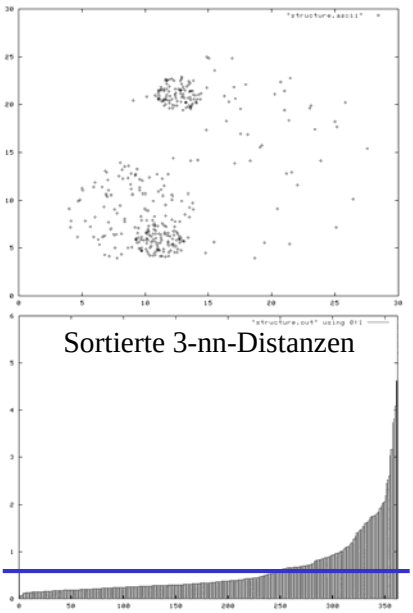
Kleine 10-nächste Nachbarn Distanz

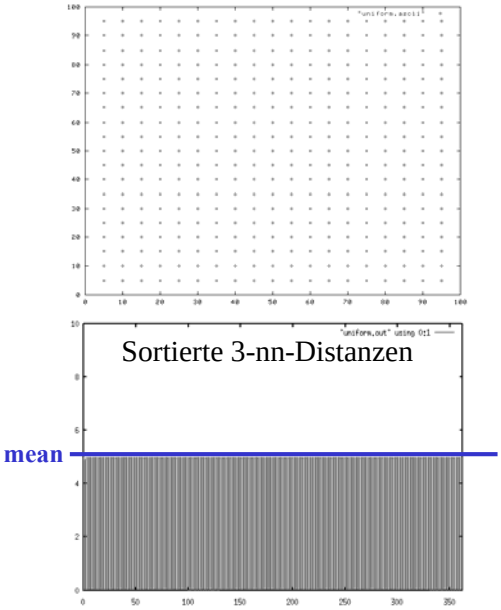


Große 10-nächste Nachbarn Distanz

3.4.4 SURFING

- Die Qualität der hierarchischen Clusterstruktur eines Unterraumes kann anhand der *k*-nn-Distanzen aller Punkte vorhergesagt werden:
  - Viele unterschiedliche *k*-nn-Distanzen ⇒ signifikante (hierarchische) Clusterstrukturen
  - Viele ähnliche *k*-nn-Distanzen ⇒ kaum (hierarchische) Clusterstrukturen





Qualititätskriterium für Unterräume

- Varianz der *k*-nn-Distanzen in einem Unterraum:
  - Nachteil: berücksichtigt die quadrierten Differenzen zum Mittelwert
- Summe der Differenzen *DIFF* unterhalb des Mittelwertes μ:

$$DIFF = \sum_{o \in DB} |\mu - nnDist_k(o)|$$

  - Nachteil: nicht unabhängig von der Dimension
- Verhältnis aus *DIFF* zum Mittelwert μ:
  - Nachteil: Mittelwert ist nicht vollständig robust gegenüber Ausreißern und kleinen sehr dichten Clustern
    - Mittelwert wird durch einige wenige Ausreißer nach oben verschoben
      ⇒ *DIFF* unverhältnismäßig hoch ⇒ *DIFF*/μ unverhältnismäßig zu hoch
    - Mittelwert wird durch wenige kleine sehr dichte Cluster nach unten verschoben
      ⇒ *DIFF* unverhältnismäßig klein ⇒ *DIFF*/μ unverhältnismäßig zu klein

⇒ Skalierung mit der Anzahl der Punkte, deren k-nn-Distanz unterhalb des Mittelwertes liegt (bezeichnet als *Below*)

Qualität eines Unterraums:

$$quality(S) = \begin{cases} 0 & \text{if } Below_S = \{\} \\ \dfrac{DIFF_S}{|Below_S|\mu_S} & \text{else.} \end{cases}$$

$DIFF_S$ = Summe der Differenzen der k-nn-Distanzen unterhalb von μ zum Mittelwert (vgl. Folie 318) im Unterraum S

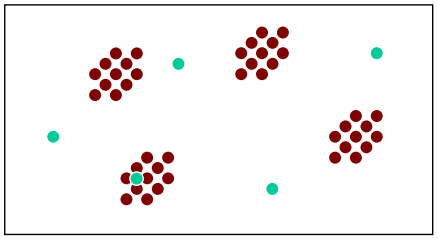
μ_S = Mittelwert der *k*-nn-Distanzen im Unterraum S

$Below_S$ = Anzahl der Punkte, die im Unterraum S eine *k*-nn-Distanz unterhalb von μ_S haben

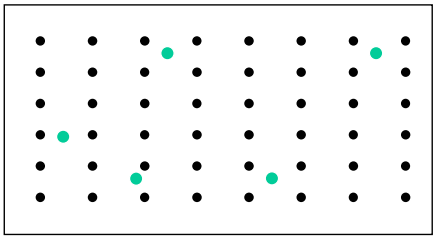
3.4.4 SURFING

Problem:

Interessante Unterräume mit Clustern gleicher Dichte und ohne Noise können nicht von irrelevanten Unterräumen mit gleichverteilten Daten unterschieden werden!



Interessanter Unterraum

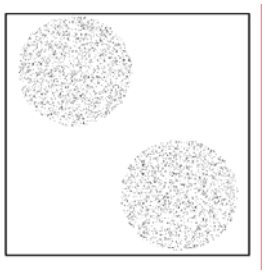


Irrelevanter Unterraum

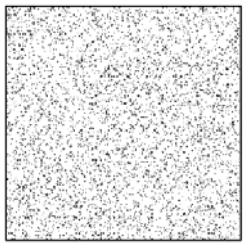
Lösung:

Füge zufällige eine kleine Menge an zusätzlichen Punkten ein bevor die Qualität berechnet wird

3.4.4 SURFING



data set A



data set B

eingefügte Punkte	quality(A)	quality(B)
0 %	0.13	0.15
0.1 %	0.15	0.15
0.5 %	0.31	0.15
1 %	0.38	0.15
5 %	0.57	0.15
10 %	0.57	0.15

Empirische Ergebnisse:

– 1% eingefügte Punkte reichen aus

– Einfügung nur wenn nötig:

Wenn Qualität eines *l*-dimensionalen Unterraums geringer ist als die quality einer *l*-dimensionalen Gauss-Verteilung

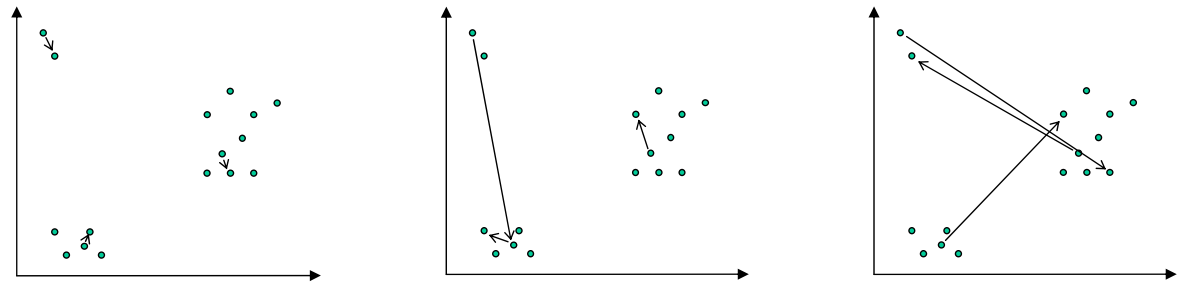
Algorithmus SURFING

- Qualitätskriterium ist nicht monoton!!!
- ABER: Qualität steigt, wenn relevante Attribute hinzu kommen bzw. sinkt, wenn irrelevante Attribute hinzukommen
- Bottom-up Unterraum Generierung ähnlich wie *Apriori*, aber kein Pruning bei der Kandidatengenerierung

  $\Rightarrow$ mehr Kandidaten in jeder Iteration zu Testen
- Heuristisches Pruningkriterium um möglichst viele Unterräume zu löschen (dadurch wird Anzahl der Kandidaten reduziert)
- Komplexität: $O(N^2 \cdot m)$ m = # generierter Unterräume

Parameterwahl

- SURFING hängt nur noch von k ab!!!
- Wahl von k relativ einfach:



k = 1 (zu klein) k = 5 (richtig) k = 10 (zu groß)

Durch zufälliges Einfügen wird die Wahl von k nocheinmal deutlich unkritischer!

3.4.4 SURFING

Fazit

- SURFING ist dank der Pruning-Heuristik sehr effizient (meist werden nur knapp 1% aller möglichen Unterräume erzeugt)
- SURFING ist mehr oder weniger parameterfrei (Wahl von k relativ einfach und bei großen, hochdimensionalen Daten typischerweise nicht kritisch)
- SURFING erzielt (in Zusammenarbeit mit OPTICS) bessere experimentelle Ergebnisse als CLIQUE, SUBCLU oder RIS, speziell wenn:
  - Cluster in stark verschieden dimensionalen Unterräumen existieren
  - Hierarchische und unterschiedlich dichte Cluster existieren

3.4.5 FIRES

Filter-Refinement Subspace Clustering

[Kriegel, Kröger, Renz, Wurst 2005]

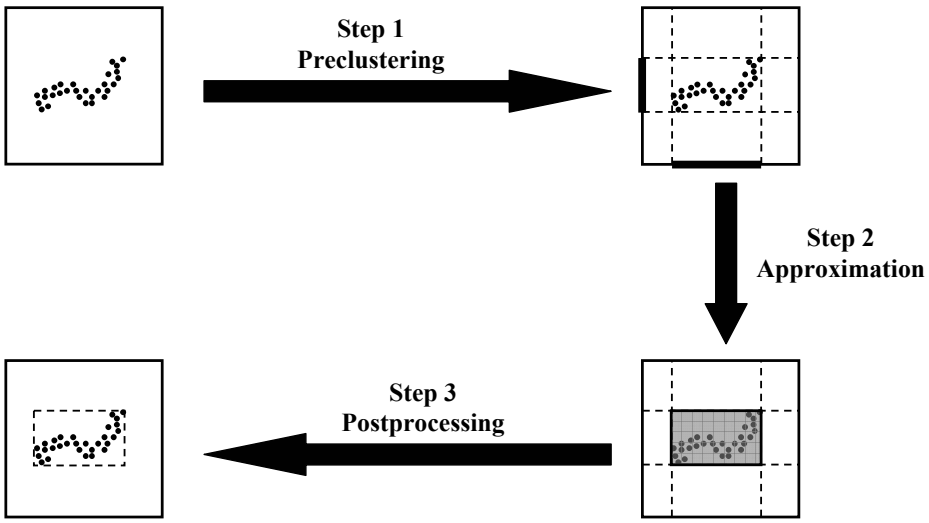
Probleme bisheriger Ansätze

- Globales Dichte-/Clusterkriterium
  - Niedrig dimensionale Unterräume vs. höher dimensionale Unterräume
  - Unterschiedlich dichte Clusters in einem gemeinsamen Unterraum
- Skalierbarkeit (*Apriori*-basierte Algorithmen):
  - Exponentiell bzgl. Dimension des Datenraumes und/oder
  - Exponentiell bzgl. Dimension des Cluster-Teilraumes
- Unvollständigkeit (einfache Heuristiken zur Teilraumsuche)
  - Unvollständige Resultate
- Subspace Ranking ⇔ Subspace Clustering
  - Viele Anwendungen benötigen explizite Berechnung der Cluster (z.B. zur automatischen Weiterverarbeitung)

3.4.5 FIRES

Filter-Refinement Subspace Clustering (FIRES)

- Preclustering: berechne alle 1D Cluster („Basiscluster")
- Approximiere Subspace Cluster: verschmelze Basiscluster
- Postprocessing: verfeinere Subspace Cluster Approximationen



3.4.5 FIRES

Preclustering

- Berechne alle Cluster in allen 1D Projektionen
- Verwende beliebegen Clusteringalgorithmus
- Resultat: „Basiscluster"

Approximation der Subspace Cluster

- Theoretisch: $O(2^n)$
- Heuristik:
  - Finde Gruppen von „ähnlichen" Basisclustern, d.h. clustere Basiscluster mit
    DBSCAN [Ester,Kriegel,Sander,Xu KDD'96]
    SNN [Ertötz,Steinbach,Kumar SIAM DM'03]
- Ähnlichkeit zwischen zwei Basisclustern c1 und c2 ???
  - Idee: Objekte im Teilraum *S* ähnlich => auch in allen Attributen, die *S* aufspannen, ähnlich => Maximiere Schnittmenge

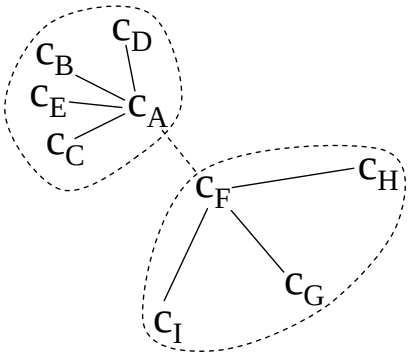
$$sim(c1, c2) = |\, c1 \cap c2 \,|$$

3.4.5 FIRES

- Theoretisch möglich: DBSCAN-Clustering der Basiscluster
  - ABER: *sim* bevorzugt große Cluster => niedrig dimensionale Teilräume
  - Gewünscht: viele Basiscluster verschmelzen => höher dimensionalere Teilräume
  - Verwende daher ein SNN-Ansatz auf Basisclustern

k-most similar clusters (*k*MSC)

- *k*MSC(c1): die *k* ähnlichsten Basisclusters (bzgl. *sim*) zu c1
- Verschmelze die Basiscluster, die möglichst ähmliche *k*MSC haben
- Beispiel (*k* = 4):
  $c_A \in k\text{MSC}(c_F)$
  $c_F \notin k\text{MSC}(c_A)$
  => c_F passt nicht zu $k\text{MSC}(c_A)$



3.4.5 FIRES

Best Merge Candidates (BMC) eines Clusters c1

- $BMC(c1) = \{c2 \mid Card(k\text{MSC}(c1) \cap k\text{MSC}(c2)) \geq \mu\}$ wobei $\mu < k$

Best Merge Cluster

- c1 ist *Best Merge Cluster* wenn *Card*(BMC(c1)) ≥ *minClu*

Algorithmus zum Verschmelzen von Basisclustern

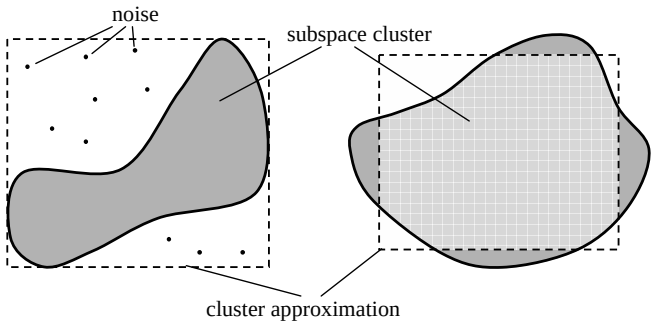
- SNN Algorithmus auf Basisclustern
- Output: Menge von Cubspace Cluster Approximationen
- Schritte:
  - Berechne alle Best Merge Cluster
  - Verschmelze Best Merge Cluster, die sich gegenseitig in ihren BMC Mengen finden
  - Weise Basiscluster, die keine Best Merge Cluster, aber in den BMCs eines Best Merge Clusters sind den entsprechenden Approximationen zu

3.4.5 FIRES

Cluster Refinement

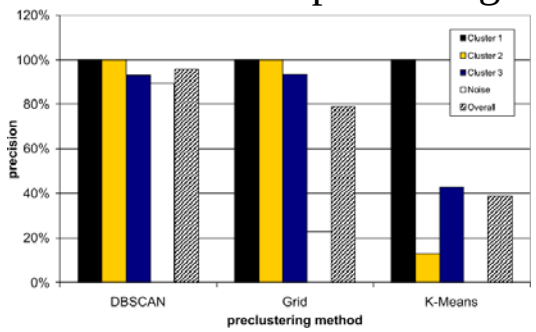
- Einfaches Verschmelzen der Basisclusters erzeugt typischerweise nur sehr grobe Approximationen der Subspace Cluster
- Wende einen beliebigen Clustering Algorithmus auf die Vereinigung der verschmolzenen Basiscluster an

  z.B. DBSCAN mit angepassten Parametern

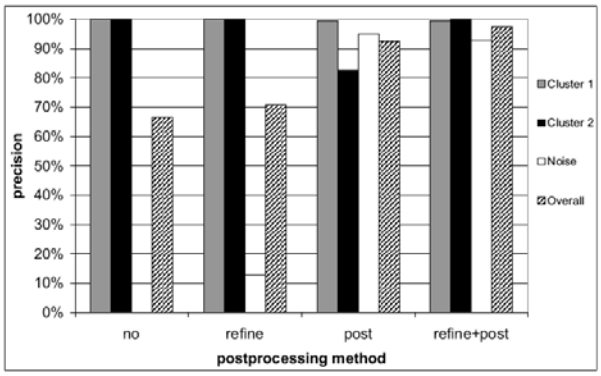


3.4.5 FIRES

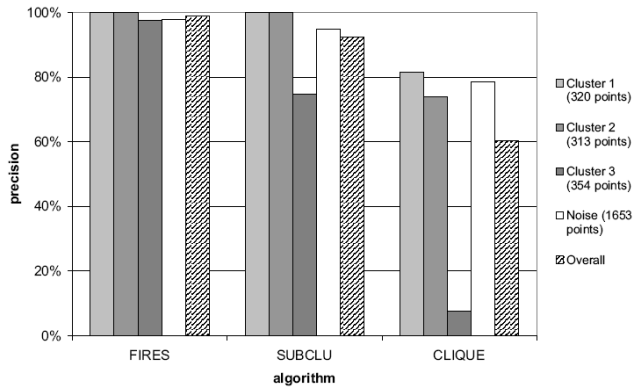
Evaluation Preprocessing

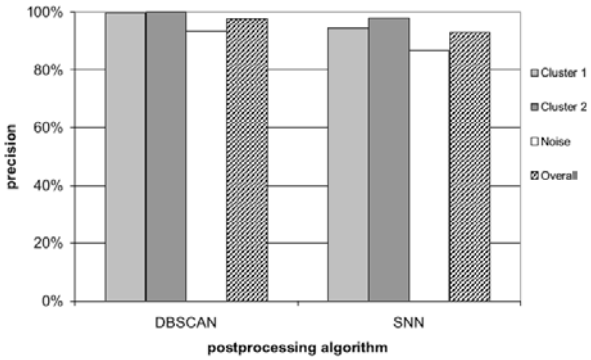


Evaluation Postprocessing

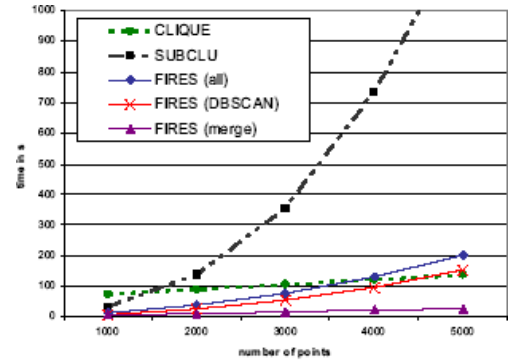


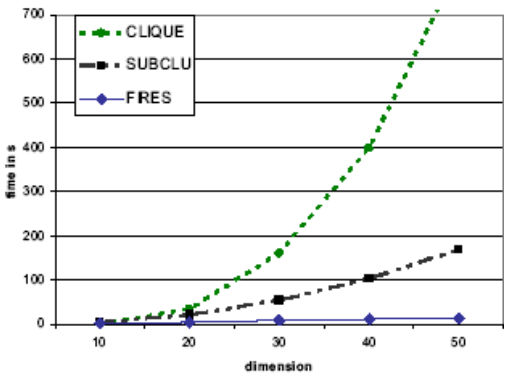
Evaluation Accuracy

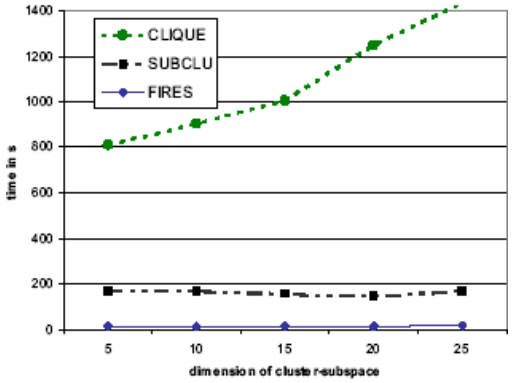


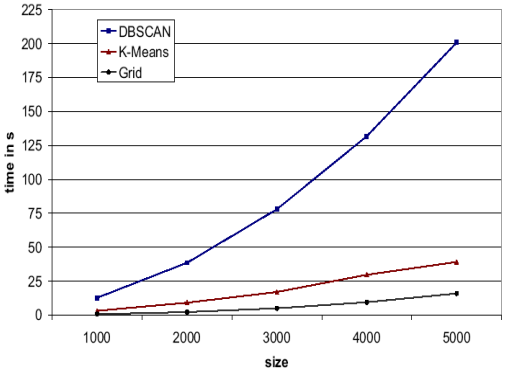


3.4.5 FIRES









3.4.5 FIRES

Genexpressionsdaten von Hefe

Analysiere 4,000 Gene bzgl. 24 Zeitpunkten

Beispiele gefundener Cluster:

ORF	Gene	Annotation
Cluster 1		
YGL019W	CKB1	subunit of CK2
YOR061W	CKA2	subunit of CK2
YOR236W	DFR1	chorismate pathway
YDR127W	ARO1	chorismate pathway
Cluster 2		
YGR172C	YIP1	ER to Golgi transport
YLR026C	SED5	ER to Golgi transport
YDR299W	BFR2	ER to Golgi transport
YNL287W	SEC21	ER to Golgi transport
Cluster 3		
YDR418W	RPL12B	part of ribosome
YIL133C	RPL16A	part of ribosome
YKL006W	RPL14A	part of ribosome

Genexpressionsdaten von Menschen

Analysiere 72 Patienten bzgl. 7070 Genen

Beispiele gefundener Cluster:

- 27 von 32 Patienten mit Leukämie Typ ALL (sowohl B-cell als auch T-cell Leukämie-Subtypen)
- 15 von 16 Patienten mit Leukämie Typ ALL und zugleich B-cell Leukämie-Subtyp
- 18 von 21 männliche Patienten

Zusammenfassung: Subspace Clustering

CLIQUE, ENCLUS, MAFIA
- Grid-basiertes Clustermodell
- Direkte Berechnung der Cluster

SUBCLU
- Dichte-verbundenes Clustermodell
- Direkte Berechnung der Cluster

RIS
- Dichte-verbundenes Clustermodell
- Ranking der Unterräume (flaches Clustering)

SURFING
- Dichte-verbundenes Clustermodell
- Ranking der Unterräume (hierarchisches Clustering)

FIRES
- Beliebiges Clustermodell
- Direkte Berechnung der Cluster

Globaler Dichteparameter
Subspacesuche:
Apriori-basiert

Lokal adaptiver
Dichteparameter
Subspacesuche:
Heuristik

Beliebiges Clustermodell
Subspacesuche:
Clustering der 1D-Cluster

Literatur

- C. C. Aggarwal and C. Procopiuc. *Fast Algorithms for Projected Clustering*. Proc. ACM Int. Conf. on Management of Data (SIGMOD), Philadelphia, US, 1999.(PROCLUS)
- Böhm C., Kailing K., Kriegel H.-P., Kröger P.: *Density Connected Clustering with Local Subspace Preferences*, Proc. 4th IEEE Int. Conf. on Data Mining (ICDM'04), Brighton, UK, 2004 (PreDeCon)
- Aggarwal, C., Yu, P.: *Finding Generalized Projected Clusters in High Dimensional Spaces*. Proc. ACM Int. Conf. on Management of Data (SIGMOD), Philadelphia, Dallas, US , 2000 (ORCLUS)
- Böhm C., Kailing K., Kröger P., Zimek A.: *Computing Clusters of Correlation Connected Objects*, Proc. ACM Int. Conf. on Management of Data (SIGMOD), Paris, France, 2004 (4C)
- Achtert E., Böhm C., Kröger P., Zimek A.: *Mining Hierarchies of Correlation Clusters*, Proc. 18th Int. Conf. on Scientific and Statistical Database Management (SSDBM'06), Vienna, Austria, 2006 (HICO)
- R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. *Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications*. Proc. ACM Int. Conf. on Management of Data (SIGMOD), Seattle, US, 1998.
- Kröger P., Kriegel H.-P.,Kailing K.: *Density-Connected Subspace Clustering for High-Dimensional Data*, Proc. SIAM Int. Conf. on Data Mining (SDM'04), Lake Buena Vista, FL, 2004, pp. 246-257.
- Kailing K., Kriegel H.-P., Kröger P., Wanka S.: *Ranking Interesting Subspaces for Clustering High Dimensional Data*, Proc. 7th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD'03), Cavtat-Dubrovnic, Croatia, 2003, (RIS)
- C. Baumgartner, K. Kailing, H.-P. Kriegel, P. Kröger, and C. Plant. *Subspace Selection for Clustering High-Dimensional Data*. Proc. 4th IEEE Int. Conf. on Data Mining (ICDM'04), Brighton, UK, 2004 (SURFING)
- Kriegel H.-P., Kröger P., Renz M., Wurst S.: *A Generic Framework for Efficient Subspace Clustering of High-Dimensional Data*, Proc. 5th IEEE Int. Conf. on Data Mining (ICDM'05), Houston, TX, 2005, pp. 250-257. (FIRES)