Wahl des initialen Clusterings

Idee

- Clustering einer kleinen Stichprobe liefert im allgemeinen gute initiale Cluster einzelne Stichproben sind evtl. deutlich anders verteilt als die Grundgesamtheit

Methode [Fayyad, Reina & Bradley 1998]

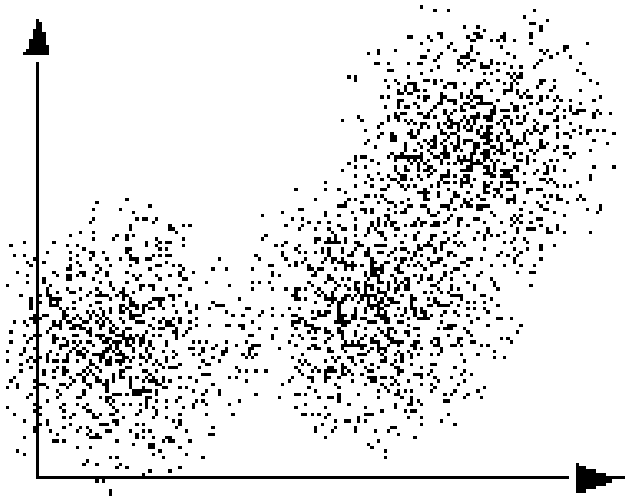
- ziehe unabhängig voneinander *m* verschiedene Stichproben
- clustere jede der Stichproben

  ⇨ *m* verschiedene Schätzungen für *k* Clusterzentren

  $A = (A_1, A_2, \ldots, A_k)$, $B = (B_1, \ldots, B_k)$, $C = (C_1, \ldots, C_k)$, . . .
- Clustere nun die Menge $DB = A \cup B \cup C..$

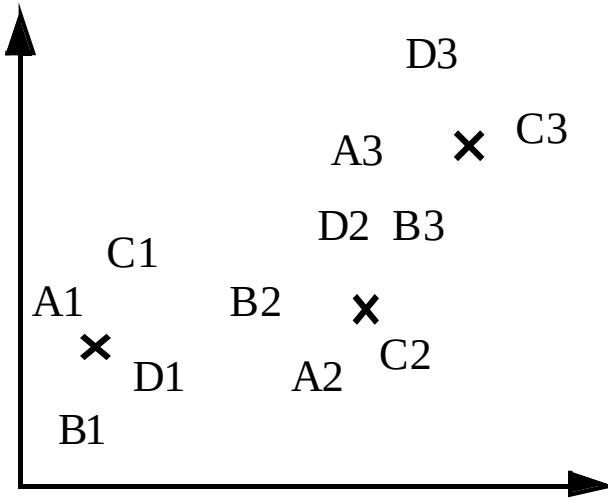
  mit *m* verschiedenen Stichproben *A*, *B*, *C*, . . . als Startkonfiguration
- Wähle von den *m* Clusterings dasjenige mit dem besten Wert bezüglich des zugehörigen Maßes für die Güte eines Clustering

Beispiel



Grundgesamtheit

$k = 3$ Gauß-Cluster



Clusterzentren

von $m = 4$ Stichproben

× *wahre Clusterzentren*

Wahl des Parameters k

Methode

- Bestimme für $k = 2, \ldots, n\text{-}1$ jeweils ein Clustering
- Wähle aus der Menge der Ergebnisse das „beste“ Clustering aus

Maß für die Güte eines Clusterings

- muss unabhängig von der Anzahl k sein
- bei k-means und k-medoid: TD^2 und TD sinken monoton mit steigendem k
- bei EM: E steigt monoton mit steigendem k
- Brauche ein von k unabhängiges Gütemaß für die k-means- und k-medoid-Verfahren

⇨ *Silhouetten-Koeffizient*

Silhouetten-Koeffizient [Kaufman & Rousseeuw 1990]

- sei *a*(*o*) der Abstand eines Objekts *o* zum Repräsentanten seines Clusters und *b*(*o*) der Abstand zum Repräsentanten des „zweitnächsten“ Clusters
- Silhouette *s*(*o*) von *o:*

$$s(o) = \frac{b(o) - a(o)}{\max\{a(o), b(o)\}}$$

$-1 \leq s(o) \leq +1$

$s(o) \approx -1 / 0 / +1$: *schlecht / indifferent / gute Zuordnung*

- Silhouettenkoeffizient s_C eines Clustering durchschnittliche Silhouette aller Objekte
- Interpretation des Silhouettenkoeffizients

$s_C > 0{,}7$: starke Struktur,

$s_C > 0{,}5$: brauchbare Struktur, . . .

k-modes Verfahren [Huang 1997]

- *k*-medoid-Algorithmus wesentlich langsamer als *k*-means- Algorithmus
- *k*-means-Verfahren nicht direkt für kategorische Attribute anwendbar

=> gesucht ist ein Analogon zum Centroid eines Clusters

- Numerische Attribute
  Centroid $\bar{x}$ einer Menge *C* von Objekten minimiert $TD(C,\bar{x}) = \sum_{p \in C} dist(p,\bar{x})$
- Kategorische Attribute
- Mode *m* einer Menge *C* von Objekten minimiert $TD(C,m) = \sum_{p \in C} dist(p,m)$
  (*m* ist nicht unbedingt ein Element der Menge *C*)
- $m = (m_1, ..., m_d)$, *dist* eine Distanzfunktion für kategorische Attribute, z.B.

$$dist(x,y) = \sum_{i=1}^{d} \delta(x_i, y_i) \text{ mit } \delta(x_i, y_i) = \begin{cases} 0, & falls\ x_i = y_i \\ 1, & sonst \end{cases}$$

Bestimmung des Modes

- Die Funktion $TD(C,m) = \sum_{p \in C} dist(p,m)$ wird genau dann

  minimiert, wenn für $m = (m_1, ..., m_d)$ und für alle Attribute A_i, $i = 1,..., d$, gilt:

  Es gibt in A_i keinen häufigeren Attributwert als m_i

- Der Mode einer Menge von Objekten ist nicht eindeutig bestimmt.

- Beispiel

  Objektmenge {(a, b), (a,c), (c, b), (b,c)}
  (a, b) ist ein Mode
  (a, c) ist ein Mode

Algorithmus k-modes

- Initialisierung
  - keine zufällige Partitionierung
  - sondern *k* Objekte aus der Datenmenge als initiale *Modes*
- Cluster-Repräsentanten
  - Mode anstelle des Centroids
- Distanzfunktion
  - anstelle der quadrierten euklidischen Distanz
  - Distanzfunktion für Datensätze mit kategorischen Attributen

Grundlagen

Idee

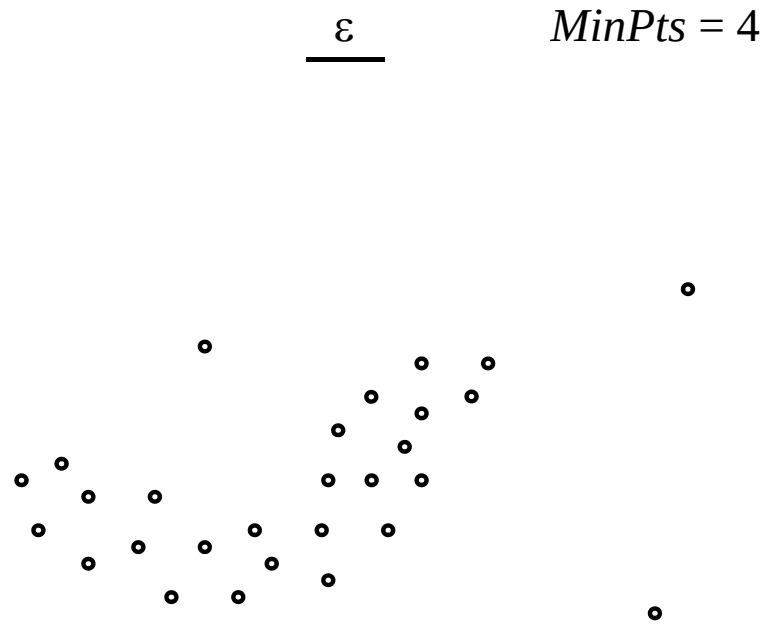
- Cluster als Gebiete im *d*-dimensionalen Raum, in denen die Objekte dicht beieinander liegen
- getrennt durch Gebiete, in denen die Objekte weniger dicht liegen

Zentrale Annahmen

- für jedes Objekt eines Clusters überschreitet die lokale Punktdichte einen gegebenen Grenzwert
- die Menge von Objekten, die den Cluster ausmacht, ist räumlich zusammenhängend

Intuition

Parameter $\varepsilon \in \Re$ und $MinPts \in \aleph$ spezifizieren Dichtegrenzwert

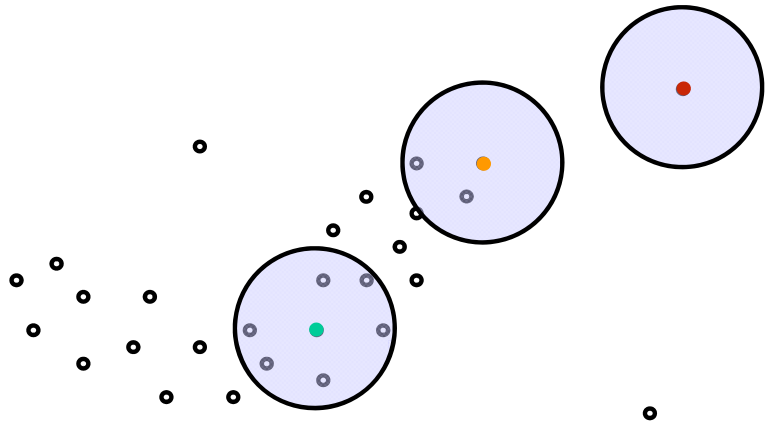


Intuition

Parameter $\varepsilon \in \Re$ und $MinPts \in \aleph$ spezifizieren Dichtegrenzwert

ε $MinPts = 4$

- *Kernpunkt*

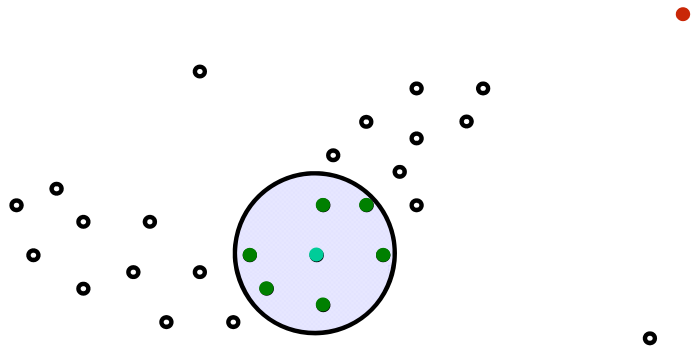


Intuition

Parameter $\varepsilon \in \Re$ und $MinPts \in \aleph$ spezifizieren Dichtegrenzwert

ε $MinPts = 4$

- *Kernpunkt*
- *Direkte Dichte-Erreichbarkeit*

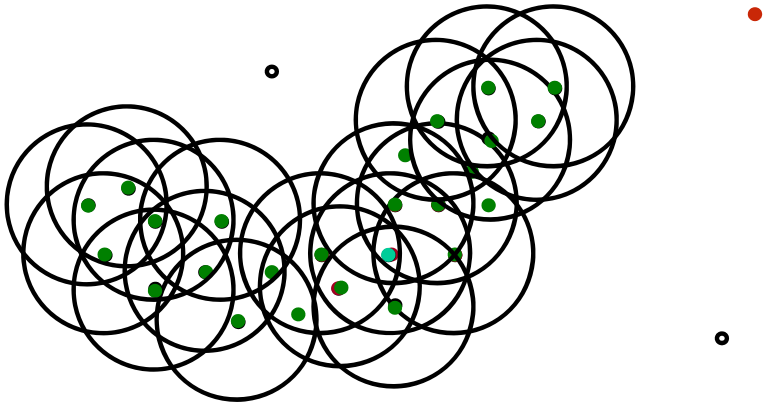


Intuition

Parameter $\varepsilon \in \Re$ und $MinPts \in \aleph$ spezifizieren Dichtegrenzwert

ε $MinPts = 4$

- *Kernpunkt*
- *Direkte Dichte-Erreichbarkeit*
- *Dichte-Erreichbarkeit*

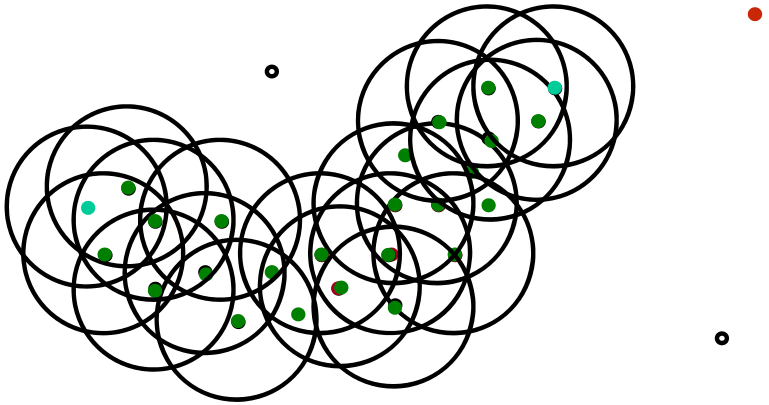


Intuition

Parameter $\varepsilon \in \Re$ und $MinPts \in \aleph$ spezifizieren Dichtegrenzwert

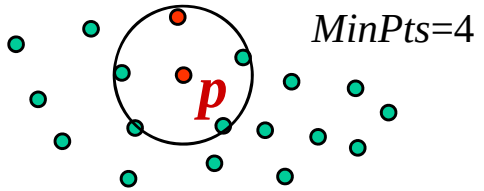
ε $MinPts = 4$

- *Kernpunkt*
- *Direkte Dichte-Erreichbarkeit*
- *Dichte-Erreichbarkeit*
- *Dichte-Verbundenheit*

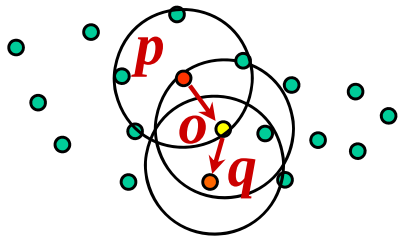


Formalisierung [Ester, Kriegel, Sander & Xu 1996]

- Ein Objekt $p \in DB$ heißt *Kernobjekt*, wenn gilt:
  $|RQ(o,\varepsilon)| \geq MinPts$
  ($RQ(o,\varepsilon)$ siehe Kap. 2.2, Folie 37)

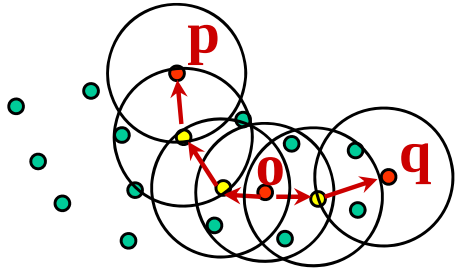


- Ein Objekt $p \in DB$ ist *direkt dichte-erreichbar* von $q \in DB$ bzgl. ε und *MinPts*, wenn gilt:
  $p \in RQ(q,\varepsilon)$ und q ist ein Kernobjekt in *DB*.



- Ein Objekt p ist *dichte-erreichbar* von q,
  wenn es eine Kette von direkt erreichbaren Objekten zwischen q und p gibt.

- Zwei Objekte p und q sind *dichte-verbunden*,
  wenn sie beide von einem dritten Objekt o aus dichte-erreichbar sind.



Formalisierung

Ein (*dichtebasierter*) *Cluster* C bzgl. ε und *MinPts* ist eine nicht-leere Teilmenge von DB für die die folgenden Bedingungen erfüllt sind:

Maximalität: $\forall p,q \in DB$: wenn $p \in C$ und q dichte-erreichbar von p ist, dann ist auch $q \in C$.

Verbundenheit: $\forall p,q \in C$: p ist dichte-verbunden mit q.

Formalisierung

Definition Clustering

Ein *dichtebasiertes Clustering CL* der Menge *DB* bzgl. ε und *MinPts* ist eine „vollständige“ Menge von dichtebasierten Clustern bzgl. ε und *MinPts* in *DB*.

Definition Noise

Die Menge $Noise_{CL}$ („*Rauschen*“) ist definiert als die Menge aller Objekte aus *DB*, die nicht zu einem der dichtebasierten Cluster $C \in CL$ gehören.

Grundlegende Eigenschaft

Sei *C* ein dichtebasierter Cluster und sei $p \in C$ ein Kernobjekt.
Dann gilt:

$$C = \{o \in DB \mid o \text{ dichte-erreichbar von } p \text{ bzgl. } \varepsilon \text{ und } MinPts\}.$$

=> ermöglicht effiziente Suche (WARUM?)

Algorithmus DBSCAN

```
DBSCAN(Punktmenge DB, Real ε, Integer MinPts)
  // Zu Beginn sind alle Objekte unklassifiziert,
  // o.ClId = UNKLASSIFIZIERT für alle o ∈ DB

   ClusterId := nextId(NOISE);
   for i from 1 to |DB| do
      Objekt := DB.get(i);
      if Objekt.ClId = UNKLASSIFIZIERT then
         if ExpandiereCluster(DB, Objekt, ClusterId, ε, MinPts)
         then ClusterId:=nextId(ClusterId);
```

5.3 Dichtebasiertes Clustering

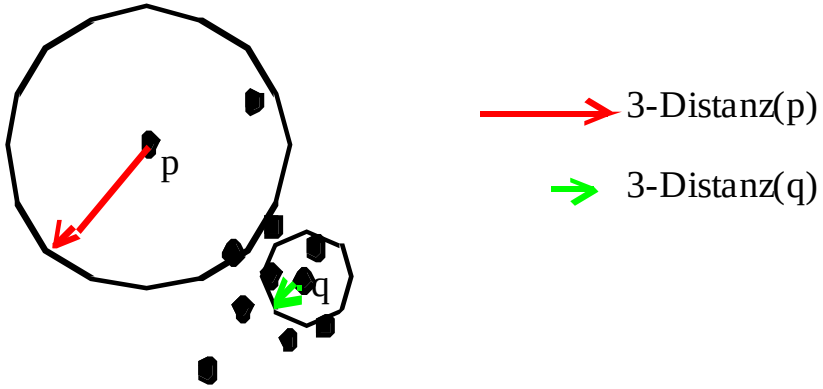

```
ExpandiereCluster(DB, StartObjekt, ClusterId, ε, MinPts): Boolean
seeds:= RQ(StartObjekt, ε);
if |seeds| < MinPts then // StartObjekt ist kein Kernobjekt
  StartObjekt.ClId := NOISE;
  return false;
// sonst: StartObjekt ist ein Kernobjekt
forall o ∈ seeds do o.ClId := ClusterId;
  entferne StartObjekt aus seeds;
  while seeds ≠ Empty do
    wähle ein Objekt o aus der Menge seeds;
    Nachbarschaft := RQ(o, ε);
    if |Nachbarschaft| ≥ MinPts then // o ist ein Kernobjekt
        for i from 1 to |Nachbarschaft| do
                p := Nachbarschaft.get(i);
                if p.ClId in {UNCLASSIFIED, NOISE} then
                        if p.ClId = UNCLASSIFIED then
                                füge p zur Menge seeds hinzu;
                        p.ClId := ClusterId;
    entferne o aus der Menge seeds;
return true;
```

Parameterbestimmung

Cluster: Dichte größer als die durch ε und *MinPts* spezifizierte „Grenzdichte“

Gesucht: der am wenigsten dichte Cluster in der Datenmenge

Heuristische Methode: betrachte die Distanzen zum *k*-nächsten Nachbarn.

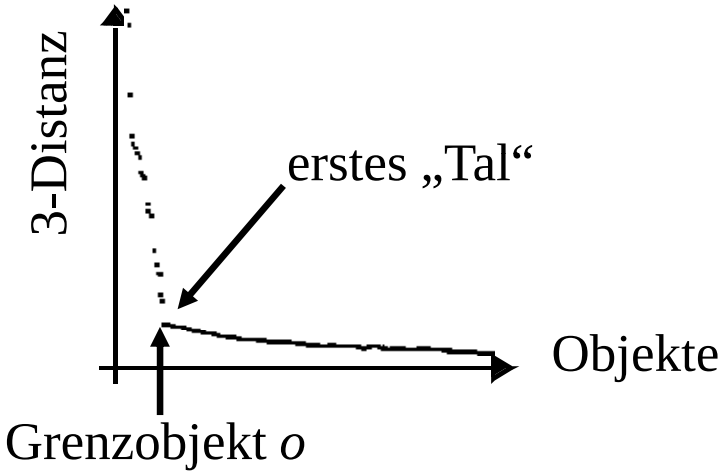


Funktion *k-Distanz*: Distanz eines Objekts zu seinem *k*-nächsten Nachbarn

k-Distanz-Diagramm: die *k*-Distanzen aller Objekte absteigend sortiert

Parameterbestimmung

Beispiel eines *k*-Distanz-Diagramms

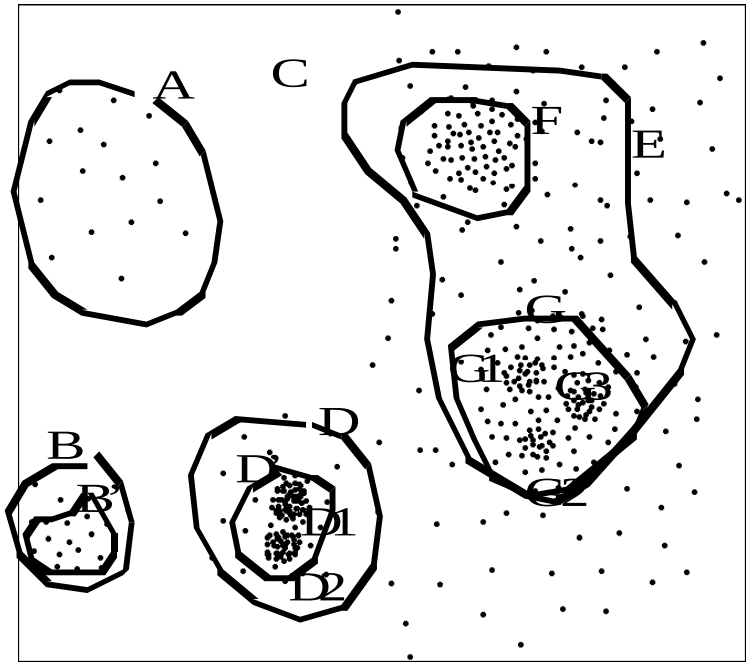


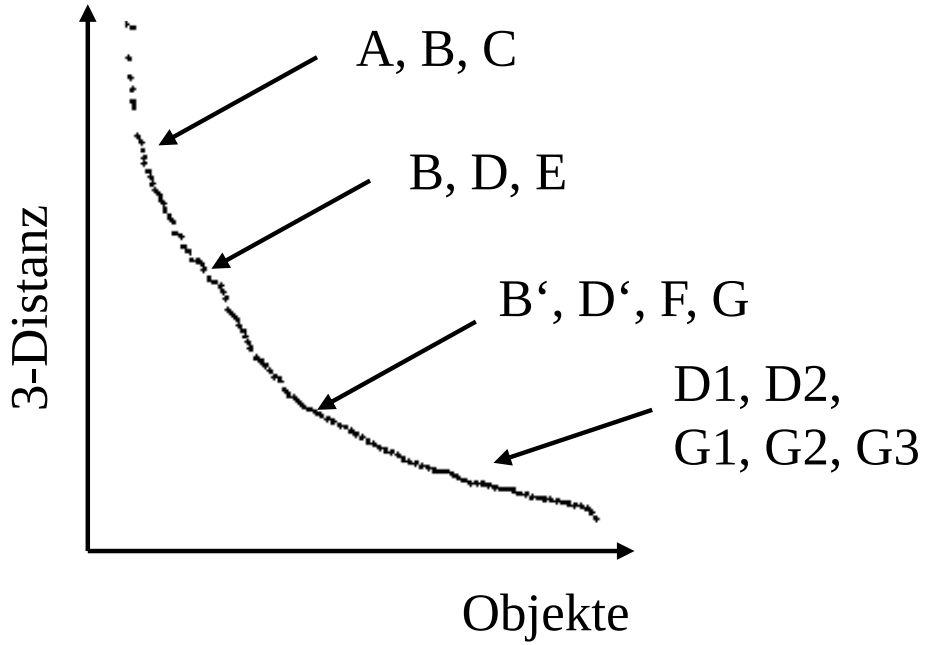
Heuristische Methode

- Benutzer gibt einen Wert für *k* vor (Default ist $k = 2*d - 1$), *MinPts* := *k*+1.
- System berechnet das *k*-Distanz-Diagramm und zeigt das Diagramm an.
- Der Benutzer wählt ein Objekt *o* im *k*-Distanz-Diagramm als Grenzobjekt aus, ε := *k*-Distanz(*o*).

Probleme der Parameterbestimmung

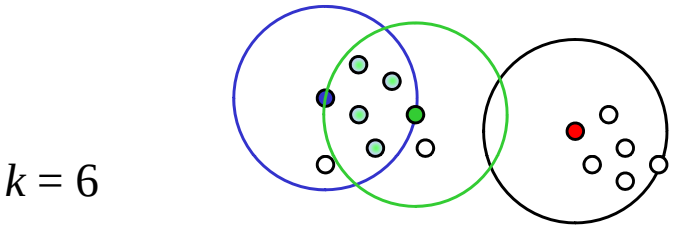
- hierarchische Cluster
- stark unterschiedliche Dichte in verschiedenen Bereichen des Raumes
- Cluster und Rauschen sind nicht gut getrennt





Shared Nearest Neighbor (SNN) Clustering

- DBSCAN
  - Erkennt Cluster unterschiedlicher Form und Größe
  - Hat Probleme bei Clustern mit unterschiedlicher Dichte
- Verbesserung: anderer Ähnlichkeitsbegriff
  - Ähnlichkeit zwischen zwei Objekten, wenn sie beide sehr nahe zu einer Referenzmenge *R* sind
  - Ähnlichkeit wird durch die Referenzmenge *R* "bestätigt"
  - Ähnlichkeit z.B. durch die Anzahl gemeinsamer nächster Nachbarn definieren (d.h. *R* ist die Menge der nächsten Nachbarn)
  - Shared Nearest Neighbor (SNN) Ähnlichkeit:
    - $SNN_k\text{-similarity}(p,q) = |NN(p, k) \cap NN(q, k)|$
    - $NN(o, k)$ = Menge der k-nächsten Nachbarn von o (vgl. Kap. 2.2)

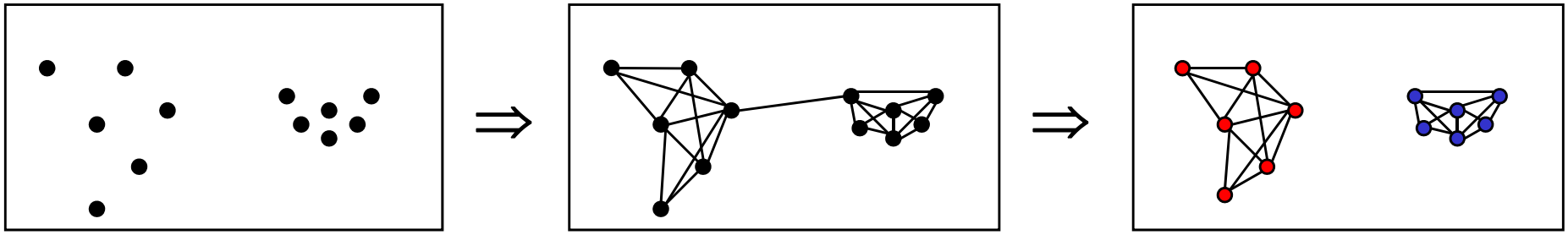


$SNN_6\text{-similarity}(\bullet, \bullet) = 4$

$SNN_6\text{-similarity}(\bullet, \bullet) = 0$

Einfaches SNN-Clustering [Jarvis, Patrick 73]:

1. Berechnung der Ähnlichkeitsmatrix und des Ähnlichkeitsgraphen
   - für alle Objekt-Paare $p,q \in DB$: berechne SNN_k-similarity(p,q)
   - SNN_k-Ähnlichkeitsgraph:
     - Knoten = Objekte
     - Kante zwischen jedem Objektpaar p,q mit Gewicht SNN_k-similarity(p,q)
   - Keine Kanten mit Gewicht 0
2. Generiere Cluster
   - Lösche alle Kanten, deren Gewicht unterhalb eines Grenzwerts τ liegen
   - Cluster = verbundene Komponenten im resultierenden Graphen

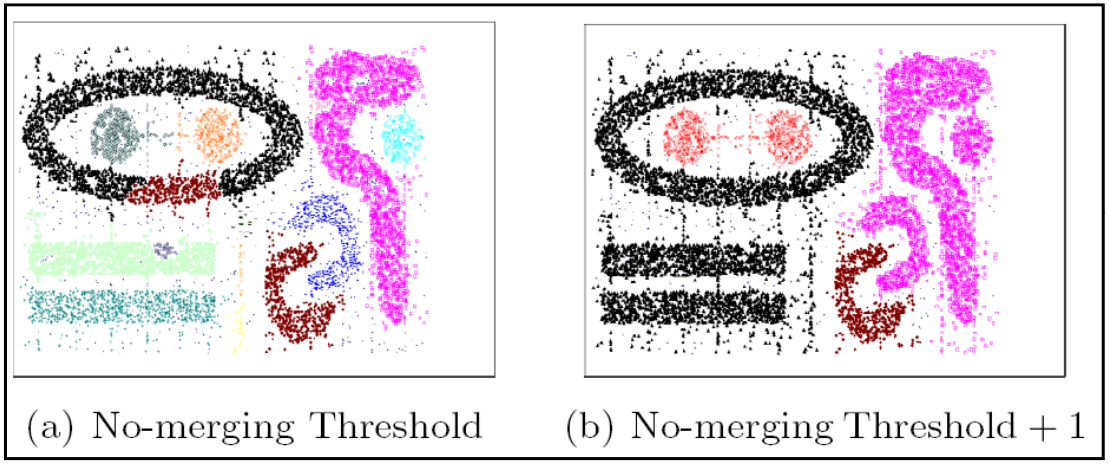


Datensatz | SNN_5 Ähnlichkeitsgraph | Generiere Cluster

5.3 Dichtebasiertes Clustering

Problem:

- Threshold τ schwer zu bestimmen
- kleine Variationen führen zu stark unterschiedlichen Ergebnissen



(a) No-merging Threshold (b) No-merging Threshold + 1

Bild aus:

[Ertöz, Steinbach, Kumar 03]

Lösung [Ertöz, Steinbach, Kumar 03]

- Kombiniere SNN-Ähnlichkeit mit dichtebasierten Konzepten
- SNN-Dichte:

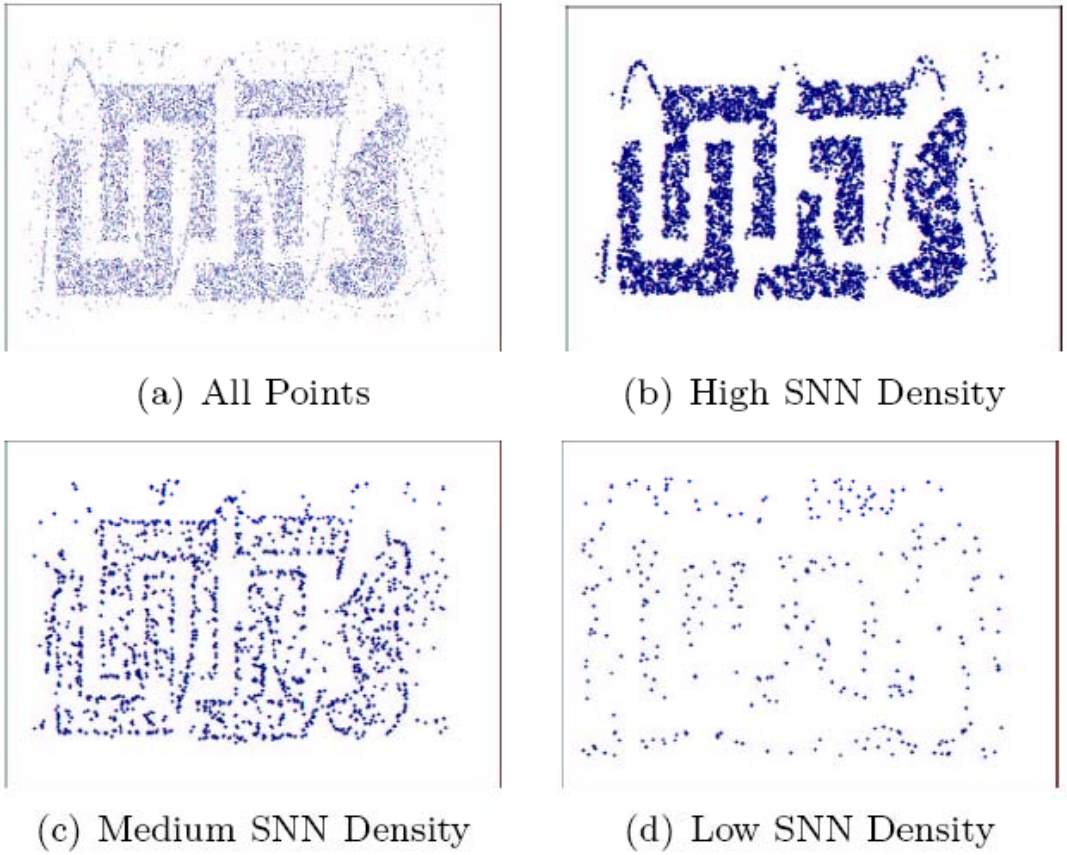
  Anzahl der Punkte innerhalb eines spezifizierten Radius ε bzgl. SNN-Ähnlichkeit

  $SNN_k\text{-density}(p,\varepsilon) = |\{q \mid SNN_k\text{-similarity}(p,q) \geq \varepsilon\}|$

5.3 Dichtebasiertes Clustering

SNN-Dichte

- Beispiel:
  - 10 000 Daten (Bild (a))
    $k = 50$, $\varepsilon = 20$
  - Bild (b): "Kernpunkte"
    alle Punkte mit SNN-Dichte ≥ 34
  - Bild (c): "Randpunkte"
    alle Punkte mit SNN-Dichte ≥ 17
  - Bild (d): "Rauschen"
    alle Punkte mit SNN-Dichte < 17



(a) All Points (b) High SNN Density

(c) Medium SNN Density (d) Low SNN Density

Bild aus: [Ertöz, Steinbach, Kumar 03]

- Analogie zu DBSCAN: $\varepsilon = 20$, *minPts* $= 34$
- Kernpunkt *p*: mehr als *minPts* Punkte haben 20 oder mehr der 50 nächsten Nachbarn mit *p* gemeinsam

SNN-Clustering Algorithmus [Ertöz, Steinbach, Kumar 03]

Eingabe: k, ε, *minPts*

1. Berechne Ähnlichkeitsmatrix und –graph
   (siehe einfaches SNN-Clustering)
2. Berechne die SNN_k-Dichte für jeden Punkt bzgl. ε
3. Bestimme Kernpunkte bzgl. *minPts*
   (alle Punkte mit einer SNN-Dichte $\geq$ *minPts*)
4. Vereinige Kernpunkte p,q, wenn SNN_k-similarity(p,q) $\geq \varepsilon$
5. Ordne Nicht-Kernpunkt p einem Cluster zu, wenn es einen Kernpunkt q gibt, mit SNN_k-similarity(q,p) $\geq \varepsilon$
6. Alle anderen Nicht-Kernpunkte sind Rauschen

DBSCAN mit SNN-Ähnlichkeit

Diskussion

- Unterschied zu DBSCAN
  - DBSCAN mit Euklidischer Distanz: nur Cluster, die dichter sind als der Grenzwert (spezifiziert durch *minPts* und ε)
  - SNN-Dichte eines Punktes *p*: Anzahl der Punkte, die mind. ε nächste Nachbarn mit *p* gemeinsam haben
  - $\Rightarrow$ unabhängig von der eigentlichen Dichte
- Parametrisierung
  - Wahl von *k* ist kritisch:
    - Zu klein: auch relativ gleichverteilte Cluster werden wegen lokalen Variationen gesplittet $\Rightarrow$ viele kleine Cluster
    - Zu groß: wenige große, gut-separierte Cluster
  - *minPts*, $\varepsilon < k$

Generalisierung von DBSCAN [Sander, Ester, Kriegel & Xu 1998]

Grundidee für dichte-basierte Cluster :

"*ε-Nachbarschaft* enthält mindestens *MinPts* Punkte"

"Distanz ≤ ε"

$NPred(o,p)$

reflexiv, symmetrisch

für Paare von Objekten

Verallgemeinerte Nachbarschaft

$N_{NPred}(o) = \{p \mid NPred(o, p)\}$

"$|N_\varepsilon| \geq MinPts$"

$MinWeight(N)$

beliebiges Prädikat für

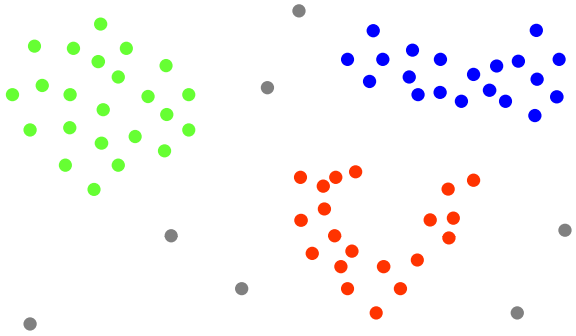
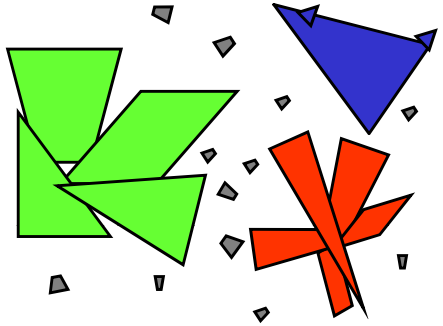
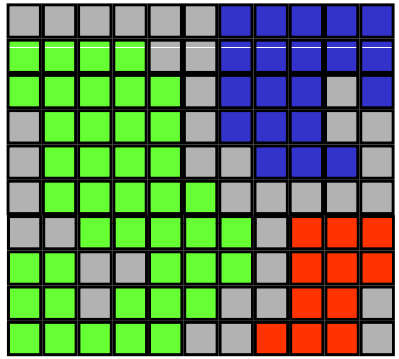
Mengen von Objekten

Verallgemeinerte minimale Kardinalität

$MinWeight(N_{NPred}(o))$

"*NPred-Nachbarschaft* hat mindestens das "Gewicht" *MinWeight*"

Beispiele

			
NPred	dist(p,q) ≤ ε	intersect(p,q)	Nachbarzelle und ähnliche Farbe
MinWeight	cardinality(. . .) ≥ MinPoints	Summe der Flächen ≥ 5 % der Gesamtfläche	true

Algorithmus GDBSCAN

- dasselbe algorithmische Schema wie DBSCAN
- anstelle einer RQ(o,ε)-Anfrage eine N_{NPred}-Anfrage
- anstelle der Bedingung $|RQ(o,\varepsilon)| \geq MinPts$
- das *MinWeight*-Prädikat auswerten
- Laufzeitkomplexität O(n log n) bei geeigneter Unterstützung der N_{NPred}-Anfrage
- Beliebige Nachbarschaftsprädikate denkbar