KDD 2 : TEIL 2
Data Mining in strukturierten Objekten

Skript zur Vorlesung

Knowledge Discovery in Databases II

im Sommersemester 2008

Data Mining in Strukturierten Objekten

Bis jetzt:

Datenobjekte werden durch Feature-Vektoren repräsentiert:

Aber:

- reelle Objekte können durch viele unterschiedliche Informationen charakterisiert werden.

  => Nicht alle Objekte lassen sich gut als Vektor von Grunddatentypen darstellen
- unterschiedliche Bedeutung der Features
- Strukturierung der Objekte durch Teilobjekte integriert zusätzlich Information.
- Integration von Domain-Knowledge durch Ausnutzen der gegebenen Modellierung

Arten von Strukturierten Objekten

1. Multirepräsentierte Objekte:
   Tupel aus Objekten unterschiedlicher Objekträume.
   Bsp.: Farbverteilung und Texturbeschreibung eines Pixelbilds

2. Multi-Instanz Objekte:
   Mengen aus Objekten. Alle Objekte sind Element des gleichen Objektraums
   Bsp.: Konfigurationen eines Moleküls, Warenangebot eines Händlers,..

3. Sequenzen
   Abfolge von Objekten i.d.R. des gleichen Objektraums
   Bsp.: Videos, Audio-Daten, Aminosäureketten, Zeitreihen...

4. Bäume
   Bäume aus Objekten, wobei Knoten und Kanten durch andere Objekte beschrieben sein können.
   Bsp.: Stammbäume, XML-Dateien...

5. Graphen
   gerichtete/ungerichtete Graphen aus Objekten, wobei Knoten und Kanten durch andere Objekte beschrieben sein können.
   Bsp.:Proteine, Bildsegmente, ...

Auswahlkriterium für Grad der Strukturierung

Jede Struktur kann als Spezialfall von Graphen aufgefasst werden.

Aber:

- häufig haben Teilobjekte keine bekannte Beziehung zueinander
- Beziehung zwischen Objekten ist oft vernachlässigbar
- Verwendung von graphstrukturierten Daten sehr aufwendig
  - Graph-Isomorphie ist nicht polynomiell berechenbar
  - Subgraph-Isomorphie ist NP-hart

Fazit: Die verwendete Strukturierung der Daten sollte so einfach wie möglich sein, aber die wesentlichen Merkmale erhalten.

Bsp: Zum Vergleich 2er Videos ist die zeitliche Abfolge von Szenen häufig nicht relevant. 2 Videos können schon als ähnlich betrachtet werden, wenn sie ähnliche Szenen in unterschiedlicher Reihenfolge enthalten.
(Sequenz von Szenen => Menge von Szenen)

Skript zur Vorlesung

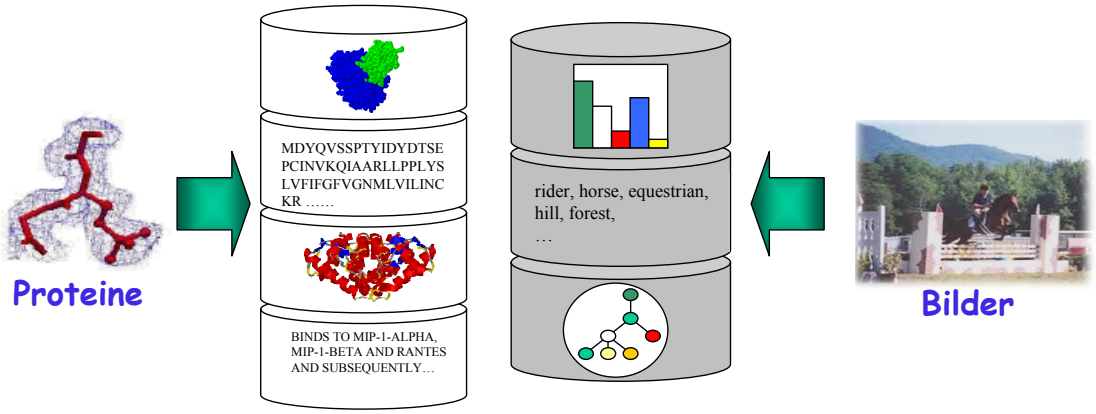
Knowledge Discovery in Databases II

im Sommersemester 2008

Kapitel 5: Multirepräsentiertes Data Mining und Ensemble-Methoden

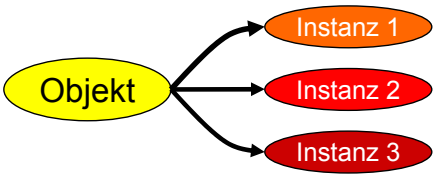
http://www.dbs.ifi.lmu.de/Lehre/KDD

Grundsituation



Gründe für Multirepräsentierte Objekte:

- unterschiedliche Featuretransformationen
- unterschiedliche Messtechniken
- unterschiedliche Aspekte desselben Objekts



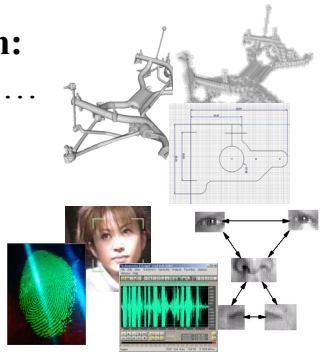
⇨ Multirepräsentierte Objekte

Was sind Multirepräsentierte Objekte ?

Weitere Anwendungen mit multirepräsentierten Daten:

• CAD-Bauteile: Voxel, Polygonzüge, Formhistogramme …

• Biometrische Daten: Sprachmuster, Gesichtszüge, Fingerabdrücke…



Formal:

- Objektrepräsentation $o = (r_1,..,r_n) \in R_1 \times .. \times R_n$, wobei R_i ein Darstellungsraum für die i-te Komponente mit $1 \leq i \leq n$.

- $R_i = O_i \cup \{-\}$, wobei O is a Featureraum und "-" ein Symbol für fehlende Instanzen.

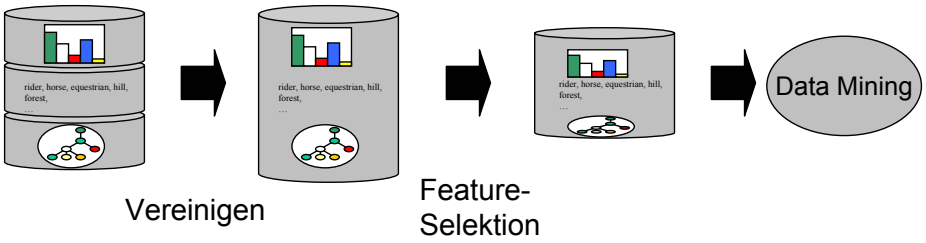
Probleme mehreren Repräsentationen

Grundproblem:

- alle notwendigen Informationen sollen dem Algorithmus zur Verfügung stehen => Verwende alle verfügbaren Informationen
- zu viele unnötige Features können das Ergebnis negativ beeinflussen => Verwende nur notwendige Features

Standard Lösungsansatz:

1. Bilde einen gemeinsamen Feature-Space aus allen Features jeder Repräsentation.
2. Benutze Feature-Reduktion oder Feature-Selektion.
3. Wende Data Mining auf reduzierten Feature-Raum an.



Probleme bei mehreren Repräsentationen

Probleme des Standard-Ansatzes :

- nicht alle Features sind vergleichbar.
  - Worthäufigkeit und Farbhäufigkeit bei Bildern
  - Aminosäurehäufigkeit und Zellmilieu bei Proteinen
- bei Nebeneinanderstellen unterschiedlicher Arten von Features, entsteht Informationsverlust bzgl. Vergleichbarkeit der Features

Lösungsansatz:

⇒ Data Mining Algorithmus erhält Tupel aus Feature-Vektoren oder anderen Objektdarstellungen.

⇒ Relevanz für Problem ist häufig auf Ebene der Repräsentationen zu entscheiden.
Beispiel: Spielen Farben ein Rolle bei Bildähnlichkeit?

⇒ hochdimensionale Daten können besser verarbeitet werden, indem Features nach Bedeutung gruppiert betrachtet werden.
=>Wissen über Zusammengehörigkeit der Features bleibt erhalten.

Multirepräsentierte Algorithmen

Möglichkeit zur Kombination mehrerer Repräsentationen:

1. **Kombination auf Feature-Ebene:**
   - unterschiedliche Merkmale werden aus verschiedenen Repräsentationen in einen Feature-Vektor vereint.
   - Feature-Selektion oder Selektion der Repräsentation sollen irrelevante Information ausschließen.

Bereits behandelt in Kap.2

2. **Kombination der Distanzen und Ähnlichkeiten**:
   Bestimme Objektähnlichkeit in jeder Repräsentation und kombiniere Ähnlichkeitsaussagen.

3. **Kombination auf Muster-Ebene:**
   Bestimme Muster in jeder Repräsentation und kombiniere die Muster zu allgemeinen Mustern.
   Bsp: Kombination der Klassenwahrscheinlichkeiten aus mehreren Repräsentationen.

Kapitelübersicht

5.1 Einleitung
Grundproblematik und Motivation

5.2 Multirepräsentierte Ähnlichkeits- und Distanzfunktionen
Lernen von Kombinationsregeln, Normalisierungen

5.3 Klassifikation mit Multirepräsentierten Objekten
Kombination von Klassifikatoren

5.4 Co-Training
Verwendung mehrerer Repräsentation zum Labeln neuer Trainingsobjekte

5.5 Multirepräsentiertes Clustering
dichtebasiertes Clustering, partitionierendes Clustering

5.2 Multirepräsentierte Ähnlichkeits- und Distanzfunktionen

Integration der verschiedenen Repräsentationen über Kombination von Ähnlichkeitsmaßen oder Distanzen.

Idee: Erhalte die Trennung der einzelnen Repräsentationen bei und kombiniere auf Ebene der Ähnlichkeitsaussagen.

Beispiel: gewichtete Linear-Kombination
$d_i(o_1,o_2)$: lokale Metrik oder lokaler Kernel in R_i.

$$D_{kombi}(o_1,o_2) = \sum_{R_i \in R} w_i \cdot d_i(o_1,o_2)$$

Normalisierung

Der Wertebereich von Distanzen in unterschiedlichen Repräsentationen kann sich stark unterscheiden.

$\Rightarrow$ Normalisierung der Ähnlichkeits- und Distanzwerte ist essentiell

Normalisierung mit Erwartungswert: Sei μ_i^{orig} der Erwartungswert aller Distanzen, die in Repräsentation R_i beobachtet wurden.

$$d_i(o,q) = \frac{d_i^{orig}(o,q)}{\mu_i^{orig}}$$

Bereichs-Normalisierung:

$$d_i(o,q) = \frac{d_i^{orig}(o,q) - \min_{r,s \in D}\left\{d_i^{orig}(r,s)\right\}}{\max_{r,s \in D}\left\{d_i^{orig}(r,s)\right\} - \min_{r,s \in D}\left\{d_i^{orig}(r,s)\right\}}$$

Gewichtung der Repräsentationen

- Normalisierung erzeugt Vergleichbarkeit bzgl. der Wertebereiche.
- Semantische Aussage der Repräsentationen ist nicht so einfach bestimmbar. Z.B.: Ist ein sehr ähnliches Farbhistogramm weniger aussagekräftig als eine sehr ähnliche Textbeschreibung?

$\Rightarrow$ Gewichtung der Repräsentationen ist essentiell für Nutzen

Annahme gewichtete Linearkombination: $D_{kombi}(o_1,o_2) = \sum_{R_i \in R} w_i \cdot d_i(o_1,o_2)$

- Auswahl durch Domain-Experten
- Integration in das Optimierungsproblem des Klassifikators (Hyper-Kernels und SVMs)
- explizites Lernen von Gewichten

Lernen von Gewichten

Formuliere Ähnlichkeit als lineares Klassifikationsproblem:

Normalenvektor der trennenden Hyperebene setzt sich aus Gewichten zusammen:

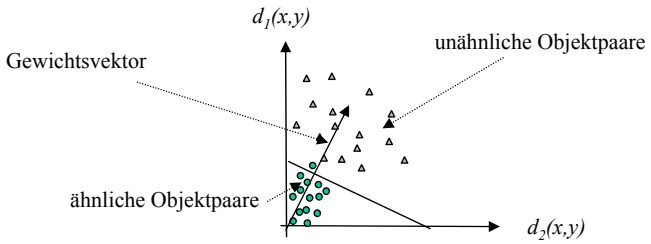
Trainingsobjekte: Paare von ähnlichen und unähnlichen Objekten

Klassen: {„ähnlich“,“unähnlich“}

Feature-Space: Abstandsvektor, $v_i = d_i(x,y)$ für alle Represenation R_i $1 \leq i \leq n$

Vorgehen:

- Bestimme Abstandsvektoren auf DB-Sample
  (Vorsicht: Es gibt quadratisch viele Abstandsvektoren! => Sample)
- Trainiere linearen Klassifikator
- Bestimme Gewichtungsvektor aus Normalenvektor der Trennebene (MMH).



Kombination von Distanzen/Ähnlichkeiten

Bemerkungen:

- Vorsicht: lineare Klassifikatoren garantieren keine positiven Gewichte für alle Repräsentationen!

- Alternativ kann auch der gelernte Klassifikator direkt zur Kombination der Ähnlichkeiten bzw. Distanzen verwendet werden. In diesem Fall ersetzt die Wahrscheinlichkeit für die Klasse „unähnlich“ die Distanz.

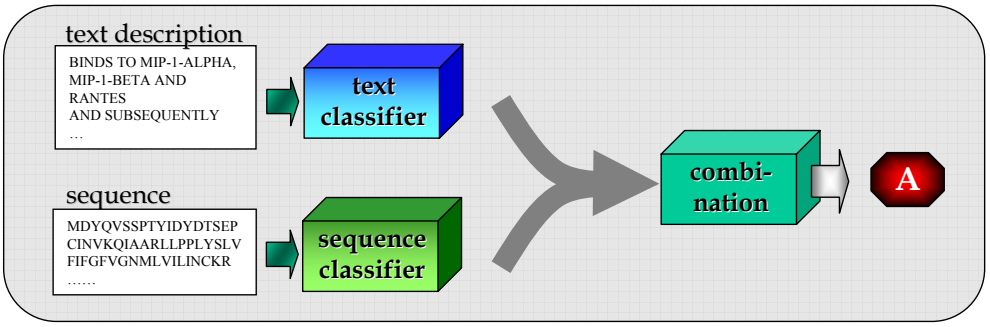
- Bei komplexeren Kombinationsregeln müssen die Metrik- bzw. Kernel-Eigenschaften erneut geprüft werden, falls das anschließende Data-Mining-Verfahren diese Eigenschaften benötigt.

5.3 Multirepräsentierte Klassifikation

Eingabe : $o \in R_1 \times .. \times R_n$,
wobei R_i der Darstellungsraum für die i-te Repräsentation ist.

Kombination mehrerer Klassifikatoren (Classifier Combination):

1. Trainiere Klassifikator für jede Repräsentation getrennt.
2. Klassifiziere neues Objekte mit jedem Klassifikator
3. Kombiniere die Resultate der Klassifikatoren zur einer globalen Klassenvorhersage.



Kombination der Klassifikatoren

Beispiel: Man betrachte 2 Doktoren *A* und *B*, die entscheiden sollen, ob ein Patient krank ist oder gesund.
Untersuche den Nutzen der zweiten Meinung in den Fällen in denen nur 1 Dr. richtig liegt, beide Dr. richtig liegen, keiner richtig liegt.

Fall A: kein zusätzlicher Nutzen durch zweiten Experten.

Fall D: kein Nutzen durch irgendeinen Experten.

Fall B und C:
Nutzen durch 2 Experten, wenn sich der richtig liegende durchsetzt.

Fazit: *Nutzen kann nur bei widersprüchlichen Diagnosen auftreten, sofern sich der Richtige durchsetzt.*

Dr. B \ Dr. A	richtig	falsch
richtig	A	B
falsch	C	D

Kombination der Klassifikatoren

Der Nutzen der Multirepräsentierten Klassifikation hängt von 2 Faktoren ab:

1. Unterscheidet sich die Vorhersage der einzelnen Klassifikatoren hinreichend genug, um genug potentiellen Nutzen zuzulassen.
   => Wie kann man entscheiden, ob Repräsentationen widersprüchliche Aussagen erzeugen ?

2. In Falle widersprüchlicher Vorhersagen, sollte die Aussage gewählt werden die richtig ist.
   => Wie kann man die Konfidenz der Aussagen bestimmen ?

Potenzieller Nutzen der Kombination

Unabhängig von der richtigen Aussage kann Nutzen nur dann entstehen, wenn widersprüchlich Klassenvorhersagen auftreten.

- Klassenvorhersage sollte nicht positiv korreliert sein
  Grund: Kein Nutzen, da überwiegend gleiche Aussagen.

- Klassenvorhersage sollte auch nicht negativ korreliert sein.
  Grund:
  - Bei vollständiger negativer Korrelation, kann immer nur ein Klassifikator richtig liegen.
  - Die durchschnittliche Vorhersagequalität der Klassifikatoren muss schlecht sein.
    => Experten raten nur !!

Fazit: *Die Klassenvorhersagen und damit die Repräsentationen sollten möglichst unabhängig voneinander sein!*

Kombination mehrerer Klassifikatoren

Wie kombiniert man Klassenvorhersagen so, dass die richtige Vorhersage bevorzugt wird?

1. Jeder Klassifikator gibt für jede Klasse A und ein Objekt x eine Vorhersagewahrscheinlichkeit c_A zurück.
   Für Konfidenzvektor $c^r(x)$ gilt: $\sum_{A \in C} c_A^r(x) = 1$

2. Klassifikation durch Kombintion der Konfidenzevektoren $c^r(x)$:

$$pred(X) = \underset{A \in C}{\mathbf{argmax}} \left(\Theta_{r \in R} \left(c_A^r \right) \right) \text{ mit } \Theta \in \left\{ \min, \max, \sum, \prod \right\}$$

Kombination mehrerer Klassifikatoren

Beispiel:

Gegeben: 2 Repräsentation für Bildobjekte: Farbhistogramme(R1) und Texturvektoren(R2).
Klassen = {„enthält Wasseroberfläche“=A, „keine Wasseroberfläche“=B}
Bayes Klassifikatoren K1 (für R1) und K2 (für R2)
Kombination mit Summe.

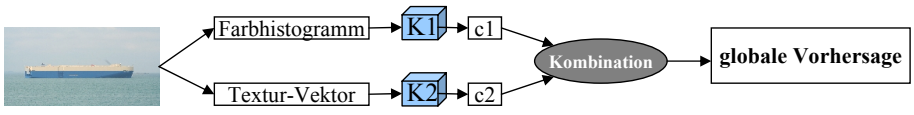
Klassifikation von Bild b:
K1(b)=c1 =(0.45, 0.55); K2(b) = c2 = (0.6, 0.4)

Kombination mit Durchschnitt (Summe):
c_{global} = (1.05, 0.95) * ½ = (0.525 , 0.475) und argmax(c_{global}) = A

Kombination mit Produkt:
c_{global} = (0.27, 0.22) und argmax(c_{global}) = A

Kombination mit Maximum:
c_{global} = (0.6, 0.55) und argmax(c_{global}) = A

Kombination mit Mininum:
c_{global} = (0.45, 0.4) und argmax(c_{global}) = A



Kombination mehrerer Klassifikatoren

Probleme:

- Performanz der kombinierten Klassenvorhersage ist stark von der Aussagekraft des Konfidenzvektor abhängig.

  Ist die Konfidenz der Aussage nicht stark mit der Richtigkeit korreliert, wir häufig die falsche Aussage bevorzugt.

- Bestimmte Klassifikatoren erzeugen keinen Konfidenzvektor sondern nur eine Klassenvorhersage.

  => Verfahren zum Abschätzen der Klassenkonfidenz für allgemeine Klassifikatoren

⇒ Ableitung von zuverlässigen Konfidenzwerten für unterschiedliche Arten von Klassifikatoren.

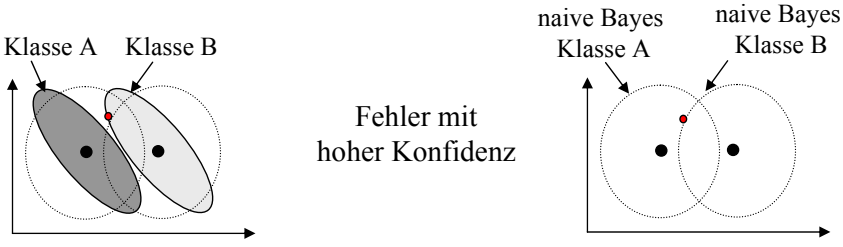
Ableiten von Konfidenzvektoren

Bayes Klassifikatoren

Bayes-Klassifikatoren berechnen ohnehin eine Wahrscheinlichkeit pro Klasse.

Vorsicht: Konfidenzwerte häufig unzuverlässig, weil sie ausdrücken inwieweit ein Objekt zum gelernten Prozess passt.

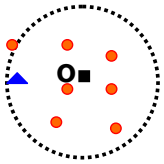
Aber: Wenn Daten schlecht durch zugrunde liegenden statistischen Prozess beschrieben werden, entstehen leicht falsche Klassifikationsergebnisse mit hoher Konfidenz.

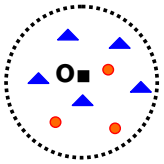


Multirepräsentierte *k*NN-Klassifikation

Idee: Um die Konfidenz für einen kNN-Klassifikator zu bestimmen betrachte die Eindeutigkeit (≡Entropie) in der Entscheidungsmenge (kNN-Sphäre).

Entscheidungs-menge in R1





Entscheidungs-menge in R2

Intuition: In R_1 scheint der Bereich um das Objekt eindeutig zu Klasse ● zu gehören. In R_2 ist der Bereich um das Objekt durch beide Klassen ▲ und ● gegeben.

=> Vorhersage in R_1 scheint zuverlässiger zu sein als die in R_2

Multirepräsentierte *k*NN-Klassifikation

[Kriegel, Pryakhin, Schubert 2005]

Klassifikator K:O → C bildet $o \in O$ auf eine Klasse c ∈ C
O besteht aus den Repräsentationen $R_1 \times \ldots \times R_n$

Klassifikation des Objekts $o = (r_1,\ldots,r_n)$:
Bestimmme Entscheidungsmenge $sphere_i(o,k)$ in jeder Repräsentation R_i,
wobei $r_i \neq$ "-" (= die Repräsentation ist vorhanden)

$$sphere_i(o,k) = \{o_1,\ldots,o_k \mid o_1,\ldots,o_k \in DB_i \wedge \neg\exists o' \in DB_i \setminus \{o_1,\ldots,o_k\}$$
$$\wedge \neg\exists \lambda, 1 \le \lambda \le k : dist_i(o',r_i) \le dist_i(o_\lambda,r_i)\}$$

Bestimme Konfidenzvektor $cv_i(o)$ auf folgende Art und Weise:

$$I \quad d_i^{norm}(o,u) = \frac{dist_i(o,u)}{\max_{v \in sphere_i(o,k)} dist_i(o,v)}$$

$$II \quad \hat{c}(o)_{i,j} = \sum_{u \in sphere_i(o,k) \wedge c(u)=c_j} \frac{1}{d_i^{norm}(o,u)^2}$$

$$III \quad \forall j, 1 \le j \le |C| : cv_{i,j}(o) = \frac{\hat{c}(o)_{i,j}}{\sum_{k=1}^{|C|} \hat{c}(o)_{i,k}}$$

$$IV \quad cv_i(o) = (cv_{i,1}(o),\ldots,cv_{i,|C|}(o))$$

Multirepräsentierte *k*NN-Klassifikation

Kombination der cv_i in jeder Repräsentation mit Gewichten:

$$Cl_{mr}(o) = \max_{j=1,\ldots,|C|} \sum_{i=1}^{m} w_i \cdot cv_{i,j}(o)$$

Gewicht $w_{o,i}$ der Objekts o in Repräsentation i :

$$w_{o,i} = \begin{cases} 0 & , falls\ r_i = \text{„-“} \\ \dfrac{1 + \sum_{j=1}^{|C|} (cv_{i,j}(o) * \log_{|C|} cv_{i,j}(o)}{\sum_{k=1}^{m} (1 + \sum_{j=1}^{|C|} (cv_{k,j}(o) * \log_{|C|} cv_{k,j}(o)))} & , sonst \end{cases}$$

Idee: Je „reiner“ die Nachbarschaft eines Objekts *o ist*,
desto zuverlässiger ist die Aussage des kNN Klassifikators in Rep. R_i.

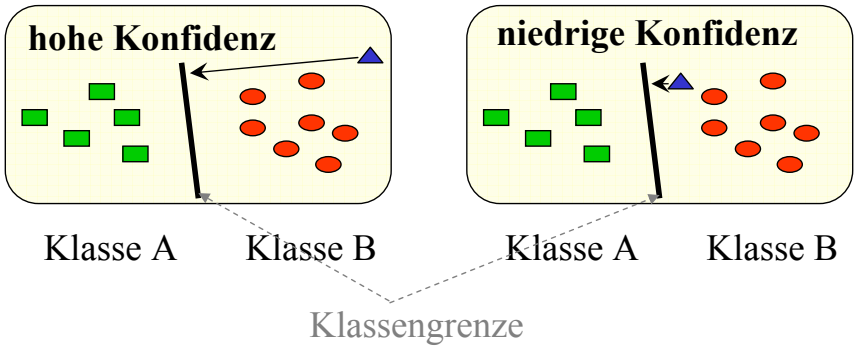
Konfidenzabschätzung mit Confidence-Ranges

Wie kann man unabhängig vom Typ des Klassifikators Aussagen über die Zuverlässigkeit einer Klassenvorhersage treffen?

Beobachtung: Die Unsicherheit nimmt an den Klassengrenzen zu. Objekte die weit von der Klassengrenze liegen, sind weit von anderen Klassen entfernt und damit verhältnismäßig sicher klassifiziert.

Idee: Bestimme inwieweit ein Objekt o verschoben werden kann, ohne dass sich die Klassenvorhersage ändert.

=> Je größer die Verschiebung desto stabiler die Klassenvorhersage.



Konfidenzabschätzung mit Confidence-Ranges

Klassifikator Kombination mit Confidence Ranges:

- Confidence Range *CRange(o)* ist der Betrag der minimalen Verschiebung, die die Klasse von o verändern würde.

$$CRange(o) = \min\left\{ \|v\| \;\middle|\; v \in F \wedge CL(o) \neq Cl(o+v) \right\}$$

mit F als zugrunde liegenden Featureraum und $CL(o)$ als Klassifikator.

- **Interpretation:** Wie groß muss *CRange(o)* sein um mit x% Konfidenz sagen zu können, dass o richtig klassifiziert wurde?

$\Rightarrow$ Lerne ein Funktion die *CRange(o)* auf Konfidenz $CE_i(o)$ mappt.

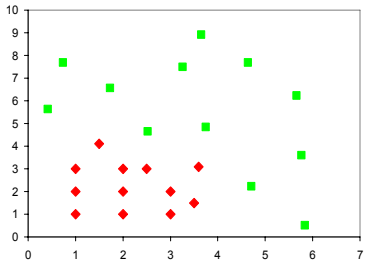
$CE_i(o)$ wird als Confidence Estimate von o in R_i bezeichnet.

- Kombinierte Klassifikation: Wähle die Klasse, die in der Repräsentation R_i mit dem höchsten Wert für $CE_i(o)$ vorhergesagt wurde.

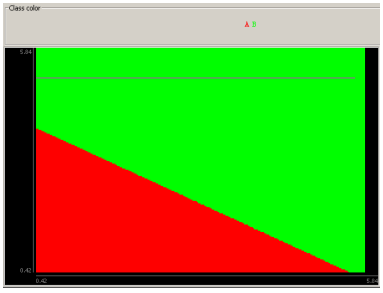
$$CL_{global}(o) = CL_{\arg\max_{0 \le j \le n}\{CE_j(o)\}}(o)$$

Bestimmung der Confidence-Range

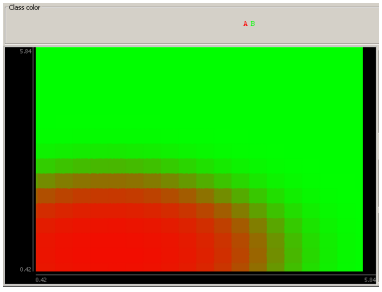
originale Datemenge



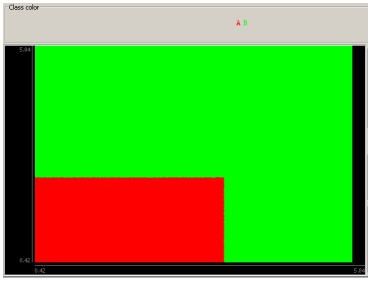
lineare SVM



naive Bayes

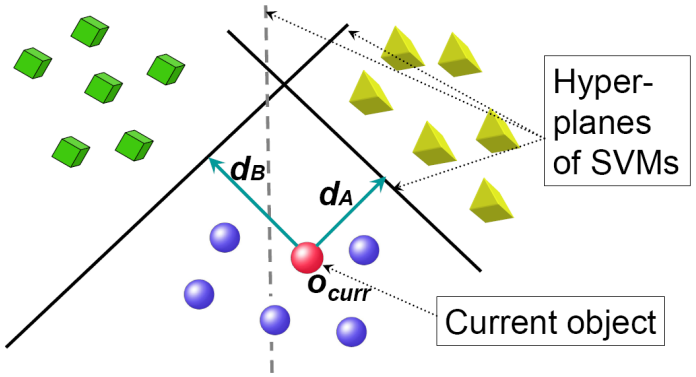


Entscheidunsbaum



Bestimmung der Confidence-Range für SVM

Support Vector Machine



Minimal **Class Set *C* = { , , }**

d_A, d_B: distances between o_{curr} and borders of its class

Bestimmung der Confidence-Range für Bayes-Kl.

Bayes Classification

$$p(c_{pred}) \cdot p(\boldsymbol{x}|c_{pred}) = p(c_{pred}) \cdot p(\boldsymbol{x}|c_{other})$$
$$\Rightarrow p(c_{pred}) \cdot p(\boldsymbol{x}|c_{pred}) - p(c_{pred}) \cdot p(\boldsymbol{x}|c_{other}) = 0$$

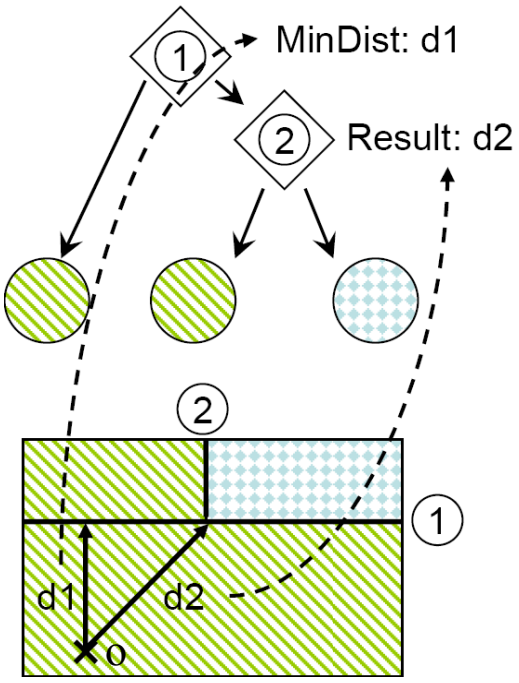
$$\min_{x \in \mathbb{R}^d} d(o, x)$$
$$s.t.\ \ p(c_{pred}) \cdot p(\boldsymbol{x}|c_{pred}) - p(c_{pred}) \cdot p(\boldsymbol{x}|c_{other}) = 0$$

nicht lineares Optimierungsproblem,
Lösung z.B. mit Gradient Descent Verfahren

$$\min_{x \in \mathbb{R}^d} d(o, x)$$
$$s.t. (x - \mu_1)^T \times (\Sigma_1)^{-1} \times (x - \mu_1)$$
$$-(x - \mu_2)^T \times (\Sigma_2)^{-1} \times (x - \mu_2) - \ln \frac{p(c_1) \cdot \Sigma_2}{p(c_2) \cdot \Sigma_1} = 0$$

Entscheidungsbaum

- normaler Durchlauf des Entscheidungsbaums um Objekte o zu klassifizieren
- Während des Durchlaufs werden die nicht verfolgten Unterbäume und Blätter in einer Priority-Queue(PQ) gespeichert
- die Einträge in PQ sind nach dem minimalen Abstand zu o sortiert.
- Nach finden des Blattknotens in dem o enthalten ist, wir PQ solange abgearbeitet bis Blatt mit anderen Klassenzuordnung auftaucht



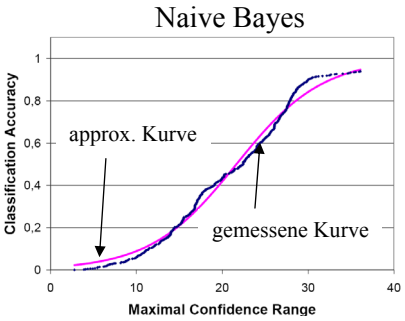
Von Confidence Ranges zu Confidence Estimates

Ziel: Ableiten eines Konfidenzwertes aus CRange(o).

Beobachtung: Betrachte Vorhersagenauigkeit für alle Objekte mit CRange(o) < x.

Die beobachtete Kurve hat die Form einer Sigmoid-Funktion.

$$sig(x) = \frac{1}{1+e^{-\alpha x+\beta}}$$



⇒ Lerne Sigmoid-Funktion , die die Genauigkeit für einen bestimmten Wert von CRange(o) vorhersagt.

$$CE_i(o) = \frac{1}{1+e^{-\alpha_i CRange_i(o)+\beta_i}}$$

Confidence Estimate $CE_i(o)$

Bestimmen von $CE_i(o)$

für jede Repräsentation i:

- bestimme für jedes Objekt o, CRange(o) und ob die Klassenvorhersage richtig ist mit 3-fold Cross-Validation.

- Bilde eine Kurve aus folgende Werten: x = CRange(o) (max. CRange)
  y = |R(x)|/|DB(x)| (Accuracy)

  $DB(x) = \{ o \in DB \wedge CRange(o) < x \}$

  $R(x) = \{ o \in DB(x) \wedge CL(o) = K(o) \}$

- Approximiere Graphen mit Regression => Parameter für CE_i
  (Verwendung der Methode von Levenberg und Marquadt)

Klassifikation mit Confidence Ranges

Geg.: CL_i und CE_i sind bereits trainiert für jede Repräsentation i.

Klassifikation nach folgenden Schema:

1. Bestimme CL_i(p) und $CRange_i(o)$ für alle R_i.
2. Bestimme $CE_i(o)$ für alle $CRange_i(o)$.
3. Bestimme R_p bei dem $CE_p(o)$ *am höchsten.*
4. Klassifiziere o durch $CL_p(o)$

5.3 Co-Training

Multiple Repräsentationen können auch dazu verwendet werden eine Trainingsmenge zu erweitern.

Gegeben: 2 Repräsentationen für die sowohl gelabelte als auch nicht gelabelte Objekte vorhanden sind.

Idee:

Benutze Klassifikator um neue Trainingsobjekte aus ungelabelten Datenobjekten zu erzeugen.

Aber: Wiso braucht man dazu mehrere Repräsentationen ?

Generieren von Trainingsobjekte mit nur 1 Repräsentation

Versuch:

- Trainiere Klassifikator ***CL*** auf allen gelabelten Objekten
- klassifiziere k ungelabete Objekte und füge sie in die Trainingsmenge ein.
- Trainiere nächsten Klassifikator auf der neuen Trainingsmenge

Problem:

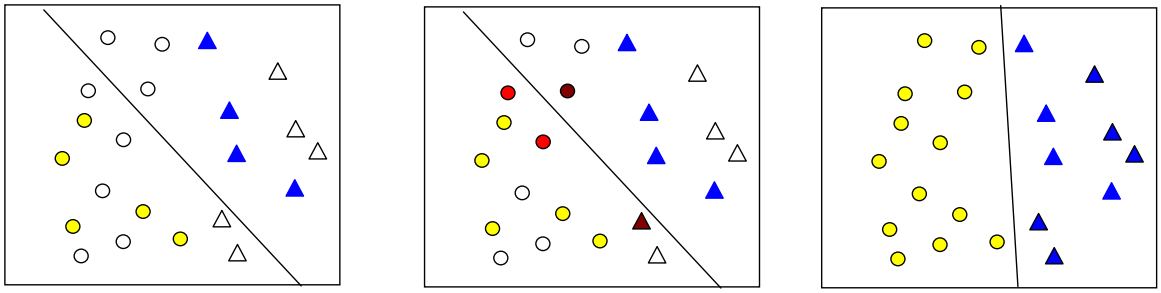
- neue Daten werden mit dem Modell von ***CL*** gelabelt
- damit neue Trainingsobjekte CL verändern können, müssten sie aber Widersprüche zum bisherigen Modell enthalten

=> Generieren von Trainingsobjekten mit einer Repräsentation verstärkt nur die Schwächen des Klassifikators

Generieren von Trainingsobjekte mit nur 1 Repräsentation

Beispiel:

- blau = gelabelte Objekte Dreieck-Klasse
- gelb = gelabelte Objekte Kreis Klasse
- rot = relabelte Objekte mit CL_1



Training auf originalen Daten | Training mit relabelten Daten | optimale Lösung

Fazit:

- Die roten Objekte bestätigen nur die Annahmen des Klassifikators, können diese aber nicht verbessern.
- Zur Verbesserung wären von CL unabhängig Informationen notwendig.

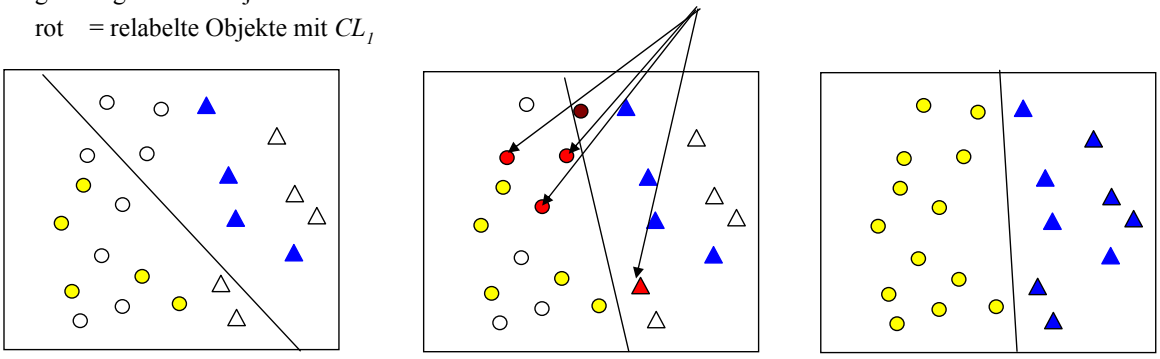
Co-Training

Idee: Klassifikatoren aus anderen Repräsentationen labeln Objekte, mit für diese Repräsentation neuen Informationen.

Beispiel:

- blau = gelabelte Objekte Dreieck-Klasse
- gelb = gelabelte Objekte Kreis Klasse
- rot = relabelte Objekte mit CL_1

Objekte die durch Cl_2 in R_2 gelabelt wurden



originaler Klassifikator | Klassifikator nach unabhängigen Relabeling | optimale Lösung

=> Durch neue unabhängig gelabelte Objekte kann sich ein Klassifikator verbessern.

Der Co-Training Algorithmus

Gegeben: 2 Mengen aus multirepräsentierten Objekten

TR = gelabelte Trainingsmenge, *U* = Menge ungelabelter Objekte.

Co-Training Algorithmus

For *k* times do

For each R_i Do

Trainiere CL_i für Repräsentation *i*.

Ziehe Sample aus *U*.

generiere neue Label mit CL_i.

füge neu gelabelte Objekte zu *TR* hinzu

Bemerkungen zum Co-Training

Ansatz ist von 2 Aspekten abhängig:

1. Jeder beteiligte Basisklassifikator muss für sich selbst ausreichend genau sein:

   Bei schlechter Vorhersagequalität einzelner Klassifikatoren sind neue Label nicht zuverlässig genug.

2. Basisklassifikatoren müssen hinreichend unabhängig voneinander sein.
   Sind sich die Klassifikatoren einig entsteht kein Nutzenpotential .

5.3. Clustering Multirepräsentierter Objekte

Anforderungen an Clustering-Algorithmen für Multirepräsentierte Objekte:

- Integration aller Informationsquellen.
- Eigenschaften in unterschiedlichen Repräsentationen müssen unterschiedlich behandelt werden.
- spezialisierte Techniken für unterschiedliche Arten von Repräsentationen sollten verwendet werden.
  (Zugriffsmethoden, Indexstrukturen, Distanzmaße …).
- Der Aufwand sollte möglichst nur linear mit jeder Repräsentation ansteigen.

Clustering Multirepräsentierter Objekte

Naive Lösungen zum Clustering Multirepräsentierter Objekte

Clustere nur die vielversprechendste Repräsentation:
+ bereits bekannte Clusteringalgorithmen sind verwendbar
- Verwendet nicht alle Repräsentationen.
- Welche Repräsentation ist die beste.

Verwende zusammengesetzten Featureraum:
+ verwendet alle Repräsentationen.
- Die Anazahl der Feature in jeder Repräsentation ergibt eine ungewollte Gewichtung .(z.B. Text-Vektoren und Farbhistogramme)
- Keine Verwendung spezialisierter Distanzmaße
  (z.B. Edit-Distanz, Cosinus-Distanz, Jaccard…)
- Unterstützung für Ähnlichkeitsanfragen schwierig.
  Keine spezialisierten Index und Speicherstrukturen.

Dichtebasiertes Multirepräsentiertes Clustering

Anforderung: Kombiniere Repräsentationen

- so früh wie möglich (während des Clusterings)
  - ➢ nur ein Durchlauf durch alle Repräsentationen
- so spät wie möglich (nach den ε-Bereichs-Anfragen)
  - ➢ ermöglicht Verwendung spezieller Indexstrukturen und Distanzmaße

➡ *Idee*: Definiere die Kernobjekt Eigenschaft neu

Vereinigungs-Methode

Idee: Ein Objekt ist in einem dichten Bereich, wenn k Nachbarn in allen Repräsentationen in der ε-Umgebung liegen.

Geeignet für : "sparse" Daten mit viel Rauschen.

Vereinigungs-Kernobjekt:
Sei $\varepsilon_1, \varepsilon_2, .., \varepsilon_m \in \Re^+$, $MinPts \in N$, $o \in O$ ist ein **Vereinigungs-Kernobjekt**, falls

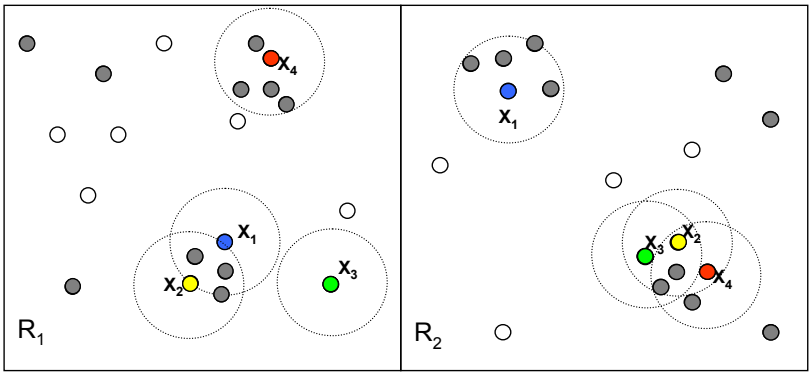
$$\left| \bigcup_{R_i(o) \in o} N_{\varepsilon_i}^{R_i}(o) \right| \geq MinPts$$, wobei $N_{\varepsilon_i}^{R_i}(o)$ die lokale ε-Nachbarschaft in Repr. i ist.

Direkte Vereinigungserreichbarkeit:
Objekt $p \in O$ ist **direkt vereinigungserreichbar** von $q \in O$ bzgl. $\varepsilon_1, \varepsilon_2, .., \varepsilon_m$ und $MinPts$, falls q ein Vereinigungs-Kernobjekt in O ist und es gilt:

$$\exists \; i \in \{1, \ldots, m\} : R_i(p) \in N_{\varepsilon_i}^{R_i}(q)$$

Vereiniguns-Methode

Clusterexpansion bei der Vereinigungsmethode



MinPts = 3

Schnitt-Methode

Idee: Ein Objekt ist in einem dichten Bereich, falls es *k* Objekte in den ε-Nachbarschaften aller Repräsentationen gibt.

Geeignet für: dichte Repräsentationen and unzuverlässige lokale Feature-Vektoren.

Schnitt-Kernobjekt:
Sei $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m \in \Re^+$, $MinPts \in N$. o ∈ O ist ein **Schnitt-Kernobjekt**, falls

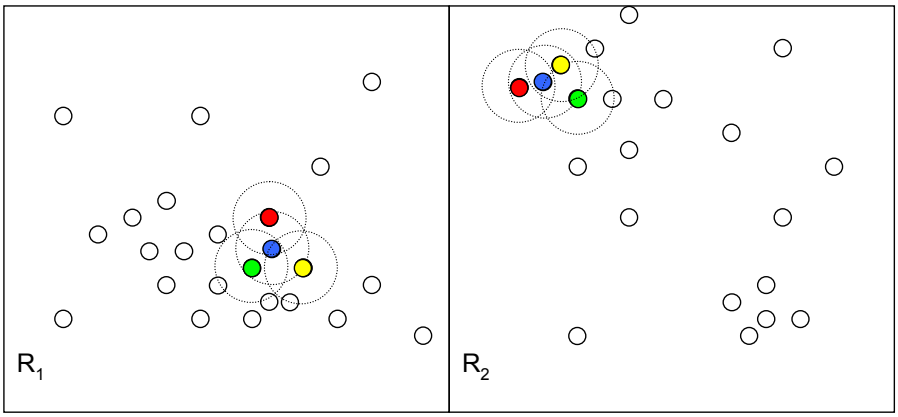
$\left| \bigcap_{R_i(o) \in o} N_{\varepsilon_i}^{R_i}(o) \right| \geq MinPts$, wobei $N_{\varepsilon_i}^{R_i}(o)$ die lokale ε-Nachbarschaft in Repr. i ist.

Direkt schnitterreichbar:
Objekt $p \in O$ ist **direkt schnitterreichbar** von $q \in O$ bzgl.
$\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ und *MinPts,* falls q ein Schnitt-Kernobjekt in O ist und es gilt:

$$\forall \quad i \in \{1, \ldots, m\} : R_i(p) \in N_{\varepsilon_i}^{R_i}(q)$$

Schnitt-Methode

Clusterexpansion mit Schnitt-Methode



MinPts = 3

Qualitätsmaß für MR-Clustering

Definition: Sei O eine Datenmenge und $C = \{C_i | C_i \subset O\}$ ein Clustering. Weiterhin sei $K = \{K_i | K_i \subset O\}$ ein Referenz-Clustering (= tatsächliches Clustering).

$$quality_K(C) = \sum_{C_i \in C} \frac{|C_i|}{|O|} \cdot \left(1 + entropy_K(C_i)\right)$$

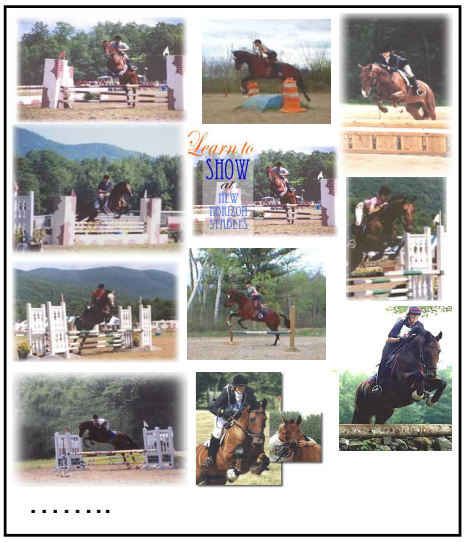
- Vergleicht Clustering mit Referenz-Clustering
- Bevorzugt Clusterings mit wenig Rauschen.
- Jeder Cluster sollte möglichst nur zu einem Cluster im Referenz-Clustering gehören. => Niedrige Entropie

Beispiel-Ergebnisse auf Bilddaten

Cluster in den einzelnen Repr.

Beispiele für Bilder im Cluster IC 5
(nur Farbhistogramme)

Beispiele für Bilder im Cluster IC 5
(nur Segmentbäume)



Cluster IC5 der auf beiden Repräsentationen
mit der Intersectionmethode gebildet wurde.

Multirepräsentiertes OPTICS

Distanzen in allen Repräsentationen müssen vergleichbar sein
=> Normalisierung der Distanzen

2 mögliche Vorteile durch mehrere Repräsentationen

- Verbesserung der Clusterqualität (Precision)
  => 2 Objekte sind dann ähnlich,
  wenn sie in allen Repräsentationen ähnlich sind
- Vergrößere die Clusterabdeckung (Recall)
  => es reicht wenn ein Objekt in einer Repräsentation ähnlich ist

Integration dieser Idee in OPTICS durch Redefinition der
Kerndistanz und der Erreichbarkeitsdistanz

Intersection Clustering (Schnittmethode)

Idee: Ein Objekt ist in einem dichten Raum, wenn es *k* Objekte gibt, die bzgl. aller Repräsentationen ähnlich sind.

Intersection *k*-nächste-Nachbar-Distanz: $k \in \mathbb{N}$

$$NN - Dist_k^{\cap} = \max\{\max_{i=1..m}\{d_i(o,q)\} \mid q \in NN_k^{\cap}(o)\}$$

Intersection Kerndistanz: $k \in \mathbb{N}$, $\varepsilon \in \mathbb{R}$

$$CORE_{\varepsilon,k}^{\cap} = \begin{cases} NN - Dist_k^{\cap}(o) & if \quad |N_{\varepsilon}^{\cap}(o)| \geq k \\ \infty & else \end{cases}$$

Intersection Erreichbarkeitsdistanz:

$$REACH_{\varepsilon,k}^{\cap}(p,o) = \max\{CORE_{\varepsilon,k}^{\cap}(p), \max_{i=1..m}\{d_i(o,p)\}\}$$

Union Clustering (Vereinigungsmethode)

Idee: Ein Objekt ist in einer dichten Region, wenn es *k* Objekte gibt, die in einer beliebigen Repräsentation ähnlich sind.

Union-k-Nächste-Nachbar Distanz: $k \in \mathbb{N}$

$$NN - Dist_k^{\cup} = \max\{\min_{i=1..m}\{d_i(o,q)\} \mid q \in NN_k^{\cup}(o)\}$$

Union-Kerndistanz: $k \in \mathbb{N}$, $\varepsilon \in \mathbb{R}$

$$CORE_{\varepsilon,k}^{\cup} = \begin{cases} NN - Dist_k^{\cup}(o) & if \quad |N_{\varepsilon}^{\cup}(o)| \geq k \\ \infty & else \end{cases}$$

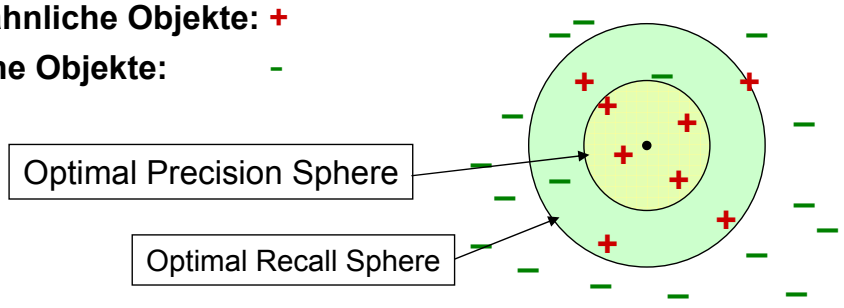
Union-Erreichbarkeitsdistanz:

$$REACH_{\varepsilon,k}^{\cup}(p,o) = \max\{CORE_{\varepsilon,k}^{\cup}(p), \min_{i=1..m}\{d_i(o,p)\}\}$$

Bedeutung der Repräsentationen

wirklich ähnliche Objekte: +

unähnliche Objekte: -



Möglichen Interpretationen der ε-Nachbarschaft:

hohe Precision- und Recall-Werte

=>1 Rep. lässt gutes Clustering zu

niedrige Precision- und Recall-Werte

=> alle Rep. lassen kein gutes Clustering zu

hohe Precision- aber niedrige Recall-Werte

$\Rightarrow$ benutze Vereinigungs-Methode

niedrige Precision- aber hohe Recall-Werte

$\Rightarrow$ verwende Schnitt-Methode

Kombinationsbäume

Klassifiziere Repräsentationen nach:

- Precision-Räume: alle Objekt weisen gute Precision-Sphere auf.
- Recall-Räume: Alle Objekte weisen gute Recall-Sphere auf.

Kombination der Precision-Spaces mit der Vereinigungsmethode

=> Ergebnis: besserer Recall-Space.

Kombination der Recall-Spaces mit der Schnitt-Methode

=> Ergebnis: besserer Precision-Space.

Für große Mengen an Repräsentationen kombiniere beide Methoden mit Kombinationsbäumen.

Kombinationsbäume

Gegeben Repräsentationen $R = \{ R_1,..,R_m \}$ ein Kombinationsbaum KB ist :

Baum mit beliebigen Grad g >0

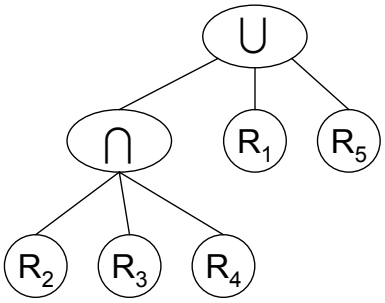
Blätter entsprechen den Repräsentationen

Innere Knoten entsprechen den Operatoren $\theta \in \{\cup,\cap\}$

Semantik:

$\cup$ verbessert Recall

$\cap$ verbessert Precision



Verwendung von Kombinationsbäumen

Kombinationsbäume implizieren folgendes rekursives Distanzmaß:

$$d^n(o,p) = \begin{cases} \min\limits_{c \in n.children} \{d^c(o,p)\} & if\ n.label = \cup \\ \max\limits_{c \in n.children} \{d^c(o,p)\} & if\ n.label = \cap \\ d_i(o,p) & if\ n.label = R_i \end{cases}$$

für den vollständigen Baum: $d_{CT}(o,p) = d^{CT.root}(o,p)$

Bemerkung: Für die sinnvolle Interpretation der Distanzen ist eine vorherige Normalisierung notwendig.

Multirepräsentiertes Partitionierendes Clustering

[Bickel,Scheffer 2004]

Auch partitionierende Clustering-Verfahren wie, k-Means, k-medoid oder EM, können auf multiple Repräsentationen angepasst werden.

Grundannahme: Cluster entstehen durch einen statistischen Prozeß je Repräsentation.

⇒ jedes Objekt gehört zu genau einem Cluster in jeder Repräsentation

⇒ multirepräsentierte Cluster bestehen 1 Cluster in jeder Repräsentation der exakt die gleichen Objekte enthält.

Idee: Korrektes Clustering in einer Repräsentation impliziert ein korrektes multirepräsentiertes Clustering.

Aber: Partitionieredes Clustering Algorithmen terminieren in lokalen Minima

=> benutze mehrere Repräsentation, um nicht in lokalen Minima zu terminieren

Multirepräsentiertes Partitionierendes Clustering

Zur Anwendung von partitionierenden Clustering Algorithmen wird eine Zielfunktion und ein Modellbildungsschritt benötigt:

- Zielfunktion: $$MRTD^2 = \sum_{R_i \in R} \sum_{C_{ik} \in C_i} \sum_{x \in C_{ik}} d(x, c_{ik})^2$$

  $R=\{R1,..Rn\}$ Repräsentationen,

  C_{ik}: k-ter Cluster in R_i, c_{ik}: Centroid des k-ten Cluster in R_i.

- Modelbildung: Gruppierung der Objekte bzgl. R_j

  Berechnung neuer Centroide c_{ik} in R_i bzgl. Aufteilung aus R_j

- Consensus-Clustering bei Nicht-Terminieren:

  MR-partitionierendes Clustering garantiert kein globales Maximum

  => Algorithmus kann MRTD^2 nicht mehr verbessern und es gibt kein einheitliches Clustering

  => Bilde globales Cluster-Modell aus allen Punkten, die in allen Repräsentationen richtig eingeordnet wurden.

$$c_k = \frac{\sum_{x \in \bigcap_{R_j \in R} C_{jk}} x}{\left| \bigcap_{R_j \in R} C_{jk} \right|}$$

Multirepräsentiertes k-Means

Gegeben: Suche k Cluster in DB aus multirep. Objekten aus n Repräsentation R_i.

Algorithmus: *MR-k-Means*
aktR:=1
$MRTD^2_{old} = \infty$
Initialisiere Clustering in R_{aktR}
Berechne Objektaufteilung pro Cluster
Bestimme $MRTD^2_{neu}$
Wiederhole bis $(MRTD^2_{old} - MRTD^2_{neu}) \leq \varepsilon$
    oldR:=aktR
    aktR:=(aktR+1)MOD n
    Bestimme Centroide in R_{aktR} mit der Aufteilung aus R_{oldR};
    $MRTD^2_{old}$:=$MRTD^2_{neu;}$
    uosdate($MRTD^2_{neu}$)
Ende der Wiederholung
Berechne Consenses Clustering

Multirepräsentiertes Partitionierendes Clustering

Bemerkungen:

- Ansatz ist auch auf EM und k-Medoid Verfahren anwendbar.
- Algorithmus terminiert nicht zwangsläufig
- Verbesserung der lokalen Clusterings durch Verwendung der anderen Repräsentationen.
- Ansatz nimmt keine Wertung der Repräsentationen vor.
  => alle Repräsentationen beeinflussen das Ergebnis gleich stark.

Literatur

- Aßfalg J., Kriegel H.-P., Pryakhin A., Schubert M.: ***Multi-Represented Classification based on Confidence Estimation***
  in proc. 11th Pacific-Asia Conference on Knowledge Discovery and Data Mining **(PAKDD 2007)**, Nanjing, China
- Achtert E., Kriegel H.-P, Pryakhin A., Schubert M.:
  Clustering Multi-Represented Objects Using Combination Trees
  in proc. 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining **(PAKDD 2006)**, Singapore
- Kriegel H.-P., Pryakhin A., Schubert M,:
  Multi-represented kNN-Classification for Large Class Sets
  10th International Conference on Database Systems for Advanced Applications **(DASFAA 2005)**, Beijing, China.
- Kailing K., Kriegel H.-P.,Pryakhin A., Schubert M.:
  Clustering Multi-Represented Objects with Noise
  Proc. 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining **(PAKDD'04)**, Sydney, Australia, 2004.
- Bickel S. ,Scheffer T.: ***Multi-View Clustering***, 4h IEEE International Conference on Data Mining **(ICDM 2004)**.
- Blum. A, Mitchell T.: ***Combining Labeled and Unlabeled Data with Co-Training***, Workshop on Computational Learning Theory (COLT 98)