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

Skript zur Vorlesung

Knowledge Discovery in Databases II

im Sommersemester 2007

Kapitel 2:Feature-Selektion und Feature-Reduktion

(Folien 21,22,60-63 aus KDD-Skript 2005)

http://www.dbs.ifi.lmu.de/Lehre/KDD2

2. Featurereduktion und Featureselektion

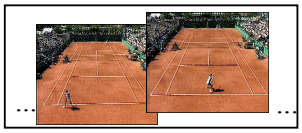
Inhalt dieses Kapitels

2.1 Einführung

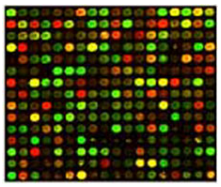
„Reale Daten sind meist sehr hochdimensional“

Beispiele:

- Fernsehbildern
  - Fernsehbilder werden in Farbhistogramme zerlegt
  - Je nach Farbauflösung: 100 – 1.000 dimensionale Feature-Vektoren pro Bild



- Biologie: Microarray Daten
  - Ein Feature entspricht z.B. einem Gen im menschlichen Körper
  - Je nach Versuchsaufbau: 20.000 dimensionale Vektoren



- Biologie: Metabolomdaten
  - Ein Feature entspricht der Konzentration eines Stoffwechselprodukts im Blut
  - Je nach Messgenauigkeit: 50 – 2000 dimensionale Feature-Vektoren

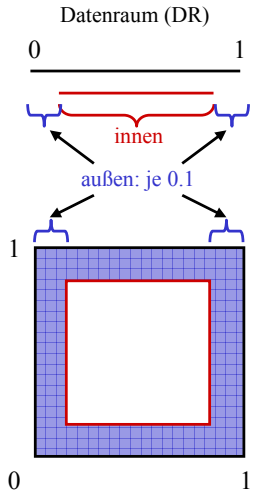
Grundprobleme

„Fluch der Dimensionalität“ (Curse of Dimensionality)

- *Distanz zum nächsten Nachbarn unterscheidet sich im Hochdimensionalen kaum von der Distanz zu einem beliebigen Nachbarn*

$$\frac{\text{nächsteNachbarnDist}}{\text{entferntesteNachbarnDist}} \approx 1$$

- *Die Wahrscheinlichkeit, dass Datenobjekte am Rand des Datenraumes liegen, steigt mit der Anzahl der Dimensionen exponentiell*



1D: $\mathcal{P}_{DR} = 1^1 = 1$; $\mathcal{P}_{innen} = 0.8^1 = 0.8$; $\mathcal{P}_{außen} = 1 - 0.8 = 0.2$

2D: $\mathcal{P}_{DR} = 1^2 = 1$; $\mathcal{P}_{innen} = 0.8^2 = 0.64$; $\mathcal{P}_{außen} = 1 - 0.64 = 0.36$

3D: $\mathcal{P}_{DR} = 1^3 = 1$; $\mathcal{P}_{innen} = 0.8^3 = 0.512$; $\mathcal{P}_{außen} = 1 - 0.512 = 0.488$

10D: $\mathcal{P}_{außen} = 0.893$

Grundprobleme

andere Interpretation des Curse of Dimensionality:

- Die Werte in jeder Dimension sind verrauscht.

  D.h. die Werte schwanken durch Störeinflüsse, die mit dem Objekt an sich nichts zu tun haben.
- bei zunehmender Dimensionalität wird die Summe der Störeinflüsse so groß, dass die beobachteten Unterschiede zwischen 2 Objekten von den Störeinflüssen dominiert werden.

  => Summe der Unterschiede hängt von der Ausprägung der Störeinflüsse ab und nicht von den Objekteigenschaften.

  => Abstand zwischen Objekt gleicht sich immer weiter an, da Störeinflüsse gleichverteilt über alle Objekte.

Grundprobleme

- Patterns und Modelle auf hochdimensionalen Daten sind oft schwer interpretierbar.

  $\Rightarrow$ Lange Entscheidungsregeln

- Effizienz bei hochdimensionalen Daten häufig problematisch.

  $\Rightarrow$ Indexstrukturen degenerieren auf hochdimensionalen Daten

  $\Rightarrow$ Distanzberechnungen werden i.d.R. teurer

- Muster treten nur in Teilräumen auf, aber nicht im gesamten Featureraum

- Cliquen von korrelierten Features dominieren das Objektverhalten.

2.2. Feature-Selektion

Idee: Bei sehr vielen Features sind nicht alle notwendig, um die Daten zu beschreiben:

- Features sind nicht aussagekräftig für ein Problem
- Informationsgehalt stark korrelierter Features ist fast identisch

Die Einschränkung auf einen Teilraum des gesamten Datenraums kann Verfahren effizienter und effektiver machen.

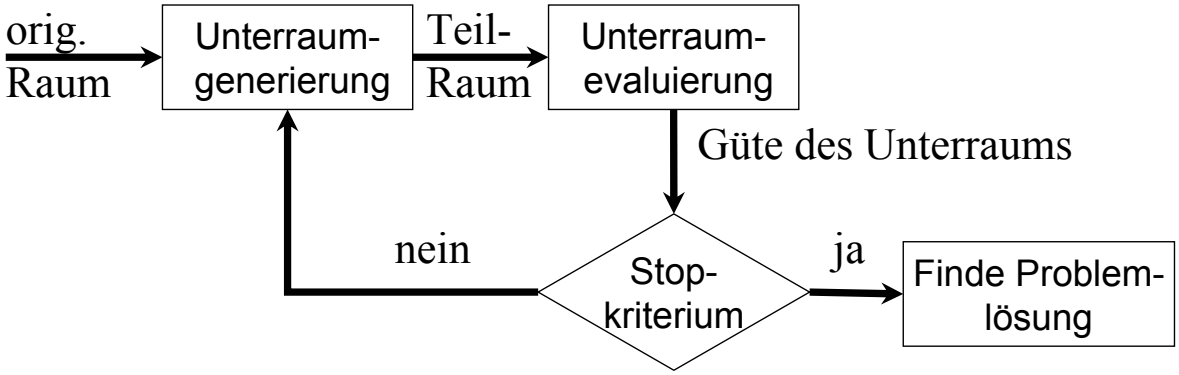
Lösung:
Streiche alle überflüssigen Dimensionen aus dem Featureraum.

Feature-Selektion

Gegeben: Vektorraum $F = D_1 \times .. \times D_n$ mit $D = \{D_1,..,D_n\}$.
Gesucht: Minimaler Unterraum M über $D` \subseteq D$, der für ein gegebenes Data Mining Problem eine optimale Lösung erlaubt.

- Minimalität erhöht Effizienz und verringert den Curse-of-Dimensionality
- Optimale Lösung ist ein sehr breiter Begriff, da Featurereduktion Vorverarbeitung zu sehr unterschiedlichen Problemstellungen sein kann.
- Problem ist sehr komplex, da es 2^n mögliche Unterräume gibt.
  => vollständige Suche wäre nur auf kleinen Dimensionalitäten möglich, bei denen Featureselektion nicht notwendig ist

Genereller Ablauf der Feature-Selektion



- obiges Schema deckt einen Großteil der Algorithmen ab
  (weitere Methoden denkbar:
  z.B. Clustering der Features im Objektraum)
- Unterscheidung anhand Unterraumgenerierung, Unterraumevaluierung und Stopkriterium.

Überblick über Teillösungen

1. Welcher der 2^n Unterräume müssen oder sollen bei der Suche untersucht werden? (Unterraumgenerierung und Suche)
   - Greedy-Ansätze
   - Heuristische Ansätze
   - optimale Ansätze mit Monotonie
   - Suchrichtung (Hinzufügen oder Löschen von Dimensionen)

2. Wann erlaubt ein Unterraum eine optimale Problemlösung? (Unterraumevaluierung)
   - monotone oder nicht monotone Kriterien
   - problemspezifische oder unspezifische Kriterien
   - supervised oder unsupervised Kriterien

Überblick über Teillösungen

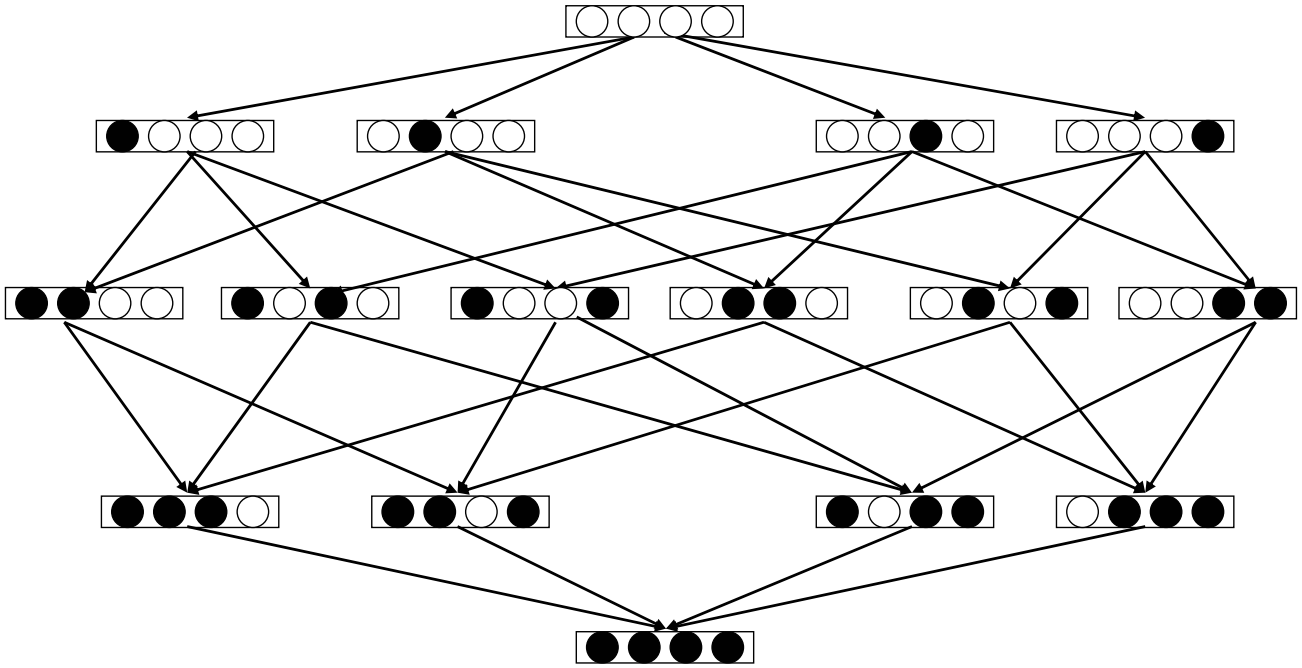
3. Wann kann der Algorithmus aufhören ? (Stop-Kriterium)
   - optimales Ergebnis wurde gefunden
   - alle Unterräume wurden untersucht.
   - vom Benutzer wird höchste Anzahl an Iterationen vorgegeben
   - Güte erreicht einen Mindestwert
   - eine angegebene Anzahl an Features wurde selektiert

Prinzipiell sind verschieden Lösungen der 3 Teilaufgaben miteinander kombinierbar.

Unterraumgenerierung und Suche

- *Backward-Elimination*: Beginne mit dem gesamten Featurespace *F* und entferne überflüssige Dimensionen.
- *Forward-Selection*: Untersuche einzelne Dimensionen D‘ $\in \{D_1,..,D_n\}$. und bilde neue Unterräume durch Kombination.
- *Greedy*: Kombiniere immer die besten Unterräume oder lösche immer die schlechteste Dimension.
- *Random*: Bilde zufällige Unterräume und evaluiere diese.
- Suche: Untersuche alle möglichen Unterräume , die durch Löschen oder Kombination bereits untersuchter Unterräume gebildet werden können.
- Branch and Bound-Verfahren: Schließe Lösungen, die nicht mehr maximale Qualität erreichen können aus.

Suchraum



Qualitätskriterien

- Modelabhängige Beurteilung (Wrapper-Methoden):
  Direkte Anwendung des Data Mining Algorithmus und Beurteilung des Ergebnisses.:
  - Klassifikation: Trainiere Klassifikator auf Unterraum und beurteile Genauigkeit
  - Clustering: Clustere Unterraum und berurteile Clusterqualität (Silhuetten-Koeffizient, Max. Likelihood, ..)

- Modellunabhängige Beurteilung (Filter-Methoden):
  - Supervised und Unsupervised Trainingdaten
  - Diskrete oder Reele Domänen
  - Beurteilung einer Dimension oder eines ganzen Teilraums

- Monotone Qualitätsmaße:
  erlaubt die Verwendung von Branch und Bound Algorithmen

Kriterien zur Evaluation von Teilräumen

Zentrale Frage: *Wie Aussagegleich sind 2 Repräsentationen derselben Datenmenge ?*

Repräsentationen können hier einzelne Features, Unterräume oder auch Klassenlabels sein.

Einteilung der Kriterien:

1. Vorwissen
   - mit Klassen gelabelt (Supervised Feature Selection)
   - ohne Einteilung in Klassen (Unsupervised Feature Selection)
2. Art der gemessenen Information
   1. Statistische und Informationstheoretische Ansätze
   2. Distanzbasierter Ansatz
   3. Konsistenzbasierter Ansatz

Einteilung nach Vorwissen

Supervised Feature-Selection:

Bewertung der Unterräume nach Korrelation zu den Klassen und Trennung der Klassen.

- Irrelevante Features sind nicht mit Klasse korreliert
- Redundante Features werden über Minimalität ausgeschloßen

Unsupervised Feature-Selektion:

Problem: Relevanz der Features kann nicht beurteilt werden, da kein Wissen über Ziel und Zweck des Datenraums

$\Rightarrow$ Elimination von redundanten Features

$\Rightarrow$ Beurteilung nach der Trennung potentieller Cluster (Wrapper-Ansatz)

Maße zur Bestimmung der Korrelation sind notwendig für beide Ansätze ! (Korrelation zwischen Klassen und Features oder zwischen Features)

1. Statistisch und Informationstheoretische Maße

- Basieren auf Verteilungen bzgl. der Klassen und der Features/Unterräume
  Achtung bei reellen Features ist Schätzung einer Zufallsvariable nicht direkt durchführbar. Daher:
  - Split zur Umwandlung von reellen Features in diskrete Features
  - Annahme über Verteilungsfunktion (z.B. Normalverteilung,..)

- Wie stark korrelieren die Verteilungen von Klassen und Features/Unterräumen?

- Wie gut kann Klassenlabel im Unterraum vorhergesagt werden?

- Wie stark unterscheidet sich Aufteilung bzgl. dieses Features/Unterraums von zufälliger Aufteilung ?

1. Statistische Verfahren

Idee: Bewerte wie gut jede Dimension die Klassen „unterscheidet“.

Kriterien: **Information Gain** (vgl. Entscheidungsbäume)

Zerlege Trainingsmenge anhand Feature/Unterraum in Teilmengen (Unterteilung: nach Werten oder Splitkriterien).
Die *Entropie* für eine Menge T von Trainingsobjekten ist definiert als

$entropie(T) = -\sum_{i=1}^{k} p_i \cdot \log p_i$ (p_i steht für Häufigkeit der Klasse i in T)

$entropie(T) = 0$, falls $p_i = 1$ für ein i
$entropie(T) = 1$ für $k = 2$ Klassen mit $p_i = 1/2$

$$informationsgewinn(T, t_i) = entropie(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot entropie(T_i)$$

Bei reelwertigen Attributen muss ein Split gefunden werden.

1. Statistische Verfahren

χ2-Statistik

Bewertet Unabhängigkeit einer Dimension von einer Klasse.
(Aufteilung der Daten anhand Splitwert s oder anhand diskreten Attribut werten)

$A = \left|\{o \mid x \le s \wedge Class(o) = C_j\}\right|$ Objekte in C_j mit Wert $x \le$ Splitwert

$B = \left|\bigcup_{l \ne j} \{o \mid x \le s \wedge Class(o) = C_l\}\right|$ Objekte anderer Klassen mit $x \le$ Splitwert.

$C = \left|\{o \mid x > s \wedge Class(o) = C_j\}\right|$ Objekte in C_j mit $x >$ Splitwert.

$D = \left|\bigcup_{l \ne j} \{o \mid x > s \wedge Class(o) = C_l\}\right|$ Objekte anderer Klassen, mit x > Splitwert

χ2-Statistik ist definiert durch: $\chi^2(t, C_j) = \frac{|DB|(AD - CB)^2}{(A+C)(B+D)(A+B)(C+D)}$

Je höher Maximum oder Durchschnitt über alle Klassen desto besser das Feature a:

$\chi^2_{\max}(a) = \max_{i=1}^{m}\{\chi^2(a, C_i)\}$ oder $\chi^2_{avg}(a) = \sum_{i=1}^{m} \Pr(C_i)\chi^2(a, C_i)$

1. Statistische Verfahren

Mutual Information (MI)

Maß für gegenseitige Abhängigkeit zweier Zufallsvariablen.

Hier: Vergleich der Abhängigkeit von der allgemeinen Klassenverteilung mit Verteilung in Dimension/Unterraum.

1. diskreter Fall.

$$I(X,Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

2. Kontinuierlicher Fall

$$I(X,Y) = \int_Y \int_X p(x,y) \log \frac{p(x,y)}{p(x)\,p(y)} dxdy$$

2. Distanzbasierte Kriterien

Idee: Unterraum ist gut, wenn Abstand von Objekten innerhalb einer Klasse durchschnittlich kleiner ist, als zwischen Objekten unterschiedlicher Klassen.

Qualitätsmaß:

Für alle Objekte o ∈ DB berechne den nächsten Nachbarn in Klasse C=Class(o) $NN_C(o)$ und den kleinsten nächsten Nachbarn $NN_{K \neq C}(o)$ in einer der anderen Klassen .

Güte des Unterraums U:

$$Q(U) = \frac{1}{|DB|} \cdot \sum_{o \in DB} \frac{NN^U_{K \neq C}(o)}{NN^U_C(o)}$$

3. Konsistenzbasierte Kriterien

Idee: Existieren im Unterraum U identische Vektoren u, v mit $v_i = u_i$ $1 \leq i \leq d$ aber unterschiedlichen Klassen labels $C(u) \neq C(v)$.

=> Unterraum ist inkonsistent

Maß für Konsistenz von U:

$X_U(A)$: Anzahl aller zu A identischen Vektoren

$IC_U(A)$: Inkonsistenz bzgl A in U

$$IC_U(A) = X_u(A) - \max_k X^k_u(A)$$

Für ganz U: $$IC(U) = \frac{\sum_{A \in S} IC_U(A)}{|S|}$$

Monotonie: $U_1 \subset U_2 \Rightarrow IC(U_1) \geq IC(U_2)$

3. Konsistenzbasierte Kriterien(2)

Vorteile:

- kann auf Unterräume angewendet werden und nicht nur auf Dimensionen
- Monotonie ermöglicht die effiziente Suche nach optimalen Unterräumen mit Branch and Bound

Nachteil:

- Anwendung beschränkt auf nominale Attribute (keine ordinalen oder reell-wertigen Attribute)

Beispiel 1: Greedy-Ansatz mit Information Gain

Gegeben: Klassifikationsproblem über F.

Ziel: Selektiere k Dimensionen

- Berechne IG für jede Dimension $D' \in \{D_1,..,D_n\}$ (bei reell-wertigen Attributen müssen alle möglichen Splitpunkte betrachtet werden)
- Sortiere Dimensionen $\{D_1,..,D_n\}$ nach IG
- Wähle die k besten Dimensionen

Nachteil:

- Dimensionen werden einzeln betrachtet: Klasse und Dimensionswert müssen direkt korreliert sein.
- Korrelierte Dimensionen: Auswahl von bedeutungsgleichen Dimensionen falls diese am stärksten mit der Klasse korreliert

Beispiel 2: Random-Ansatz mit Modellabhängiger Beurteilung

Gegeben: Klassifikationsproblem über *F*.

Ziel: Selektiere *k* Dimensionen

- Berechne die Klassifikationsgenauigkeit mit Überkreuzvalidierung für *n* zufällige Teilräume der Dimensionalität *k*.
- Wähle den Unterraum, in dem die beste Klassifikationsgüte gefunden wurde.

Nachteil:

- kann je nach Anzahl der zu testenden Unterräume sehr lange dauern.
- bester Unterraum mit *k* Dimensionen muss nicht im Sample enthalten sein

Beispiel 3: Branch and Bound mit Inkonsistenzkriterium

Gegeben: Klassifikationsproblem über *F*.

Ziel: Selektiere *k* Dimensionen

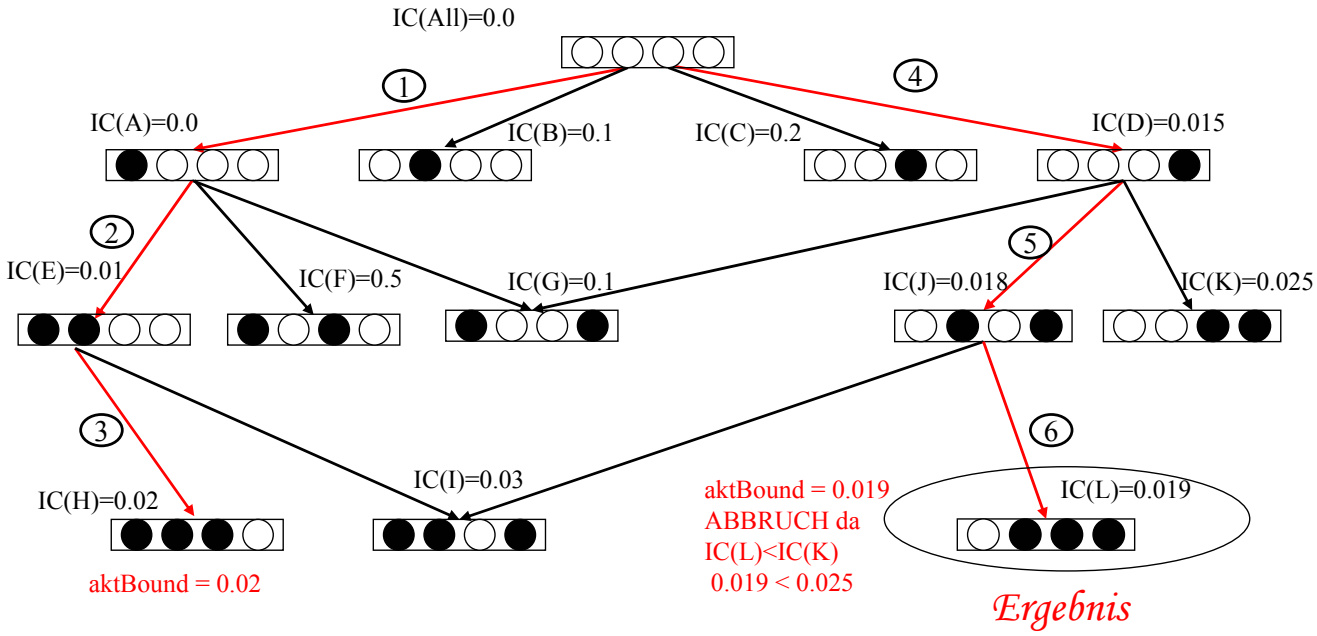
Backward-Elimination mit Branch and Bound:

```
FUNCTION BBwithInkonsistency(Featurespace F, int k)
        queue.init(ASCENDING);
        queue.add(F,IC(F))
        aktBound:= INFINITY;
        WHILE queue.NotEmpty() oder aktBound < queue.top() DO
                aktURaum := queue.top();
                FOR ALL Unterräume U von aktURaum DO
                        IF U.dimensionality() = k THEN
                             IF IC(U)< aktBound THEN
                                  aktBound  := IC(U);
                                  BestURaum := U;
                        ELSE
                             queue.add( U, IC(U));
        RETURN BestURaum
```

Beispiel 3: Branch and Bound mit Inkonsistenzkriterium

Beispiel: k = 1.

○ Feature selektiert ● Feature ausgeschlossen



Beispiel 3: Branch and Bound mit Inkonsistenzkriterium

Vorteile:

- liefert optimale Lösung bzgl. Inkonsistenzkriterium meist relativ effizient

Nachteile:

- Komplexität immer noch exponentiell da Suchraum immer noch exponentiell
- Anwendbarkeit des Inkonsistenzkriteriums ist auf nominale Attribute eingeschränkt.

Beispiel 4: Genetischer Algorithmus

Gegeben: Klassifikationsproblem über *F*.

Ziel: Selektiere *k* Dimensionen

Vorgehen: Genetischer Algorithmus

Gegeben:

- Population von Lösungen := Menge *k*-dimensionaler Unterräume
- Fitnesskriterium: Korrelationsmaß zwischen Klassen und Unterräumen
- Mutationregel und Mutationwahrscheinlichkeit:
  mit Wahrscheinlichkeit x % wird Dimension D_i in U durch D_j ersetzt
- Fortpflanzung: Kombinationsregel für 2 Unterräume U1 und U2:
  Wähle 50 % der Dimensionen aus U1 und 50 % aus U2
- Selektionsregel: Alle Kandidationräume die z % schlechtert Fitness haben als der beste der bisherigen Generation sind nicht lebensfähig.
- Freilos: Zusätzlich zur Selektion kann jeder Unterraum mit Wahrscheinlichkeit u % in die nächste Generation übernommen.

Beispiel 4: Genetischer Algorithmus

Ablauf:

```
Initialisiere Population
WHILE Max_Fitness > Old_Fitness  DO
   Mutiere Population gemäß Mutationsrate
  WHILE nextGeneration < PopulationSize DO
    Generiere neuen Kandidaten K durch Fortpflanzung
    IF K hat Freilos oder K ist fit enough THEN
      K darf in die nächste Generation
RETURN fittester Unterraum
```

Beispiel 4: Genetischer Algorithmus

Bemerkung:

- hier nur Skizze des Grundalgorithmus (viele Erweiterungen)
- Konvergenz meist nur unter „Simulated Annealing“ (Freiloswahrscheinlichkeit sinkt mit Anzahl der Generationen)

Vorteil:

- Vermeidung von lokalen Maxima
- häufig gute Approximation des optimalen Unterraums

Nachteile:

- kann lange Laufzeiten aufweisen
- viele Parameter müssen richtig gewählt werden, um einen guten Trade-Off zwischen Qualität und Laufzeit zu erzielen

Beispiel 5: Feature-Clustering mit Korrelation

Gegeben: Clusteringproblem über F.

Ziel: Reduziere Featurespace auf k Dimensionen.

Vorgehen: Da man irrelevante Attribute nicht erkennen kann, beschränkt man sich auf die Elemination von redundanter Information.

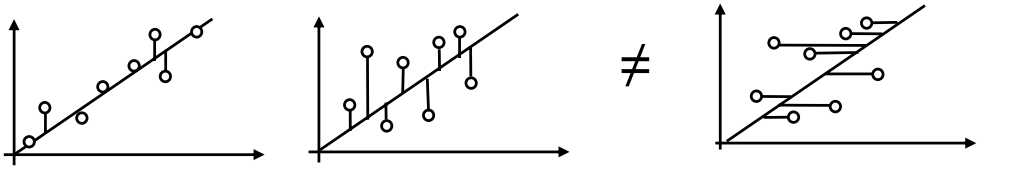
Idee: Clustere die Feature im Raum der Objekte und selektiere 1 Repräsentanten pro Cluster.

Maß für die Abhängigkeit Ähnlichkeit der Features:

- Korrelation zwischen 2 Features:

$$COR(X,Y) = \frac{COV(X,Y)}{\sqrt{VAR(X) \cdot VAR(Y)}^2}$$

- Regression: Bilde Regressionsgerade aus X für Y und messe quadratischen Fehler. Fehler klein => starke Korrelation.



stark abhängige Dimensionen

schwach abhängige Dimensionen

≠

(Achtung: Asymmetrie)

Beispiel 5: Feature-Clustering mit Korrelation

Maximaler Information Compression Index (MICI):

Idee: Messe den kleinsten Eigenwert der Kovarianzmatrix Σ zwischen beiden verglichenen Unterräumen.

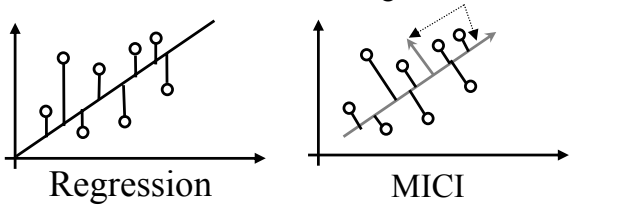
$$\det(\Sigma - \lambda E) = \det\begin{pmatrix} VAR(X)-\lambda & COV(X,Y) \\ COV(X,Y) & VAR(Y)-\lambda \end{pmatrix}$$

$$= (VAR(X)-\lambda)\cdot(VAR(Y)-\lambda) - COV(X,Y)^2$$

$$\Rightarrow 0 = \lambda^2 - \lambda\cdot VAR(Y) - \lambda\cdot VAR(X) + VAR(X)\cdot VAR(Y) - COV(X,Y)^2$$

$$\Rightarrow \lambda = \frac{(VAR(Y)+VAR(X)) \pm \sqrt{(VAR(Y)+VAR(X))^2 + 4\cdot 1\cdot VAR(X)\cdot VAR(Y) - COV(X,Y)^2}}{2\cdot 1}$$

$$MICI(X,Y) = VAR(Y)+VAR(X) - \sqrt{(VAR(Y)+VAR(X))^2 + 4\cdot VAR(X)\cdot VAR(Y) - COV(X,Y)^2}$$



MICI:
- Symmetrisch
- Kann aus beiden Dimensionen 1 gebildet werden, die die gesamte Information beider widerspiegelt.

Beispiel 5: Feature-Clustering mit Korrelation

Ablauf:
- Cluster Feature mit k-medoid Clustering für k Cluster und Abstandsmaß MICI.
- Selektiere die Clusterrepräsentanten als Featuredimensionen

Bemerkung:
- Versucht für jede Gruppe abhängiger Dimensionen eine representative Dimension
- Anwendung anderer Clustering Algorithmen denkbar.
  (K-Means: Wähle Dimension die am nächsten am Cluster-Centroid liegt)
- Häufig werden Cluster-Algorithmen für Streams verwendet, wegen Ihrer Laufzeit O(d)

Beispiel 5: Feature-Clustering mit Korrelation

Vorteile:

- Verhältnismäßig schnelle Selektionsmethode
- Kommt ohne Klasseneinteilung aus

Nachteile:

- Meist kein eindeutiges Ergebnis, da Clustering von Parametern und Reihenfolge abhängen kann.
- Repräsentative Dimensionen wechseln bei unterschiedlichen Cluster Algorithmen
- basiert auf paarweiser Korrelation

  =>höherwertige Dimensionen werden nicht untersucht.

Feature-Selektion

Diskussion:

- Viele Algorithmen basierend auf unterschiedlichen Heuristiken
- Feature können aus 2 Gründen eleminiert werden:
  - es existieren andere bedeutungsgleiche Feature (Redundanz)
  - Features sind nicht mit der Aufgabe korreliert
- häufig können auch schon nicht optimale Ergebnisse sowohl Effizienz als auch Effektivität verbessern
- Vorsicht: Selektierte Features müssen keinen direkten Einfluß auf Zielvariable haben, sondern können auch nur von den gleichen versteckten Einflüssen abhängen.