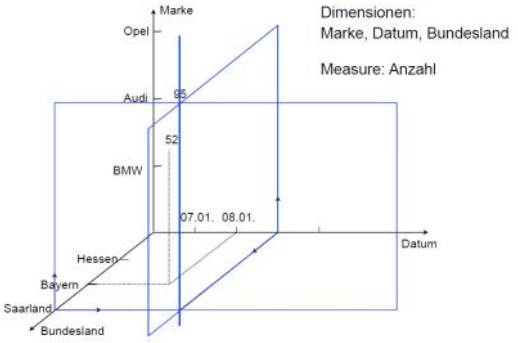
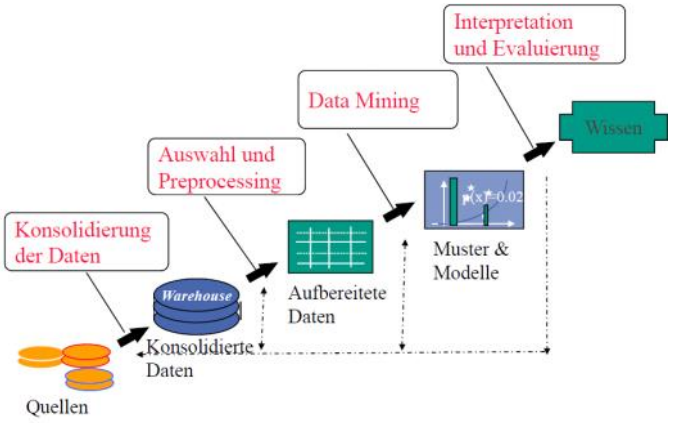
Analysetechniken für große Datenbestände 2

Dienstag, 16. Oktober 2018 12:10

Einleitung

- Bottom-up Vorgehensweise
  - Keine vorige Hypothese, Analyse und danach Frage, was Auffälligkeiten bedeuten
- Top-down Vorgehensweise
  - Anwender hat konkrete Hypothese, Überprüfung mit zB SQL-Abfragen
- Data-Cube
  - 

- Aktuelle Situation Datenbanksysteme
  - Große Datenmengen in unterschiedlichen Beständen verteilt, hohes Maß an Heterogenität
  - Probleme: Wissen ist nicht offensichtlich. Zuviele Attribute, niemand hat Überblick, wechselndes Personal
- Wissensextraktion als Prozess



  - Ersten drei Phasen sind Teil von AGD1/2
- Data-Warehouse
  - Repository mit Daten aus verschiedenen heterogenen Quellen
  - Daten sind verdichtet/aggregiert
  - Mehrere Sichten auf Datenbestand
  - Unterschiede zu normaler Datenbank:
    - Ausgezeichnete Rolle in IT-Infrastruktur, Sichten auf Datenbestand
    - Multidimensionales Datenmodell
    - Informationsbedürfnisse: Nur OLAP/für Warehouse geeignete Anfragen
- Datenströme
  - Eigenschaften
    - Endlose Folge von Datenobjekten
    - Erzeugt von einer/mehreren Quellen
    - Messwerte vs. Events
    - Konstanter verfügbarer Speicher
    - Strom mit mehrdimensionalen Daten =/= mehrere eindimensionale Ströme (können unsynchronisiert sein)
  - Unterschiedliche Informationsbedürfnisse auf Datenströmen
    - Herkömmliche Anfragen / One-Shot Queries
      - Welcher Aktienkurs ist gerade am höchsten?
    - Halbwegs neue Informationsbedürfnisse
      - Continuous Queries: Benachrichtige mich, wenn Aktienkurs unter Wert fällt
      - Change Detection
    - Weiterentwicklung bestehender Mining-Paradigmen
      - zB Clustering
  - Lösungsmuster zur Analyse von Datenströmen
    - Online/Offline-Processing: Online-Pr. passt Datenstruktur bei Eintreffen eines neuen Objektes an, Offline-Pr. leitet

Anrageergebnis aus der Datenstruktur ab.
        - Ausdünnen existierender Datenobjekte: Je älter Objekt, desto eher Löschung/Vergröberung
- Interessebasiert vs objektbasiert
    - Interessebasiert: Raum wird nach Clustern/Outlier abgesucht, siehe AGB1
    - Objektbasiert: Bei vielen Attributen ist jedes Objekt für bestimmte Kombination von Attributen auffällig, für bestimmte Objekte suche genau diese Attribute
    - (Zweiphasiges) Vorgehen: Subspace-Suche zuerst
        i. Finde interessante Teilräume
        ii. Durchsuche Teilräume nach Clustern/Outlier
        - Welche Teilräume sind interessant?
        - Triviale/Nichttriviale Outlier: TODO F1.52
    - Probleme bei Subspace Suche
        - Suchaufwand exponentiell mit #Attribute

Grundlagen

- Momente
    - k-ter Moment: $m_k := E(X^k)$
    - Absolutes Moment k-ter Ordnung: $M_k := E(|X|^k)$
    - Zentrales Moment k-ter Ordnung: $\mu_k := E((X-\mu)^k)$
    - Zentrales absolutes Moment k-ter Ordnung: $\overline{\mu_k} := E\left(|X-\mu|^k\right)$
    - $\mu_1 = 0$, $\overline{\mu_1} = E(|X-\mu|)$ (mittlere absolute Abweichung), $\mu_2 = E((X-\mu)^2)$ (Varianz)
    - Schiefe
        - rechtsschief: Wert kleiner als Mittelwert sind häufiger
        - $v(x) = \frac{\mu_3(X)}{\sigma^3(X)}$ ist Schiefe (dritter zentraler Moment normiert auf Standardabweichung)
        - v<0 => linksschief, v>0 => rechtsschief
    - Wölbung
        - Wölbung $\beta_2 := \frac{\mu_4(X)}{\sigma^4(X)}$
        - Exzess $\gamma := Wölbung\ -3$
        - gamma>0 => steilgipflig, gamma<0 => flachgipflig
- Hoeffding-Ungleichung:
    - Summe von Abweichungen von Zufallsvariablen von ihrem jeweiligen Erwartungswert
- Chernoff-Ungleichung
    - Dasselbe, aber Bernoulli-verteilte Zufallsvariable
- Poisson-Verteilung
    - Beispiel: Geschäft wird in einer Stunde erwartungsgemäß von λ=5 Kunden betreten. Wie groß ist die Wsk, dass in einer bestimmten Stunde genau k Kunden den Laden betreten? => $P_\lambda(k)$
    - X = Anzahl Kunden, die Laden in einer konkreten Stunde betreten. Summe $\sum X_i$ stochastig unabhängiger Poisson-verteilter Zufallsvariablen λ_i mit ist selbst wieder poisson-verteilt mit Parameter $\lambda = \sum \lambda_i$
- Log-Normalverteilung…
- Pearson-Koeffizient
    - $-1 \leq r_{A,B} \leq 1$
    - r_AB >0 => positive Korrelation, r_AB <0 => negative Korrelation für A und B
    - Korrelation =/= Kausalität. Pearson ist gut für lineare Zusammenhänge, aber nicht für andere Arten von Korreliertheit.
    - r_AB=0 =/= Unkorreliertheit!
- Mutual Information
    - Intuition: Wie sehr wird Unsicherheit einer Zufallsvariable X reduziert, wenn andere Variable Y bekannt ist?
    - $I(X,Y) = \int\int p(x,y) \cdot \log \frac{p(x,y)}{p_X(x) \cdot p_Y(y)} dy\, dx$
- Ähnlichkeitsmaße für Strings
    - Verschiedene Ähnlichkeitsmaße
        - Sequenzbasiert: Editierdistanz, Needleman-Wunch, Affine Gap, Smith-Waterman, Jaro, Jaro-Winkler
        - Mengenmasiert: Überlappung, Jaccard, TF/IDF
        - Hybrid: Verallgemeinerter Jaccard, soft TF/IDF, Monge-Elkan
        - Phonetisch: Soundex
    - Herausforderungen
        - Tippfehler: David vs Davod
        - Formatierungsdifferenzen: 10/8 vs Oct 8
        - Abkürzungen/Auslassungen: Daniel Walker Herbert Smith vs Dan W. Smith
        - Vertauschungen: Dept. of CS vs Computer Science Dept.
    - Sequenzbasiert
        - Editierdistanz
            - d(x,y) = minimale Kosten, um x in y zu transformieren (Word Error Rate)
        - Needleman-Wunch
            - Alignment: Buchstaben zweier Wörter x und y werden ausgerichtet, Lücken sind möglich.

        - Penale für affine Lücken (Affine Gap)
            - Erweiterung von Needleman-Wunch, Lücken sollen keine so großen Kosten erzeugen.
            - Lösung: Unterscheidung zwischen Kosten fürs "Öffnen" einer Lücke und fürs "Fortsetzen" einer Lücke
        - Smith-Waterman
            - Bisher immer Versuche, global alle Buchstaben aneinender auszurichten. Nicht passend für unterschiedliche Formatierungen oder Vertauschungen
            - Idee: Finde zwei Substrings von x und y mit maximaler Ähnlichkeit. "Lokale Ausrichtung"
            - Finde beste lokale Ausrichtung, diese ist die Bewertung von x und y.
        - Jaro
            - Sinnvoll für kurze Strings (zB Vor+Nachnamen)
            - Finde identische Buchstaben, die nah beieinander sind ($x_i = y_i \; und \; |i-j| \leq \min \frac{\{|x|,|y|\}}{2}$)
            - Formel siehe F2.71
- Ähnlichkeitsmaße in mehrdimensionalen Räumen
    - Mahalanobis-Distanz
        - Was ist der Abstand vom Clustermittelpunkt?
        - Kovarianzmatrix S charakterisiert Datenbestand
        - $d(\bar{x},\bar{y}) = \sqrt{(\bar{x}-\bar{y})^T S^{-1}/(\bar{y}-\bar{x})}$

Klassifikation, Association Rules, Clustering - fundamentale Datenanalysetechniken

- Klassifikation
    - Binäre Entscheidungsbäume: Aus AGB1. Siehe Splitentropie
    - Data-Reduction
        - Ziel: Deutlich geringeres Datenvolumen mit möglichst geringem Informationsverlust
        - Aggregation und Verallgemeinerung, Dimensionalitätsreduktion, Komprimierung, Numerosity Reduction (zB Clustering)
        - zum Teil Data-Mining Techniken
        - Auswahl von Attributen
            - Irrelevante Attribute weglassen.
            - "Attribut ist relevant bezüglich Klasse" := Werte dieser Attribute erlauben mit hoher Wsk eine Differenzierung zwischen Elementen dieser Klasse und anderen Klassen.
            - Bsp: Geburtsmonat wenig relevant bzgl Wohlstand, Automarke sehr relevant.
        - Information Gain
            - Es existiert eine Stichprobe aus Samples, jeweils bezüglich Klassen.
            - Wie überraschend ist Klassenzugehörigkeit eines Samples? => Entropie
            - Hoher Information Gain => Attribut diskriminiert gut (Attribut ist relevant)
            - InformationGain = EntropieDesSamples - ErwartungswertDesAttributs
            - Nutzung des IGs zum Aufbau von Entscheidungsbäumen
                - Diskreter Wertebereich
                    - Attribut mit höchstem Information Gain als Split-Attribut
                    - Baum ist nicht binär, ein Kind pro Attributwert
                - Kontinuierlicher Wertebereich
                    - Suche nicht nur Split-Attribut, sondern auch Split-Stelle
                    - Vorgehen: In Trainingsdaten vorkommende Werte des Attributs A sortieren: $a_{i1} \leq a_{i2} \leq \cdots \leq a_{in}$, potentielle Split-Stellen sind $\frac{a_{i_j}+a_{i_{j+1}}}{2}$, Bewertung des Splits wie üblich (Splitentropie).
        - Gain Ratio
            - Information Gain bevorzugt Attribut mit vielen Werten
                - Bsp: Attribut Kreditkartennummern. Entropie ist klein, jede Nummer diskriminiert gut => hoher Information Gain. Konstruierter Entscheidungsbaum: Für jede Nummer ein Blatt.
            - Maßnahme: Normalisierung, Attribute mit viel Unordnung in Trainingsdaten (wie Kreditkartennummern) penalisieren
            - Gain Ratio = (Information Ratio) / (Entropie des Attributs in Trainingsdaten)
        - Gini-Index
            - $Gini(D) = 1 - \sum_i p_i^2$
            - Beispiele
                - p1=1, p2=…=p10=0. Gini = 1-1 = 0
                - p1=…=p10=0,1. Gini = 1 - 10*0,1^2 = 0,9
- Association Rules
    - Vgl. Association Rules Folien aus AGB1. Siehe Frequent Itemsets, Minimum Support, Support/Confidence
    - Korrelation von Association Rules: Neu eingeführte Eigenschaft wie Support oder Confidence.
        - Verschiedene Korrelationsmaße: Lift, chi^2, all_confidence, max_confidence, Kulczynski/Kulc, Cosinus
        - Lift
            - $lift(A,B) = \frac{P(A \cup B)}{P(A) \cdot P(B)}$

- lift<1 => negative Korrelation
- lift=1 => Keine Korrelation
- lift>1 => positive Korrelation
- Nicht Null-Invariant

- χ^2-Wert
  - Chi^2-Wert fasst ganze Contingency-Tabelle zusammen, Lift nur eine Zelle.
  - Idee: Vergleiche tatsächliche und erwartete Werte und summiere Abweichungen

	game	$\overline{game}$	Σ_{row}
video	4000 (4500)	3500 (3000)	7500
$\overline{video}$	2000 (1500)	500 (1000)	2500
Σ_{col}	6000	4000	10.000

  - $$\chi^2 = \sum \frac{(\text{observed} - \text{expected})^2}{\text{expected}} = \frac{(4000-4500)^2}{4500} + \frac{(3500-3000)^2}{3000} + \frac{(2000-1500)^2}{1500} + \frac{(500-1000)^2}{1000} = 555{,}6$$

- all_confidence: $allconf = \frac{\sup(A \cup B)}{\max\{\sup(A), \sup(B)\}}$
- maximum_confidence: $maxconf = \max\{P(A|B), P(B|A)\}$
- Kulczynski: $Kulc(A,B) = \frac{1}{2}\left(P(A|B) + P(B|A)\right)$
- Cosinus: $cosine(A,B) = \frac{P(A \cup B)}{\sqrt{P(A) \cdot P(B)}} = \frac{\sup(A \cup B)}{\sqrt{\sup(A) \cdot \sup(B)}} = \sqrt{P(A|B) \cdot P(B|A)}$
  - Wie Lift, nur mit Wurzel oben.
  - Null-Invariant (im Gegensatz zu Lift) (TODO: WIESO?)
- Vergleich der Maße

	milk	$\overline{milk}$	Σ_{row}
coffee	mc	$\overline{mc}$	c
$\overline{coffee}$	$m\bar{c}$	$\overline{mc}$	$\bar{c}$
Σ_{col}	m	$\overline{m}$	Σ

(‚m' steht für ‚Milch', ‚c', für ‚Kaffee')

Data Set	*mc*	$\overline{mc}$	$m\bar{c}$	$\overline{mc}$	χ^2	*lift*	*all_conf*	*max_conf*	*Kulc*	*cosine*
D_1	10000	1000	1000	100000	90557	9,26	0,91	0,91	0,91	0,91
D_2	10000	1000	1000	100	0	1,00	0,91	0,91	0,91	0,91
D_3	100	1000	1000	100000	670	8,44	0,09	0,09	0,09	0,09
D_4	1000	1000	1000	100000	24740	25,75	0,5	0,5	0,5	0,5
D_5	1000	100	10000	100000	8173	9,18	0,09	0,91	0,5	0,29
D_6	1000	10	100000	100000	965	1,97	0,01	0,99	0,5	0,10

Lift in Zeile 1:
- (10000 /112000) / (11000/ 112000)²
- Kürzen: 10000 / (11000²/112000)

Lift in Zeile 2:
- (10000 / 12100) / (11000/12100)²
- Kürzen: 10000 / (11000²/12100)

  - Positive Korrelation zwischen m und c, da mc >> $\overline{m}$c und mc >> m$\bar{c}$ für sowohl D1 und D2. Lift ist aber sehr verschieden (andere Kennzahlen sind gleich)
  - Vergleiche Formel für Lift: Lift hängt von $\overline{mc}$ ab ("Wie viele Leute kaufen weder Milch noch Kaffee"), aber was hat das mit dem Zusammenhang Milch und Kaffee zu tun?
  - Übungsaufgabe: Berechne Angaben in Abhängigkeit der Variablen mc, m, c, ..
  - Außerdem: Bei D3 existiert negative Korrelation zwischen m und c, Lift für D3 ist aber größer als bei D2.
  - D5: Von 11000 Milchtrinkern trinken wenige Kaffee, von 1100 Kaffeetrinkern trinken viele Milch.
    - all_conf, cosine sehen negativen Zusammenhang, Kulc neutral, max_conf bewertet als positiven Zusammenhang
- Null-Invarianz
  - Null-Transaktion = Transaktion, die keins der betrachteten Items enthält
  - Maß ist Null-Invariant: Anzahl der Null-Transaktionen bleibt außen vor.
  - Bei Lift nicht der Fall (bei den anderen schon)
  - Nicht der Fall bei Support-Berechnung in all-confidence sowie in sämtlichen bedingten Wahrscheinlichkeiten
- Imbalance-Ratio: $IR(A,B) = \frac{|\sup(A) - \sup(B)|}{\sup(A) + \sup(B) - \sup(A \cup B)}$

- Clustering
  - Problemstellung
    - Finde Teilmengen der Datenobjekte, die jeweils nahe beieinander liegen
    - Unterschiedliche Problemstellungen, zB: k-Median, SSQ-Minimierung, facility location
    - k-Median
      - Wähle aus Menge N von Datenobjekten k aus (c1,…,ck), sodass $\sum_{i=1}^{k} \sum_{x \in N_i} d(x, c_i)$ minimal ist mit $N_i = \left\{x \in N \middle| \forall j \in \{1, \dots, k\}: d(c_j, x) \geq d(c_i, x)\right\}$.
    - SSQ Minimization
      - Wie k-Median, nur: Quadratsumme statt linearer Summe, beliebige Positionen als Centroide möglich (nicht nur die von Datenobjekten)
    - facility location
      - k nicht vorgegeben, aber jeder Median/Centroid, auch *facility*, verursacht Kosten (facility costs)

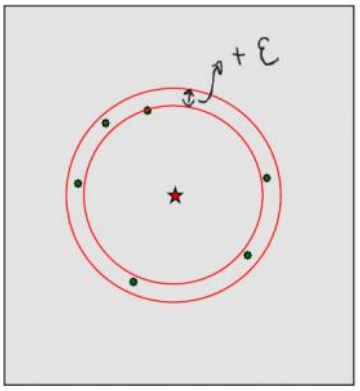
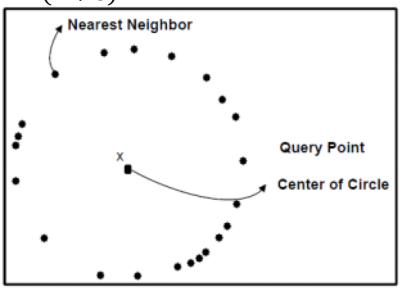
        - Finde $C \subseteq N$, sodass $FC(N,C) = z \cdot |C| + \sum_{i=1}^{|C|} \sum_{x \in N_i} d(x, c_i)$ minimal.
    - Für diese Problemstellungen existieren Approximationsalgorithmen mit Garantien.
- Graph Clustering
    - Graph Cluster: Menge von Knoten, zwischen denen relativ viele Kanten sind ("hohe Kantendichte")
    - Etablierte Lösungen
        - min-cut Berechnung
        - Bestimmung von Quasi-Cliquen
        - Finden dichter Teilgraphen
    - Unterscheidung zwischen
        - Clustering eines großen Graphen (wie eben)
        - Clustering einer Menge von Graphen (dh Graphen sind als Ganzes Objekte, verwende Abstandsmaß zwischen zwei Graphen zur Ähnlichkeit)
    - Weitere Untersuchungen:
        - Berücksichtige sowohl Attributwerte der Knoten als auch Kantenstruktur.
        - Berücksichtige Kantenlabels
- Alternate Clustering
    - Idee: Clustering wurde gefunden. Wie kann noch geclustert werden?
    - Ziel: Finde Cluster, das sich vom bereits gefundenen Clustering erheblich unterscheidet und gut ist (>Schwellwert)
    - COALA Verfahren
        - Basiert auf hierarchischem (agglomerativem) Clustering (siehe AGD1)
        - Beliebiges zugrundeliegendes Abstandsmaß
        - Notation
            - D={x1, …, xn} - Datenobjekte
            - C={c1, …, cp} - bestehendes Clustering
            - S={s1, …, sr} - neues Clustering
            - (Also Resultat ist eine Menge, obwohl agglomeratives Clustering eine Hierarchie liefert. => Waagerechter Schnitt durch Baum)
        - Welche beiden Cluster werden gemerged? (bei agglomerativem Clustering)
            - Schouldnot-link Constraint: Paar von Datenobjekten, die nicht im gleichen Cluster von S sein dürfen. Alle Paare von Objekten aus gleichem Cluster in C bilden dieses Constraint.
            - d(ci, cj) - Abstand zwischen Clustern ci und cj. Hier: average-linkage
            - In jedem Schritt Berechnung von zwei Cluster-Paaren:
                - Qualitatives Paar: Paar mit kleinstem Abstand (Basisalgorithmus würde hier mergen)
                - Unähnliches Paar: Paar mit kleinstem Abstand aller Paare, die Shouldnot-Link Constraint erfüllen, also im gleichen Cluster sein dürfen.
            - Merge des qualitativen Paares: Alternate-Ziel nicht erreicht.
            - Merge des unähnlichen Paares: Qualitätsanforderung i.Allg. nicht erfüllt. Schwierig, unähnliches Paar zu finden.
            - Vorgehen: Ist $\frac{Abstand\ qualitatives\ Paar}{Abstand\ unähnliches\ Paar} > Schwellenwert$?
        - Evaluation: Wie groß sind Präzision und Recall des alternate-Schritts?
            - Recall: Wieviel tatsächliche Cluster sind im Ergebnis?
            - Präzision: Wieviele Ergebnis-Cluster sind im tatsächlichen Cluster?
    - Alternate Subspace Clustering
        - In hochdimensionalen Räumen bedeutet Unähnlichkeit sowohl unterschiedliche Objekte und Dimensionen. Es gibt kein Alternate Clustering Verfahren für hochdimensionale Räume. Alternate Subspace Clustering?

Hochdimensionale Daten

- Spatial Data
    - Dimensionen korrespondieren mit (zB geografischem) Koordinatensystem
- High-dimensional Data
    - Hochdimensionale Punkte werden zu Featurevektoren abstrahiert
- Patterns hidden in subspaces
    -

object ID	age	income	blood pressure	sport activ.	profession
1	50	51.000			
2	49	48.000			
3	52	54.000			
4	47	50.000			
5			110	football	
6			112	football	
7			108	football	
8	18				student
9	19				student

    - Bei mehr und mehr Attributen werden Objekte einzigartig
    - Irrelevante Attribute tragen zu allgemeiner Ungleichheit der Objekte bei
    - Übliche Methoden können keine Muster mehr erkennen

- Mit steigender Dimensionalität, Distanzen zwischen Objekten steigen auch, Kontrast wird kleiner
  - vgl. Distances Grow Alike (Vektorbeispiel in 2D) auf F4.9
- Concentration Phenomenon
  - Mit steigender Dimensionalität sinkt der Kontrast durch gewöhnliche Metriken
  - Die Verteilungen von Normen/Distanzen in einer Verteilung von Punkten konzentrieren sich
  - Theorem "Concentration of the Norms and Distances"
    - y ist ein d-dimensionaler vektor [v1, …, vd] mit unabhängigen und identisch verteilten Komponenten
    - Dann gilt für Mittelwert und Standardabweichung:
      - $\mu_{||y||} = \sqrt{a \cdot d - b} + O(d^{-1}) \quad and \quad \sigma_{||y||})O(d^{-\frac{1}{2}})$
      - Mit Parameter a und b, die nur von den zentralen Momenten der Ordnungen 1,2,3,4 abhängen
      - Die Norm von zufälligen Variablen wächst abhängig zu $\sqrt{d}$, aber die Varianz wird kleiner für ausreichend große d.
      - Mit steigender Dimensionalität wird der relative Fehler durch Verwendung von $\mu_{||y||}$ statt $\left||y|\right|$ vernachlässigbar.
- Instabilität von hohen Dimensionalitäten
  - Nachbarschaften basieren auf Konzept: Ähnliche Objekte sind in derselben Nachbarschaft, unähnliche nicht.
  - Bei geringem Kontrast von Distanzen: kNN Distanzen sind fast identisch, k-nächster Nachbar ist ein zufälliges Objekt. => Alle Objekte sind in derselben Nachbarschaft.
  - Die Situation ist *instabil (unstable).*
  - 
- NN-Instability Result
  - Definition: Eine NN-Abfrage ist instabil für ein ε, wenn die Distanz von der Abfrage zu den meisten Datenpunkten ist geringer das $(1 + \varepsilon)$ fache der Distanz vom Abfragepunkt zu seinem nächsten Nachbarn.
  - 

  - Mit steigender Dimensionalität konvergiert die Wahrscheinlichkeit der Instabilität einer Abfrage auf 1.
  - Für einen d-dimensionalen Abfragepunkt Q und N d-dimensionale Samplepunkte X1, …, XN (unabhängig und identisch verteilt) sei definiert

$$DMIN_d = min\{\{dist_2(X_i, Q)|1 \leq i \leq N\}\}$$

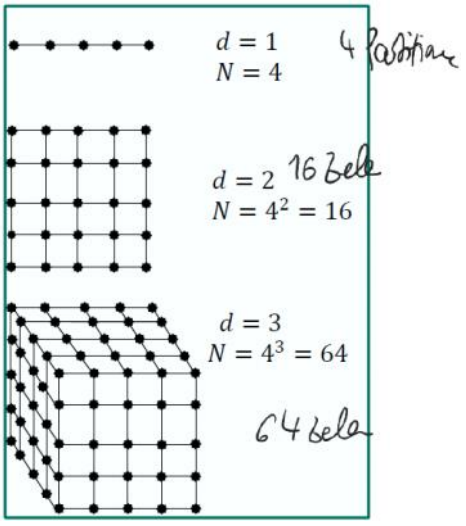
$$DMAX_d = max\{\{dist_2(X_i, Q)|1 \leq i \leq N\}\}$$

  - THEOREM Dann gilt:
    - Für gegebene Sequenz von Datenverteilungen (nicht Datenpunkte!), Sequenz von Abfrageverteilungen (Abfrage = Punkt), Q_m (zufällige Abfrage von der m-ten Abfragesequenz), P_m (zufälliger Datenpunkt der m-en Datenverteilung):
    - Wenn $\lim_{m\to\infty} VAR(\frac{d_m(P_{m,1},Q_m)}{E[d_m(P_{m,i},Q_m)}$ gilt,

      dann gilt für alle $\varepsilon > 0$: $\lim_{m\to\infty} P\left[DMAX_m \leq (1+\varepsilon) \cdot DMIN_m\right] = 1$
    - Das Konzept "Nächster Nachbar" hat keine Bedeutung mehr
- Gamma Funktion
  - Erweiterung der Fakultät auf negative und reale Zahlen
  - Definition für Integers: $\Gamma(n) = (n-1)!$
  - Definition für reale Zahlen: $\Gamma(z) \coloneqq \int_0^\infty t^{z-1} \cdot e^{-t} \cdot dt$
  - Anforderungen an Fakultät gelten: $\Gamma(1) = 1, \Gamma(t+1) = t \cdot \Gamma(t)$
- Expected NN-Distance
  - In einem d-dimensionalen Datenraum mit N uniform verteilten Punkten und Abfragepunkt Q - Wie viele Punkte liegen um Abfragepunkt mit Abstand r (Ziel: Schätze die Anzahl k der Punkte in einer d-dimensionalen Sphäre mit Zentrum Q und Radius

r)

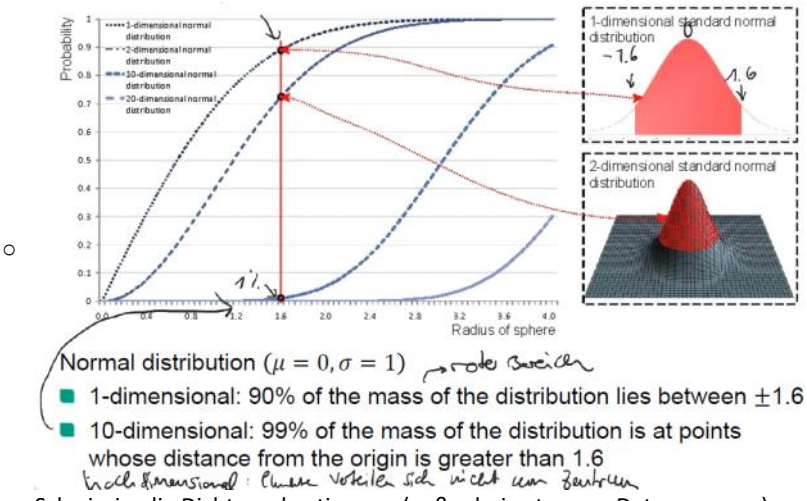
- $k = N \cdot \frac{V_{sphere}(r)}{V_{cube}(1)} = N \cdot \frac{V_{sphere}(r)}{1} = N \cdot V_{sphere}(r) = N \cdot \frac{\sqrt{\pi^d} \cdot r^d}{\Gamma\left(1+\frac{d}{2}\right)}$
- Gesucht: Notwendige Größe der Sphäre sodass k=1 gilt.
  - $r = \sqrt[d]{\frac{k \cdot \Gamma\left(1+\frac{d}{2}\right)}{N \cdot \sqrt{\pi^d}}}$
- Probleme: Stochastisch nicht exakt, Randfälle nicht betrachtet, NN-distanz ist tatsächlich größer
  - Mit steigender Dimensionalität ist r größer als der Datenraum selbst!!

- Multi-Dimensional Scaling
  - Datensatz von Punkten in hochdimensionalen Datenraum gegeben. Projeziiere Punkte in niedriger dimensionalen Raum sodass Distanzen zwischen Objekten möglichst erhalten bleiben.
- Empty Space Phenomenon
  - Grundlegende Annahmen über Volumenverhalten (Sphäre, Würfel) und Verteilungen aus niedrig dimensionalen Räumen gelten bei hochdimensionalen Räumen nicht mehr. Räume sind sparse/empty. Wahrscheinlichkeit eines Objektes in einem fixen Bereich nähert sich 0.
  - Partitioniere d-dimensionalen Raum in Partitionen der Größe 1/m, Anzahl der Zellen $N = m^d$ wächst exponentiell mit d.
  - In niedrig dimensionalen Räumen: Wenig leere Partitionen, viele Punkte pro Partition.
  - In hochdimensionalen Räumen: Mehr Partitionen als Punkte, viele leere Partitionen.
  - 

  - Beispiel: Daten werden in für jede Dimension in zwei Hälften partitioniert. für d Dimensionen werden 2^d Partitionen erzeugt. Für $N = 10^6$ Samples macht die Partition für d<=10 Dimensionen Sinn, für d=100 existieren 10^{30} Partitionen, sodass die meisten Partitionen leer sind.
  - Wahrscheinlichkeit, dass Datenpunkt in d-dimensionalen Hyperwürfel der Kantenkänge s liegt, entspricht s^d.
  - => Range Queries sind in hochdimensionalen Datenräumen nicht sinnvoll.
  - Betrachte maximal große sphärische Abfrage in einem d-dimensionalen Datenraum. Anteil des Datenraums, der durch die sphärische Abfrage gedeckt ist, entspricht

$$f_d = \frac{\sqrt{\pi^d} \cdot r^d}{\Gamma\left(1+\frac{d}{2}\right) \cdot (2r)^d} = \frac{\sqrt{\pi^d}}{\Gamma\left(1+\frac{d}{2}\right) \cdot 2^d}$$

- Hypervolumen einer dünnen sphärischen Hülle
  - Betrachte d-dimensionale Sphäre mit Volumen von 1. Volumen der Hülle (des Rands) der Sphäre mit einer Dicke von ε entspricht $(1^d - (1-\varepsilon)^d)/(1^d)$. Mit steigendem d, der Quotient nähert sich 1.
  - => In hochdimensionalen Datenräumen enthält eine dünne Hülle um eine Sphäre fast ihr ganzes Volumen.
- Importance of Tails

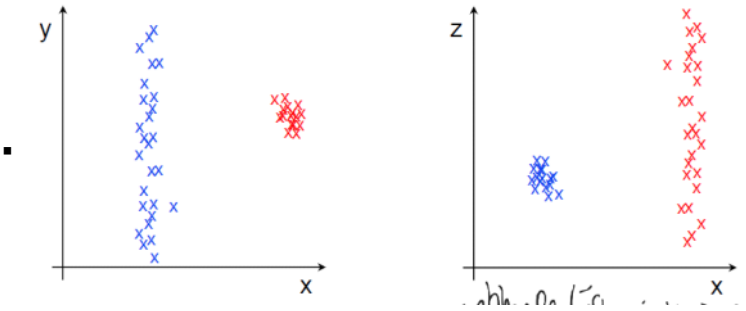
- 

  Normal distribution ($\mu = 0, \sigma = 1$)
  - 1-dimensional: 90% of the mass of the distribution lies between ± 1.6
  - 10-dimensional: 99% of the mass of the distribution is at points whose distance from the origin is greater than 1.6
  - Schwierig, die Dichte zu bestimmen (außer bei extremen Datenmengen)
  - In sehr hohen Dimensionen, alle Daten werden in den "Tails" liegen
- Baumbasierte Indexstrukturen (zB kd-Baum) sind nicht sinnvoll für hochdimensionale Daten
  - Bei hoher Dimension, viele Dimensionen dienen nicht mehr als Splitdimension
  - Curse of Dimensionality
- Zusammenfassung
  - High dimensional data:
    - in many applications large number of attributes
    - curse of dimensionality poses additional challenges
      - distances grow more and more alike
      - neighborhoods become meaningless
      - space partitions become empty
    - patterns hidden in subspaces disappear in high dimensional data
    - ➢ traditional methods are not able to detect patterns

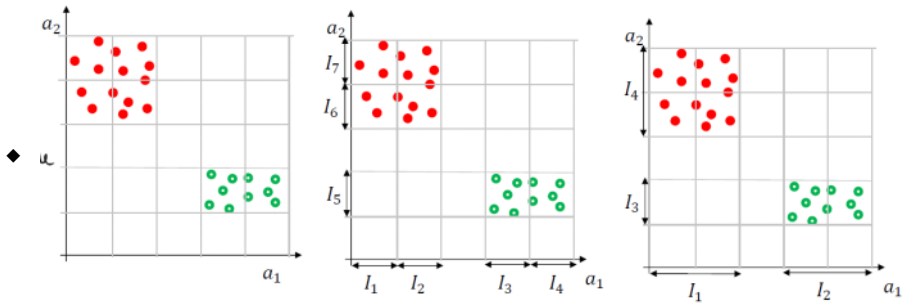
    Advanced data mining algorithms:
    - ➢ identify relevant dimensions (subspaces)
    - ➢ restrict distance computation to these subspaces
    - ➢ enable detection of patterns in projection of high-dimensional data

Projected Clustering (weggelassen)

- Erinnerung Chi-Quadrat Test: Sind zwei Verteilungen (von ordinalen/diskreten Ereignissen) unabhängig voneinander?
- Einführung Projected Clustering
  - Ziel: Nur interessanten Teilraum betrachten bei hochdimensionalen Daten.
  - Projected Clustering vs Subspace Clustering
    - Projected Clustering
      - Finde für jeden Cluster eine Untermenge der Dimensionen, die für den Cluster spezifisch sind
      - Paradigma ist Clustering-orientiert
      - Jedes Objekt gehört zu maximal einem Cluster
      - Größe der Ausgabemenge ist per Parameter fix.
    - Subspace Clustering
      - Finde Cluster in beliebiger möglichen Subspace Projekten für eine gegebene Cluster Definition (zB DBSCAN oder gridbased-model)
      - Objekt kann zu mehreren Clustern gehören
      - Größe der Ausgabemenge kann extrem hoch werden (hohe Redundanz)
- k-Means based Projected Clustering
  - k-Means: Erinnerung aus AGB1
    - Iterativer Hill-Climbing Algorithmus. Verschiebe jeden Medoiden in Richtung des Schwerpunkts der ihm zugeordneten Punktemenge.
    - Problem: Interessante Cluster sind in allen Dimensionen vielleicht nicht mehr interessant.
    - ```
      Wähle Seeds;
      repeat
        ordne jedes Datenobjekt
          dem ihm nächsten Medoid zu;
        berechne Schwerpunkte dieser Partitionen;
      until Verbesserung nicht mehr signifikant
      ```
      ```

- 

  - Projective Cluster
    - Sei S eine Menge von d-dimensionalen Objekte.
    - Ein projektives Cluster ist ein Tupel (C,D) mit
      - $C \subseteq S$ Menge von Objekten aus dem projektiven Cluster
      - $D \subseteq \{1, \dots, d\}$ der Subspace des projektiven Clusters (Menge von Dimensionen des pr. Clusters)
  - Projected Clustering: Erinnerung aus AGB1
    - Eingabe: Anzahl k der zu findenen Clusters, durchschnittliche Anzahl d der Dimensionen pro Cluster.
    - Ausgabe: Partitionierung der Daten in k+1 Cluster (zusätzliches Cluster für Noise)
    - Teilmenge Di der Menge der Dimensionen für jeden Cluster i.
    - ```
      Wähle Seeds;
      repeat
        bestimme Dimensionen zu jedem Medoid;
        ordne jedes Datenobjekt
          dem ihm nächsten Medoid zu;
        berechne Schwerpunkte dieser Partitionen;
      until Verbesserung nicht mehr signifikant
      ```
    - Vgl. Lokalität bei Projected Clustering.
- Statistics based Clustering
  - P3C
    - Projected Clustering via Cluster Cores
      - P3C verfährt bottom-up auf den Dimensionen der Projektion
      - Daten werden in Subspace geringerer Dimension projeziert.
      - Intervalle mit ungewöhnlich vielen projezierten Objekten werden erkannt.
    - Definitionen
      - Gegeben eine Datenbank DB von #n d-dimensionalen Objekten
      - Projeziertes Cluster: Tupel (X_i, Y_i) aus
        - Untermenge X_i von DB
        - Untermenge Y_i von Attributen sodass die Projektion der Objekte in Xi entlang der Attribute $a \in Y_i$ eine geringere Varianz als die des gesamten Datensatzes zu a hat
        - Die Objekte in X_i sind gleichverteilt entlang aller Attribute, die nicht in Y_i sind.
      - Intervall $I = [V_I, V_U]$ auf einem Attribut a_j: enthält alle Werte x, sodass $V_I \leq x \leq V_U$
      - Breite eines Intervalls I: $width(I) = V_U - V_I$
      - SupportSet eines Intervalls I: $SuppSet(I) = \{x \in D | x.a_j \in I\}$
        - $=$ Menge der Punkte, die im Intervall liegen
      - Support eines Intervalls I: $Supp(I) = |SuppSet(I)|$
      - p-signature S ist eine Menge $\{I_1, \dots, I_p\}$ von Intervallen von p *verschiedenen* Attributen
      - Support-Set einer Signatur S: $SuppSet(S) = \{x \in D | \bigcap_{i=1}^{p} SuppSet(I_i)\}$
    - Berechnung der Intervalle
      - Ziel: Berechne für jedes Attribut Intervalle, die Projektionen von p-Signaturen auf Attribute gut approximieren (oder übereinstimmen).
      - Attribute mit einer Gleichverteilung müssen identifiziert werden, für sonstige Attribute müssen Intervalle mit unüblich hohem Support identifiziert werden.
      - Verwendet wird **Chi-Square (Chi-Quadrat) goodnees-of-fit** Test
        - Jedes Attribut wird in die gleiche Anzahl von gleichgroßen Bins aufgeteilt
        - Anzahl der Bins: $m = \lfloor 1 + \log_2(n) \rfloor$
        - Für jedes Attribut werden die Daten auf eine Gleichverteilung innerhalb der Bins getestet
        - Chi-Quadrat Test: $\mathrm{X}^2 = \sum_{i=1}^{m} \frac{(h_i - h_E)^2}{h_E}$
          - h_i: beobachteter Support des i-ten Bins
          - h_E: erwarteter Support eines Bins
        - => Zähle für jeden Bin, wieviele Objekte dort sind und vergleiche mit der erwarteten Zahl von Objekten
        - Daten sind nicht nicht gleichverteilt in den Bins wenn $\mathrm{X}^2 > \mathrm{X}^2_{1-\alpha,m-1}$
          - m-1 : Degrees of Freedom
          - α: 0:001 Signifikanz-Level
        - Auf nicht gleich-verteilten Attributen wird der Bin mit dem höchsten Support markiert, der Test wird für unmarkierte Tests wiederholt. Am Ende werden aneinandergrenzende markierte Bins gemerged.
      ```



- Cluster Cores
  - Eine p-Signatur S={I1, …, Ip} einschließlich ihres SupportSets(S) heißt Cluster Core falls:
    - Für alle q-Signaturen $Q \subseteq S$ und alle $I \in S \backslash Q$: $Supp(Q \cup \{I\}) \gg ESupp(Q \cup \{I\})$
    - Für alle Intervalle $I \notin S$: $\mathrm{Supp}(S \cup \{I\}) \not\succ\!> \mathrm{ESupp}(S \cup \{I\})$ (Merge, sodass Menge maximal wird)
    - Beispiel (an obiger Illustration): p-Signatur $\{I_1, I_4\}$, Schnittmenge $Q = \{I_4\}$ des Intervalls $I = I_1$ muss weit mehr Punkte enthalten als $\frac{Supp(I_4)}{width\ of\ I_1}$.
  - Erinnerung Poisson-Verteilung aus AGB1
    - Beispiel: Laden wird in einer Stunde erwartungsgemäß von 5 Kunden betreten ($\lambda = 5$). Wie groß ist Wsk, dass in einer bestimmten Stunde genau k Kunden den Laden betreten?
    - Wann ist Wsk so klein, dass der Wert von λ offensichtlich nicht zutrifft?
    - $P_\lambda(k) = \frac{\lambda^k}{k!} \cdot e^{-\lambda}$
  - Berechnung der Cluster Cores

- Cell based Clustering

[weggelassen]

Change Detection

- Einleitung
  - Vgl. kontinuierliche Datenströme aus Einleitung
    - Phänomen mit zeitlichem Verlauf (zB Aktienentwicklung, Sensordaten). Andere häufige Enventualität: Viele Datenobjekte, eigentlich unabhängig voneinander (zB Eingehende Suchanfragen)
    - Bei letzterem: Outlier Detection und Clustering. Bei ersterem: Change Detection
  - Problem: Concept Drift: Verteilung einer Zielvariable ändert sich über die zeit
  - Change Detection: Erkenne Concept Drift kontinuierlich
  - Definition Change Detection
    - Gegeben Datenstrom x1, …, xt, unabhängige Datenobjekte generiert von unbekannter Verteilung Dt
    - Change Point zum Zeitpunkt t+1: Dt =!= D_t+1 (Also Verteilung verändert sich)
    - Change Detection: Erkenne Change Points und gib Alarm aus.
    - Herausforderungen: Echte Änderungen schnell erkennen, False Positives vermeiden
  - Qualitätskriterien
    - Mean Time between False Alarms (MTFA, false positives)
    - Missed Detection Rate (false negatives)
    - Mean Time to Detection (MTD)
      Average Run Length ($\mathrm{ARL}(\delta)$ - durchschnittliche Dauer, bis Änderung in Höhe δ erkannt wurde)
- ADWIN: Adaptive WINdowing
  - Vgl. Hoeffding
  - Sliding Window W: Betrachte die neusten n Werte (Diese sind das Window)
  - Jeder Wert x_t wird von Verteilung Dt erzeugt. Dt's sind unabhängig voneinander.
  - $\mu_t = Erwartungswert\ von\ D_t, \hat{\mu}_W = beobachteter\ Mittelwert\ im\ Fenster$
  - Oft wird Fenstergröße n als Parameter gewählt
    - Schwierig, da: Zu klein => Gute Darstellung der Verteilung, aber evtl versehentlicher Alarm. Zu Groß => Akkuratere Modelle in stabileren Phasen (mehr Trainingsdaten), aber Change Detection zu träge.
    - Adaptive Windowing: Fenstergröße anpassen.

        - Fenster vergrößern, wenn keine Änderung in Verteilung aufritt.
        - Fenster verkleindern, wenn Änderung auftritt.
        - Zu widerlegende Null-Hypothese durch Adaption: "Mittelwert im Fenster ist konstant". Zu wiederlegen mit Konfidenz δ
    - Grundidee
        - Partitioniere W in W0 und W1, berechne jeweils Mittelwerte. Sind Mittelwerte unterschiedlich genug? Wohl Change, verkleinere Fenster.
        - Alternativ: Wie lassen Fenster W so groß wie möglich, sodass Nullhypothese mit δ gilt.
    - Algorithmus
        1. Initiatize Window W;
        2. **foreach** t > 0 **do** {
        3. add x_t to the head of W;
        4. **while** ($|\hat{\mu}W0 - \hat{\mu}W1| \geq \epsilon_{cut}$ for some split W = W0W1) **do**
        5. drop elements from the tail of W;
        6. output $\hat{\mu}W$; }
    - Schwellwert ε_{cut} hängt von Konfidenzniveau δ ab und basiert auf Hoeffding-Ungleichung
    - ADWIN2
        - Kritik an ADWIN: Betrachtung von *allen* Subfenstern, die genügend groß sind
        - ADWIN2 mit neuer Datenstruktur: Fenster wird in Buckets variabler Größe aufgeteilt.
        - Neue Items werden in neuen Buckets eingeführt, Buckets werden ggf. Gemerged
            - x_t = 1:

1010101	101	11	1	1	1
Content: 4	2	2	1	1	1
Capacity: 7	3	2	1	1	1

            - 3 > M = „Zwei Buckets mit Content=1" ⇒ compress

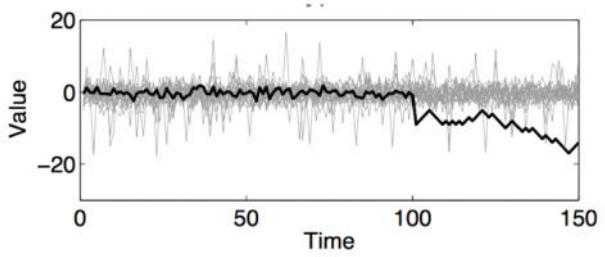
1010101	101	11	11	1
Content: 4	2	2	2	1
Capacity: 7	3	2	2	1

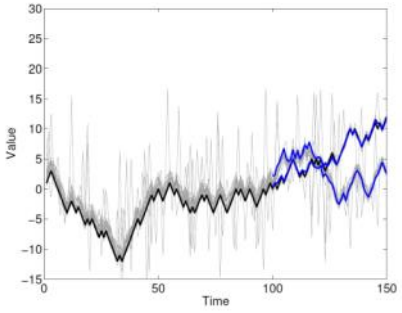
            - 3 > M = „Zwei Buckets mit Content=2" ⇒ compress

1010101	10111	11	1
Content: 4	4	2	1
Capacity: 7	5	2	1

            - Wenn Change entdeckt: Lösche das älteste Bucket.
        - TODO: Vorteil?
    - CUSUM
        - Ursprünglicher Sinn: Qualitätskontrolle. Alarm, wenn Produktionsprozess "out of control"
        - Beispiel: Milch in 1L Flaschen, Qualitätszahl = Füllmenge, Zielqualität = 1L. Berechne für jede Flasche Differenz zur Füllmenge, Summe der Differenzen = S, S=0 => under control, S=/=0 => out of control. Differenzen nach oben/unten sollen eigentlich dann als separate Effekte behandelt werden.
        - Algorithmus
            - **Algorithm 2:** CUSUM
              Data: Schwellwert *h*
              1 $S^+ \leftarrow 0$;
              2 $S^- \leftarrow 0$;
              3 **foreach** $t > 0$ **do**
              4 $S^+ \leftarrow max(S^+ + x_t, 0)$;
              5 $S^- \leftarrow min(S^- + x_t, 0)$;
              6 **if** $S^+ > h$ *or* $S^- < -h$ **then**
              7 output 'Alarm';
              8 $S^+ \leftarrow 0$;
              9 $S^- \leftarrow 0$;
              In etwa: Summe der Abweichungen nach oben
                - Bei Sollmittelwert von 0
                - Vgl. Beispiel auf F6.25
        - Summe der positiven Abweichungen und negative ignorieren würde Grundrauschen nur einseitig berücksichtigen
            - Daher Praxis:
                - $S_n^+ = \max(S_{n-1}^+ + x_n - k, 0)$
                - $S_n^- = \min(S_{n-1}^- + x_n + k, 0)$
                - k = Schwellenwert um kleine Abweichungen (Grundrauschen) herauszurechnen
        - Schwellenwert
            - Wähle Schwellenwert zum Alarm h so, dass ARL(delta) (average run length - durchschnittliche Dauer, bis Änderung delta detektiert wird) optimal. In der Praxis: zB h=5*sigma

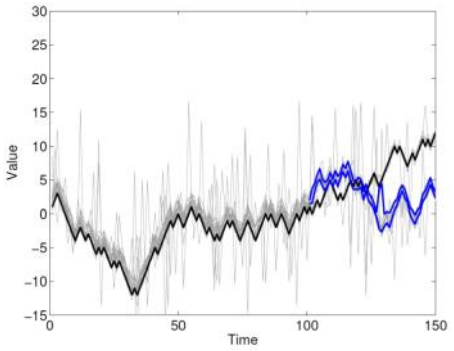
- CTC (Contextual Time Series Change)
    - Bisher: Change ist Eigenschaft einer Zeitreihe. Change kann aber vom Kontext abhängen, dh. von anderen Zeitreihen mit ähnlichem Verhalten. Unveränderheit kann Change sein!

- 

  - Graue Linien: Referenz/Kontext.
  - Zeitreihe =/= Datenstrom: Bei der Zeitreihe liegt der gesamte Datenbestand explizit vor.
- Algorithmus
  - 1. Kontext auf Referenzintervallen konstruieren. 2. Scoring auf neuem Zeitintervall.
  - Ergebnis: Change Score Matrix.
    - Spalten: Zeitpunkte
    - Zeilen: Einzelne Zeitreihen.
    - Wie groß ist der Context Change Score zu verschiedenen Zeitpunkten?
- Kontext konstruieren
  - Ziel: Konstruiere $G = \{X^1, \dots, X^m\}$, sodass $dist\left(X^j|_{TC}, O|_{TC}\right) < \varepsilon$. Dann gilt $\Pr[O|_{TC} \in G|_{TC}] \approx 1$.
  - O: Verhalten / Zeitreihe, die die Entwicklung eines Objektes über die Zeit aufzeichnet
  - Kontext/dynamic peer group G zu O: Zu O ähnliche Zeitreihen
  - Es existieren Techniken zumn Finden ähnlicher Zeitreihen in Intervall.
  - Problem: Outlier. (vgl. Abbildung F6.40, bei mehreren Outliern: Outlier oder verschiedene Zeitreihen?)
  - Lösung: Statistische Distanz k-ter Ordnung
    - Berechnung von $dist(O_T, X_T)$:
      - Berechne paarweise Distanz zwischen O_T und X_T zu jedem Zeitpunkt
      - Zeitpunkt absteigend nach Distanz sortieren
      - Entferne die ersten (höchsten) k-1 Zeitpunkt aus O_T und X_T
      - Auf restlichen Zeitpunkten Distanz wie gewohnt berechnen
    - k zu klein => Miss Rate zu groß. (Wir verpassen viele Elemente der Peer Group)
    - k zu groß => False Positives Rate (FPR) zu groß. (Zu viele falsche Elemente in Peer Group)
    - Wähle k, sodass False Positives Rate minimal ist (dh k möglichst klein)
      - Annahme: Wsk dass zu Zeitpunkt ein Outlier ist, gegeben.
      - Wsk berechenbar, dass Zeitreihe an min. k Stellen große Abweichung hat (Bernoulli-Experiment)
      - Dh. Miss Rate abhängig von k. Kleinstes k, mit dem Miss Rate noch unter Schwellenwert.
- Scoring
  - Motivation: Möglichst parameterfrei
  - Score: Maß für Abweichung von O zu G
  - Pro Zeitpunkt normalisierte Abweichung vom Durschnittswert der Peer Group (normalisiert = zB durch Standardabweichung teilen). Diese Abweichungen über alle Zeitpunkte von T_S aufsummieren.
  - 

  - Problem: Multimodal Behaviour. Wenige Zeitreihen der Peer Group entwickeln sich wie aktuell betrachtete target-Zeitreihe.
- Multimode Remover
  - Finde "Major Mode" (Verhalten der Mehrheit der Zeitreihen in G)
  - Algorithmus:
    foreach Zeitpunkt **do**
      repeat
        Entferne die ersten 10% der Zeitreihen aus *G*,
        die am weitesten vom Mittelpunkt von *G* entfernt sind
      until Mittelwert von *G* stabilisiert sich

    Berechne auf den Zeitreihen im Major Mode die Quantile.

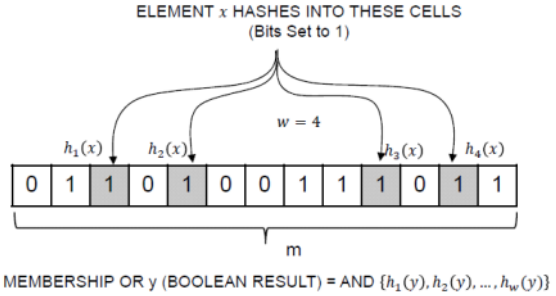
- 

- Abschluss
  - Change Detection:
    - Erkenne Änderungen in der Verteilung der Zielvariable.
  - Prinzip Windowing
    - ADWIN: Adaptiv gewählte statt apriori gewählte Fenstergröße.
    - Verbesserung: ADWIN2.
  - CUSUM: Erkenne Änderungen im Mittelwert durch Berechnung von kumulativen Summen.
  - CTC
    - Erkenne Änderungen in Zeitreihen auf Basis von ähnlichen Zeitreihen.

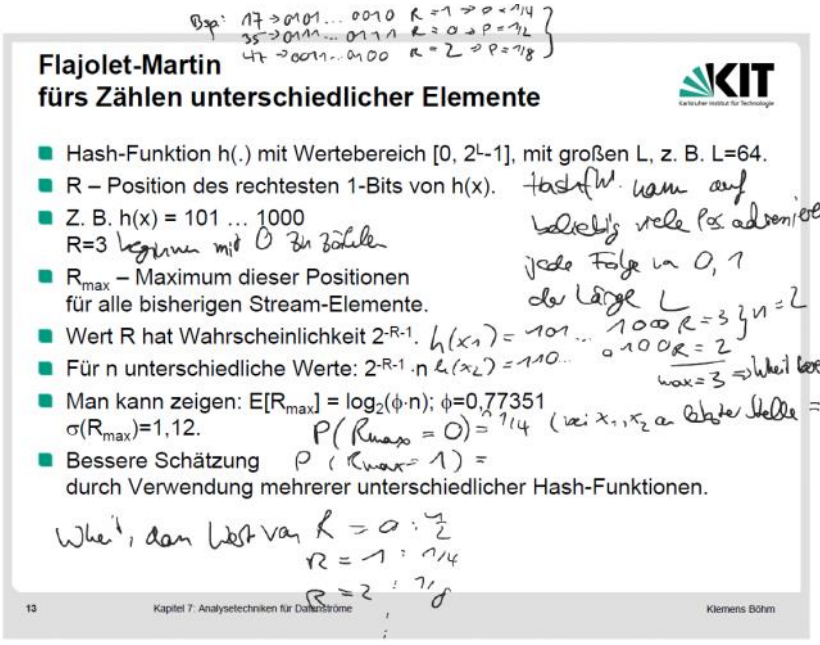
Verallgemeinerung konventioneller Datenanalysetechniken für Datenströme

- Einleitung
  - Online vs Offline Processing:
    - Online: "Echtzeitverarbeitung", Verarbeitung beim Eintreffen eines neuen Datenobjekts => zB Indexaktualisierung
    - Offline: Verarbeitungsschritte "on Demand only"
  - Fragen: Datenstrom muss effiziente Online- und Offline-Phasen ermöglichen. Wann auf Disk schreiben? Wie Analyseergebnisse aus Zusammenfassung ableiten?
  - Zusätzliche Dimension: Zeitlicher Bezug
- Synopsen
  - Generische Analysevorgehensweise: Stichprobe ziehen, gewöhnlichen Online-Algorithmus anwenden
    - Keine neue Implementierung notwendig, meist anwendbar, außer für "distinct-element counting"
  - Reservoir Sampling
    - Methode, um Sample eines Stroms zu erstellen und aktuell zu halten
    - n = #Datenobjekte, wächst kontinuierlich ohne Grenze
    - k = Größe Reservoir (fix => k/n wird kleiner)
    - Entscheidung bei neuem Objekt: Neues Objekt ins Sample übernehmen? Welche Objekte ggf verdrängen?
    - Lösung: n-tes Objekt mit Wsk k/n verdrängen. Zu verdrängendes Objekt zufällig auswähen.
    - Lemma: Nach Ankunft von n Datenobjekten ist jedes Objekt mit gleicher Wsk k/n im Reservoir => Objekte sind aus jedem Zeitpunkt unabhängig vom Alter gleich vertreten
  - Sampling bei Concept Drift
    - Bias Function => Ordne neueren Objekten höhere Wichtigkeit zu
      - Signatur f(r, n) (Wie wichtig ist Objekt?)
        - n = #Datenobjekte bis jetzt
        - r = wievieltes Objekt des Stroms
        - p(r,n) = Wsk, dass r-tes Objekt im Sample. p~f
        - Beispielfunktion: $f(r,n) = e^{-\lambda(n-r)}$
          - lambda=0 => Kein bias
      - TODO

  - Bloom-Filter
    - Set-Membership Query: Ist Wert jemals im Datenstrom vorgekommen?

- Filter ist Bit-Array der Länge m
- #w Hashfunktionen kodieren TODO JOHANNA?
- 

  - Idee: Filter speichert Info welche Daten vorkamen. Bei neuen Daten werden diese gehasht und Filter wird angepasst. Falls Flags aus dem Filter mit Hash übereinstimmen, wird das Objekt als bereits enthalten angenommen.
  - Kollision => False Positives
  - TODO Vorlesung
- Flajolet-Martin



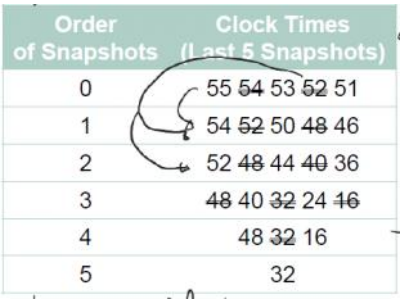
- Sketches
  - Kompakte Zusammenfassung von Datenströmen zur ungefähren Bestimmung der #Ereignissen im Datenstrom
  - Problem: Tabelle aller Ereignisse plus jeweils Anzahl passt nicht in RAM
  - Beispiel: Strom von E-Mails, Anzahl Mails von X an Y, für alle X und Y.
  - Count-min Methode
    - #w Hashfunktionen mit Wertebereichen [0, h-1]
    - Sketch = Array der Größe w x h => #h Zellen pro Hashfunktion
    - Verarbeitung eines Ereignisses: Hashwert mit jeder Hashfunktion berechnen. Eintrag in entsprechender Zeile inkrementieren (um 1 oder Ereignisgewicht).
    - Offline Berechnung
      - Wie viele Vorkommen eines Ereignisses bisher etwa?
      - Eintrag in entsprechender Zeile lesen. *Abschätzung ist Minimum dieser Einträge (Count min)*
  - Garantien
    - Kollisionen führen zu größeren Einträgen => eher über als unterschätzen. Minimum ist Abschätzung
    - Parameter δ (Wie oft darf Schätzung erheblich abweichen) und ε (Was bedeutet "unerheblich"?).
    - Mit Wsk=$(1-\delta)$ oder mehr ist Schätzung höchstens $tatsächlicheAnzahl + \varepsilon \cdot \#DatenobjekteimStromBisher$
    - Grundparameter: $w = \left\lceil \ln\left(\frac{1}{\delta}\right) \right\rceil, h = \frac{e}{\varepsilon} + 1$

- Clustering
  - STREAM
    - vgl. Facility Location/Costs von vorher: Statt vorgegebenem k wie kMedian finde facilityLocations = Centroid mithilfe von facility costs)
    - Zusammenhang k-Median zu facility location
      - z = facility costs
      - Je größer z => desto weniger Cluster
      - z=0 (zb keine Kosten, ein Lager an einem Standort zu haben) => Alle Objekte sind Cluster
      - z zu groß: Nur ein Cluster
      - => Jeder Wert von k kommt vor, wenn z von 0 bis unendlich variiert wird.

- Für jedes k gibt es ein z, sodass die Lösung für k-Median identisch ist mit Lösung für facility location.
- Angenommen, wir haben Algorithmus für facility location. K-Median mit binärer Suche über z lösen.
  - Gegeben
    - Menge N von Objekten, |N|=n
    - facility cost z
    - Metrik d(a,b)
  - gain(x)
    - Clusteringlösung ist gegeben, $x \in N$ ist kein Teil davon (=> x ist keine facility)
    - gain(x) : Wieviel würden wir sparen, wäre x eine facility? Evtl. andere schließen.
    - Dh. Einsparung durch "Reassignment"
  - Algorithmus
    - Initialisierung
      - N in zufällige Reihenfolge bringen
      - Erster Punkt ist facility
      - Für alle anderen Punkte in Reihenfolge:
        - d := Abstand Punkt zur nächsten facility
        - Mit Wsk d/z: mache Punkt zu neuer facility
    - LSEARCH
      - Initialisierung: Finden einer Lösung
      - Wiederhole $\Omega(\log n)$-mal
        - N in zufällige Reihenfolge bringen
        - Für jedes x, der Reihe nach: Wenn gain(x)>0, Lösung anpassen

- Algorithmus STREAM für Stream Clustering
  - Problemstellung: SSQ für Streams
  - Strom von *Menge von* Datenobjekten X1, …, Xn
  - Algorithmus
    - Für alle i
      - Clustering der i-ten Menge von Objekten X_i mit LSEARCH
      - Gewicht jeden Medians := Summe der Gewichte der Cluster-Elemente
    - RAM löschen, LSEARCH auf Centroide von X1, …, Xi anwenden
      - Also Clustering auf Menge von Centroiden (als Datenobjekte)
    - TODO vgl. Vorlesungsmitschnitt
  - Qualitätsverlust durch lokale Zuordnung

- BIRCH für Datenströme
  - BIRCH
    - Informationen zu jedem Cluster: Clustering Feature $CF = N, LS, SS)$
    - N - #Punkte im Cluster, $LS = \sum_i^N X_i \ \ (Linear\ Sum), SS = \sum_i^N (X_i)^2 \ \ (Square\ Sum)$
    - CF-Baum passt in RAM, höhenbalanciert, jeder Knoten=1 Cluster. Knoten A in B => Cluster A in B. Blatt = Menge von Clustering Features.
    - Parameter: B - Fan-Out, B' - Blattkapazität, T - Schwellenwert: (Durchmesser) eines Elementarclusters muss kleiner als T sein
    - BIRCH ist bereits einfaches Stream-Clustering Verfahren!
  - Übersicht
    - Objekte in bestimmten Zeitfenster in Vergangenheit *ab jetzt*
    - Gesucht ist Clustering des Datenbestandes mit festen #Clustern=k
    - Micro-Cluster: Verallgemeinerung von Cluster-Features für Datenströme
    - Pyramidal Time Frame: Speichervariante für Datenobjekte, zur systematischen Ausdünnung alter Objekte
  - Micro-Cluster
    - Micro-Cluster ist $(2 \cdot DatenDimensionalität + 3)$-Tupel $(\overrightarrow{CF2^x}, \overrightarrow{CF1^x}, CF2^t, CF1^t, n)$
      - n = Anzahl Datenobjekte
      - $\left(\overrightarrow{CF1^x}\right)_p = \sum_{j=1}^{n} \left(\left(Datenpunkt_j \right)^p \right)$
      - $\overrightarrow{CF2^x}$*:* Wie CF1, nur Quadratsumme
      - CF1^t: Summe der n Zeitstempel
      - CF2^t: Quadratsumme der n Zeitstempel
      - Notation: CFT(C) = Microcluster der Datenobjekte in C
    - Diskussion
      - Konzeptionell nichts neues. Merge zweier Cluster nur möglich durch Micro-Cluster.
      - Subtraktivität: $C_1, C_2\ Mengen\ von\ Datenobjekten; C_2 \subseteq C_1. \Rightarrow \overrightarrow{CFT}(C_1 - C_2) = CFT(C_1) - CFT(C_2)$
  - Pyramidale Speicherstruktur
    - Idee: Snapshots großzügig anlegen, ältere aber mit höherer Wsk löschen

- ◆ 

Order of Snapshots	Clock Times (Last 5 Snapshots)
0	55 ~~54~~ 53 ~~52~~ 51
1	54 ~~52~~ 50 ~~48~~ 46
2	52 ~~48~~ 44 ~~40~~ 36
3	~~48~~ 40 ~~32~~ 24 ~~16~~
4	48 ~~32~~ 16
5	32

  - ◆ Tabelle stellt gesamten Speicher dar. Einträge sind Zeitpunkte, zu denen Snapshots existieren. 55 ist aktuell (oben sind neuen Einträge).
  - ◆ Order: Abstände aufeinanderfolgender Snapshots pro Zeile. Im Beispiel: alpha=2
  - ◆ Tabelle wächst nur logarithmisch mit "Gesamtdauer der Vergangenheit)
  - ◆ Beweisbar: Fenstergröße h, es existiert 1+ Snapshot im Zeitraum $(t_{current} - 2h, t_{curret}]$
  - ◆ Dazu: Bias-Funktion: Wie wahrscheinlich ist es, dass wir Objekt speichern, abhängig vom Alter?
    - ◇ Monoton fallende Funktion.
- Online-Phase
  - □ Pro Zeitpunkt: Speicherung von #q Mikroclustern, jedes Mikrocluster bekommt ID zum Erstellungszeitpunkt. Beim Erstellen werden Mikrocluster ggf. gemerged.
  - □ Initialisierung: Statisches Verfahren (zB k-Means) angewendet auf Stromanfang zur Erzeugung von q Clustern
  - □ Entscheidung ob neues Objekt in neues Mcluster oder in bestehendes Mcluster:
    - ◆ Neues Objekt wird in bestehendes Cluster eingefügt, wenn es in seiner Maximum boundary liegt
    - ◆ Maximum Boundary factor eines Clusters: t-Standardabweichung der Abstände der Clusterobjekte vom Clustermittelpunkt
    - ◆ Falls Cluster nur ein Datenobjekt enthält: MaxBoundary = Abstand zum nächsten Cluster
    - ◆ Mergen oder Löschen bestehender Mikrocluster?
      - ◇ #Datenobjekte pro Cluster schlechtes Kriterium. Lieber: Durchschnittszeitpunkt der jüngsten m Objekte in Mikrocluster.
      - ◇ Wird mit Mikrocluster-Featues (erfordert normalverteilte Zeitstempel in MClustern) approximiert da sonst großer Speicheraufwand
      - ◇ Relevanzstempel eines Clusters = diese Approximation
      - ◇ Merge wenn alle Microcluster genügend aktuell (nähesten Cluster mergen)
- Offline-Phase
  - □ Exogene Parameter: Fenstergröße h und #gewünschteCluster k
  - □ S(t) = Micro-Cluster im Zeitraum/Zeitpunkt t
  - □ Gesucht: $S([t_{current} - h, t_{current}])$
  - □ $t_{current} - h'$: $kleinster\ Zeitpunkt, sodass\ Snapshots\ dazu\ existieren\ und\ h' \geq h, h' minimal$
  - □ Berechnung von $S([t_{current} - h, t_{current}])$
    - ◆ $Cluster \in S(t_{current})\ und\ Cluster\ \notin S(t_{current} - h') \Rightarrow$ $Alle\ Objekte\ im\ Cluster\ fallen\ in\ Zeitraum\ [t_{current} - h, t_{current}]$ (Fall: Rotes Objekt. Ist aktuell neu, daher wohl im Fenster dazugekommen)
    - ◆ $Cluster \in S(t_{current})\ und\ Cluster \in S(t_{current} - h') \Rightarrow nur\ neue\ Objekte\ interessant$ (Fall: Blaues Objekt)
    - ◆ Gesucht: IDs der grünen Objekte, die wohl gemergt worden sind.
      - ◇ ID-Liste durchgehen. Für jede ID:
        - ▸ MicroCluster t' mit dieser ID in $S(t_{current} - h')$ finden. Falls vorhanden, Cluster Features davon von dem von t abziehen.
        - ▸ IDs von t' aus der ID-Liste entfernen
    - ◆ 

    - ◆ *TODO: Folienmitschnitt nochmal schauen*
  - □ Eigentliches Clustering
    - ◆ Micro-Cluster sind Surrogate für Punkte. #Datenobjekte in Mcluster sind Gewicht des Punktes.
    - ◆ k-Means anwenden. Bei Sampling und Abstandsberechnung Gewicht berücksichtigen.
- Diskussion
  - □ Anfragen zu sehr vergangenen Daten haben weniger genaue Ergebnisse

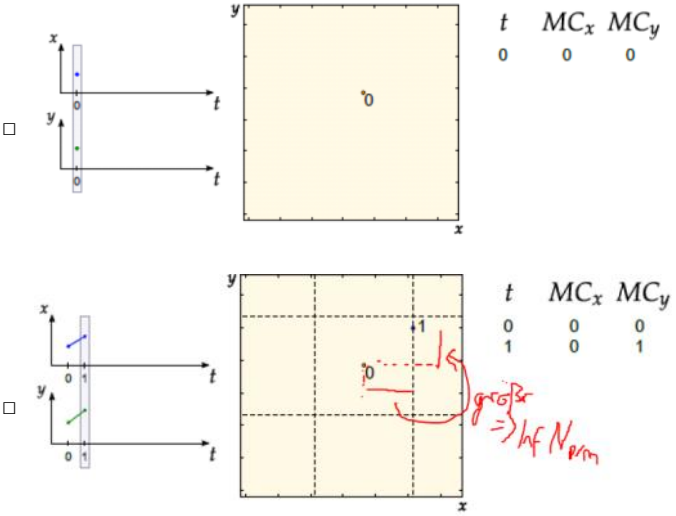
- Outlier
  - ○ Streaming Outlier Detection
    - Unterscheidung zwischen Outliern in Datenströmen/Outliern in Zeitreihen
      - □ Zeitreihe: Endlich, meist existieren Menge von Zeitreihen (bsp: Simulationsverläufe)
    - In Datenströmen: Zwei Arten von Outliern
      - □ Ungewöhnlicher einzelner Datensatz

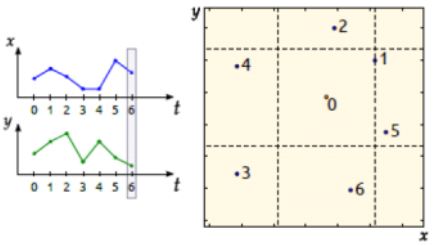
        - Definition berücksichtigt zB k-NN Distanz in laufendem Fenster der Länge w
    - Ungewöhnliche Aggregierung von Datenstätzen
        - "Aggregate Change Points als Outlier"
        - Kernel-Functions
            - Quantifizieren die Ähnlichkeit von zwei Objekten
        - Velocity Density
            - Wende Kernelfunktion auf Punkte aus Menge [Zeitpunkt-Fenster, Zeitpunkt] (F) und auf Punkte aus Menge [Zeitpunkt, Zeitpunkt+Fenster] (R) an.
            - Velocity Density = (F-R)/Fenster
            - Punkte mit ungewöhnlich hohen absoluten Werten der Velocity Density sind Outlier.
- Frequent Pattern Mining
    - Verwende Synopsen (Reservoir Sampling)
        - Verwalte Reservoir Samples und wende darauf Frequent Pattern Mining Algorithmus an
        - Wsk für False Positives lässt sich mit Chernoff Ungleichung herleiten
    - Verlustbehaftetes Zählen
        - Schwieriges Zählen bei riesiger Grundmenge. "Lossy Counting".
            - zB: 5000 Items => 5000^2 2-Itemsets, …
        - Strom in gleichgroße Segmente aufteilen, wenn man Grenze eines Segments erreicht, "vergisst" man Items, die nicht frequent sind.
        - Decremental Trick: Bei Segmentgrenze Zähler--, dann Pruning auf Items mit Frequency=0, dann großzügigere Korrektur… TODO

- Klassifikation
    - Existieren Ansätze, meist Fokus auf Langzeitverhalten
    - …

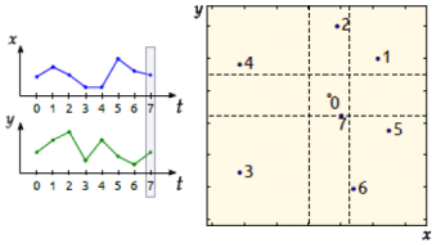
MISE: Korrelationsabschätzung für Datenströme

- Erinnerung Mutual Information
    - The reduction of uncertainty on one random variable X given knowledge of another variable Y.
    - I(X,Y)=H(X)+H(Y)-H(X,Y) (I = Mutual Information, H=Entropy)
    - Entropy Estimator: nearest-neighbour information
    - Approximation ist notwendig um nicht den ganzen Datenstream zu speichern
    - Queries on Multiple Time Scales
        - Letzte Minute? Letzte Stunde? Vorletztes Jahr? Wie dieselbe Qualität über beliebige Zeitskalen erhalten?
- Introduction
    - Grundlagen von MI Estimation
        - Kraskov Prinzip ist der führende Estimator auf statistischen Daten. Verlangt Dynamik von NN-Beziehungen zu handeln. Nutze Kraskov in unserem Anwendungsfall.
        - NN vs NN-Distanz: Verwende "Infinity Norm": Verwende Abstand in der Dimension, in der er am größten ist.
        - Marginal Count, Marginal Slice



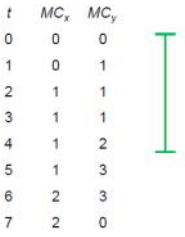
- 

t	MC_x	MC_y
0	0	0
1	0	1
2	1	1
3	1	1
4	1	2
5	1	3
6	2	3

- 

t	MC_x	MC_y
0	0	0
1	0	1
2	1	1
3	1	1
4	1	2
5	1	3
6	2	3
7	2	0

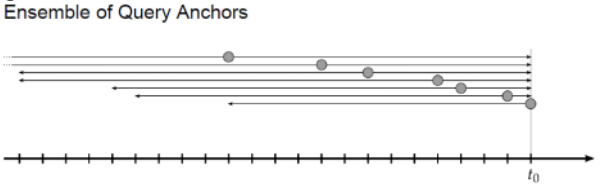
  - Marginal Slice wird zum NN Nachbarn entsprechend Infinity Norm gesetzt (Kastenradius in x- und y-Richtung = längste Dimensionsdistanz zum NN Nachbarn)
  - MC=Marginal Count = Anzahl Elemente in derselben Slice (wie hier die von Objekt 0)

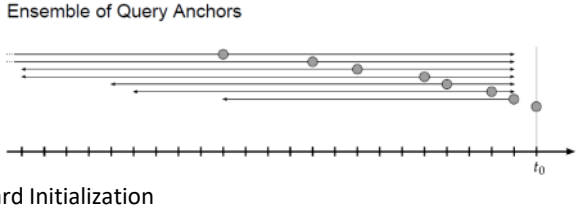
    Why is this data important?
- 

t	MC_x	MC_y
0	0	0
1	0	1
2	1	1
3	1	1
4	1	2
5	1	3
6	2	3
7	2	0

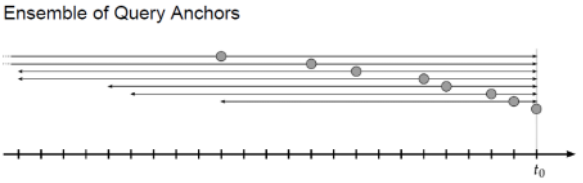
    Adapted Kraskov principle: Digamma function

$$MI_{est} = \psi(N) + \psi(k) - \psi(MC_x + 1) - \psi(MC_y + 1)$$

    For subsequence Q_0^4: $MI_{est} = \psi(5) + \psi(1) - \psi(2) - \psi(3)$
    - k=1 (kNN)
  - Query Anchor
    - Datenstruktur, um Marginal Counts zu zählen
      - Korrespondiert zu spezifischem Zeitpunkt t
      - Speichert (X,Y) Daten von Punkte
      - InsertRight(Q_t'): Fügt einen Datenpunkt Q_t' rechts auf der Zeitachse ein (t'>t)
      - InsertLeft(Q_t'): Fügt einen Datenpunkt Q_t' links auf der Zeitachse ein (t'<t)
      - Query(t1, t2) mit t1<=t<=t2: Liefert Marginal Counts $MC_X\left(Q_t, Q_{t_2}^{t_1}\right), MC_y\left(Q_t, Q_{t_2}^{t_1}\right)$
    - Implementierung
      - Dynamischer Array: Random Access, anpassbare Größe
      - Pro Arrayelement: Marginal Points, k-NN Set
        - Wenn Satz des k-NN geändert wird, wird neues Set gespeichert/angehängt
- MISE (Mutual Information Stream Estimator)
  - Framework
    - Ensemble von Query Anchors
      - Ausgang
        - 

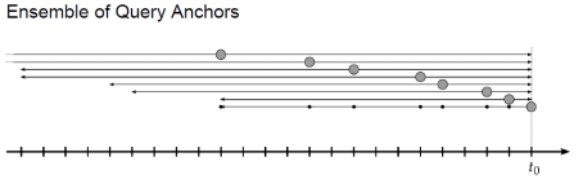
      - Neuen Query Anchor dazufügen
        - 

      - Forward Initialization



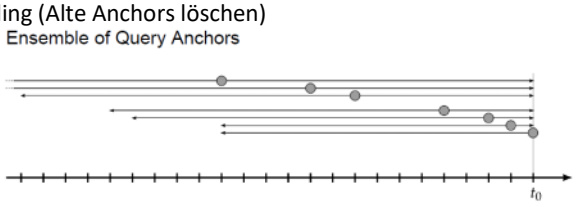
- Mit Insert Right in bestehende Anchors eingefügt

- Reverse Initialization

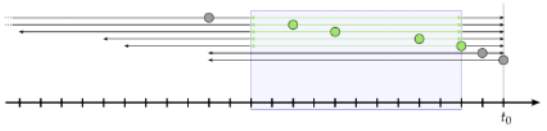


- Mit Insert Left werden bestehende Anchors in neuen eingefügt.

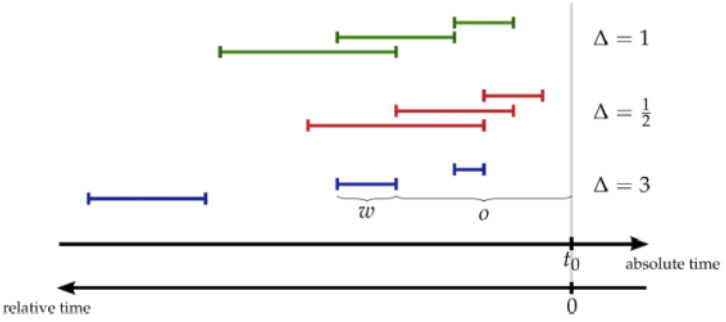
- Sampling (Alte Anchors löschen)



- Query



- Finde relevante Anchors, delgiere Querie dorthin und aggregiere Resultat.

- Multiscale Sampling
  - Fragestellung bei Sampling: welche Objekte (Query Anchors) werden gelöscht, für welche Objekte werden neue Anchors erstellt?
  - Definiere Quantität ohne Einheit: $\Delta = \frac{Abstand}{Fenster} = \frac{o}{w}$
  - 

    - Multiscale Query Equivalence: Zwei Queries sind …-äquivalent, wenn ihr Δ übereinstimmt.
    - In äquivalenten Queries wollen wir dieselbe Qualität
      - Vgl. Wunsch von vorhin: Qualitätserhaltung für Maße "letzte Stunde" vs "letztes Jahr"
      - Bemerke: Je weiter Fensterblock von t0 entfernt ist, desto größer ist das Fenster!
      - Je kleiner Delta => Desto größer Schätzqualität
    - Anzahl der Query Anchors (Sampling elements) ist in allen Queries derselben Äquivalenzklasse gleich groß
      - => In kleineren Queries (aktuellere Queries) sind mehr Querie Anchors => Exakter
- Herleitung der Bias Funktion (wann löschen? oder: Wie wahrscheinlich, dass Query Anchor Zeitabschnitt gut repräsentiert?)
  - f(t) - Dichtefunktion von Query Anchors
    - f(t) = c <=>: Es gibt einen Query Anchor in der Eps-Umgebung von t mit Wsk c.
    - f(t) = 1 => Bei Zeitpunkt t existiert ein Query Anchor.
  - Erwartete #QueryAnchors im Zeitintervall $[o, o+w]$:
    - $\int_o^{o+w} f(t)dt \cdot \#QueryAnchorsInsgesamt = \int_{w\cdot\Delta}^{w\cdot(\Delta+1)} f(t)dt \cdot \#QueryAnchorsInsgesamt$
    - Der Wert muss für alle w=Fenstergrößen gleich sein, also konstant.
    - Im kontinuierlichen Fall erfüllt $f(t) = \frac{C}{t}$ das. (C=Konstante?TODO)
    - Diskreter Fall
      - Punkte werden durchnummeriert.
  - Jetzt: Wann löschen?
    - Sampling Function
      - $SP_n = \begin{cases} 1 & if\ n \leq \alpha \\ \frac{P_n}{P_{n-1}} & otherwise \end{cases}$

        - SP = Stepwise Sampling Probability
    - TODO F8.65

- Experiments
    - Stream processing worthwhile even when queries are rare
    - Can high-frequency-Streams be processed?: Frequencies possible up to 100Hz (high estimation quality) or 20kHz (low estimation quality)
    - How does sampling influence estimation quality? Very low estimation error. Experiment: alpha=1000, Delta=1, MI Präzision +-0.01
    - What are the benefits of multiscale sampling? Traditional sampling does not work with queries comprising multiple time scales.
- Conclusion
    - Ausblick
        - Query Anchors enthalten keine Löschen Funktion, daher nicht maximal effizient.Es existieren bessere Varianten.
        - Zu verbessernde Schätzqualität.
        - Tradeoff Performance vs Qualität. Was ist, wenn wir mit groben MI-Werten zufrieden sind?

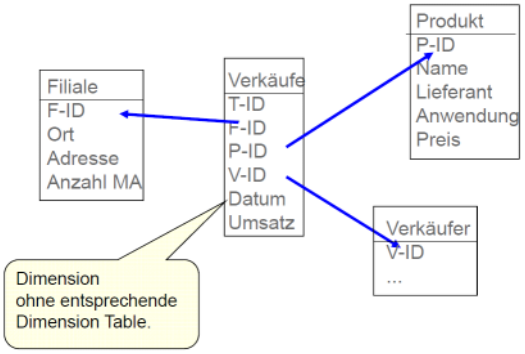
Vom relationalen zum multidimensionalen Datenmodell

- Erinnerung Aggregatsfunktionen aus AGB1
    - distributiv, algebraisch, holistisch
    - self-maintainable: Neuer Aggregatswert wird bei geänderten Daten aus Delta hergeleitet
- Dimensionen vs Measures
    - Klassifizierung der Attribute einer Relation in Dimensionen und Measures
        - Dimension: Attribute, anhand derer man Tupel klassifizieren identifizieren/klassifizieren kann
        - Measures: Wert, der einem Tupel zugeordnet ist
        - Dimension oft kategorisch, Measure eher kontinuierlich. Muss aber nicht!
        - Measures flache Hierarchieren, Dimensionen oft nicht.
    - Beispiel
        - Dimensionen: Ort, Hypothek, Alter
        - Measure: Kontostand
        - Was, wenn mehrere Personen am gleichen Ort gleichen Alters gleiche Hypotheken haben?
            - Mehrere Kontostandwerte an derselben Position (als Menge)
            - Aggregierter Wert
    - Trennung: strukturelle und inhaltliche Aspekte
        - Umsatz kann Funktion von Gewinn sein (Umsatz ist Measure), oder Gewinn als Funktion von Umsatz (Umsatz ist Dimension).
    - Mehrdimensionale Tabellen
        - Relation R(Part, City, Year, Month, Cost Sale) mit Tabellenschema Sales=<{Category, Time}, {Part, City, Year, Month, Cost, Sale}, par>, wobei par(Category)={Part, City} und par(Time)={Year, Month}
        -

SALES			TIME						
			Year	1996			1997		
			Month	Jan	Feb	…	Jan	Feb	…
CATEGORY	Part	City	(Cost, Sale)						
	PC	Montreal		(5,6)	(5,7)	…	(4,6)	(4,8)	…
		Toronto		(5,7)	(5,8)	…	(4,8)	(4,9)	…
		…		…	…		…	…	
	Inkjet	Montreal		(7,8)	(7,9)		(6,9)	(6,8)	…
		New York		(6,9)	(6,9)		(5,8)	(5,9)	…
		…		…	…		…	…	
	…	…		…	…		…	…	

        - Sales[0]={Category, Time} sind Dimensionen, Sales[1]={Part, City, …} sind Attribute. par identifiziert die Bestandteile der Dimensionen.
        - Im Beispiel wurde "zufällig" die richtige Darstellung gewählt, nämlich zB dass Year > Month
        - Anderes Beispiel: Gleiche Relation, anderes Tabellenschema
        -

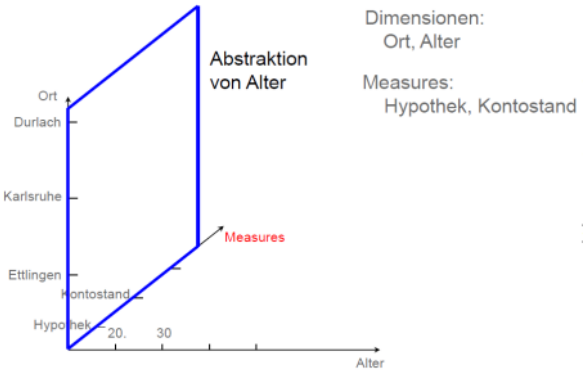
SALES		COMPONENT			
		Part	PC	Inkjet	…
LOCA-TION	City	(Year,Month, Cost,Sale)			
	Montreal		(1996, Jan, 5, 6)	(1996, Jan, 5, 6)	…
	Montreal		(1996, Jan, 5, 7)	(1996, Jan, 5, 7)	…
	…		…	…	
	Montreal		(1997, Jan, 4, 6)	(1997, Jan, 4, 6)	…
	Montreal		(1997, Feb, 4, 8)	(1997, Feb, 4, 8)	…
	…		…	…	
	Toronto		(1997, Jan, 5,7)	/	…
	Toronto		(1996, Feb, 5, 8)	/	…
	…		…	…	

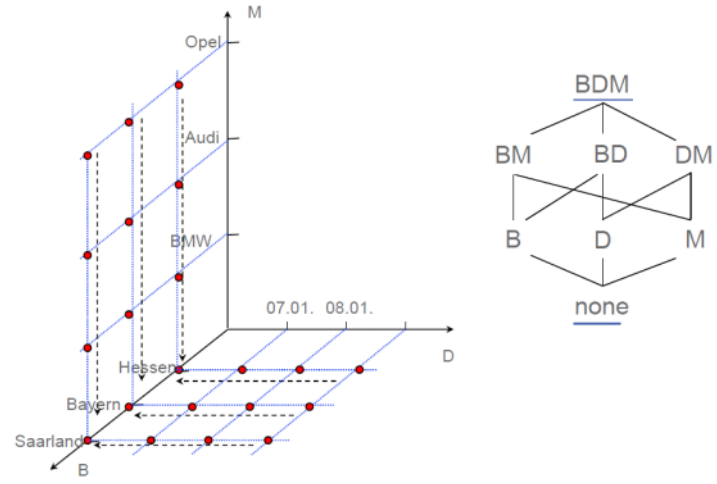
        - Sales=<{Location, Component}, {Part, City, Year, Month, Cost, Sale}, par> mit par(Location)={City}, par(Compoennt)={Part}
    - Einfügen von Null-Werten in Tabelle für Zellen, deren Zeilen/Spalten existieren (zB. Karlsruhe, 1996, April; existieren alle, aber nicht in der Kombination), die aber keine Belegung haben. Voraussetzung für das problemlose Wechseln zwischen Modellen.
  - Warum Differenzierung Measures vs Dimensionen wichtig?
    - Fest einstellbar bei den meisten Werkzeugen. Operationen auf Data Cubes unterschieden dazwischen.
- Schema-Typen
  - Star Schema
    - Fact Table: Enthalten Dimensionen und Measures
    - Dimension Table: Beschreiben die Dimensionen
    - 

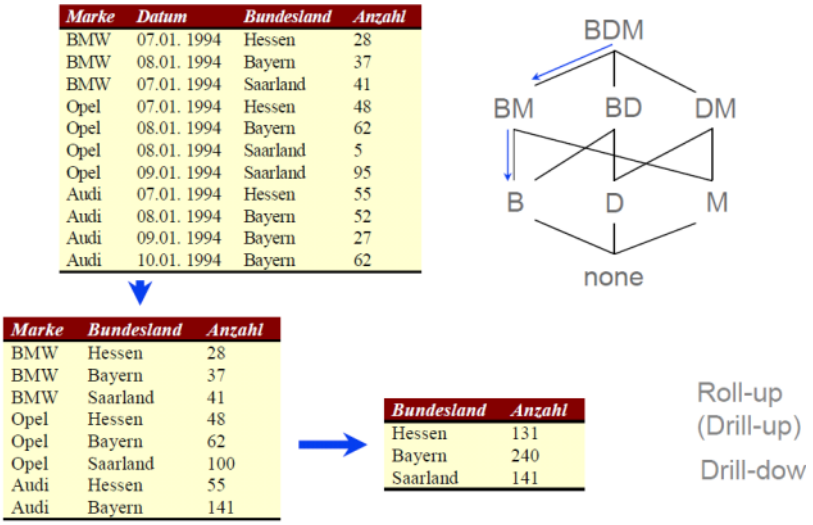
    - Mittlere Tabelle: Fact Table. Enthält statt Produktdaten als Dimension nur Produkt-ID.
    - Tabellen außen rum: Dimension Table. Enthalten konkretere Dimensionsdaten über zB Produkt, gegeben seiner ID.
  - Snowflake Schema
    - Verfeinerung Star Schema, Attribute der Dimension Tables werden erneut durch weitere Relationen ("Sub-Dimensiontables") beschrieben.
    - Keine Zyklen erlaubt.
- GroupBy & Data Cube
  - GroupBy: Unterteilt Tabelle in Gruppen, auf jede Gruppe wird Aggregatsfunktion angewendet.
  - Parameter von GroupBy: Gruppierungsattribute, Aggregatsfunktion
  - Mehrdimensionales GroupBy: Gruppieren nach mehr als einem Attribut. Vgl Kreuzprodukt (Geht mit GroupBy eigentlich nicht?)
Marke	*Bundesland*	*Anzahl*
BMW	Hessen	28
BMW	Bayern	37
BMW	Saarland	41
Opel	Hessen	48
Opel	Bayern	62
Opel	Saarland	100
Audi	Hessen	55
Audi	Bayern	141
    - Bessere Darstellung mit CrossTable (im Fall von genau zweidimensionalen Aggregationen)
T-Shirt Verkäufe			
T-Shirt	**1996**	**1997**	**total** (ALL)
rot	500	450	950
blau	300	400	700
total (ALL)	800	850	1650
- Cube-Operator
  - n-dimensionale Verallgemeinerung von GroupBy. Berechnen n Hyperebenen statt nur einer.
Marke	Datum	Bundesland	**Anzahl**
BMW	07.01. 1994	Hessen	28
BMW	08.01. 1994	Bayern	37
BMW	07.01. 1994	Saarland	41
Opel	07.01. 1994	Hessen	48
Opel	08.01. 1994	Bayern	62
Opel	08.01. 1994	Saarland	5
Opel	09.01. 1994	Saarland	95
Audi	07.01. 1994	Hessen	55
Audi	08.01. 1994	Bayern	52
Audi	09.01. 1994	Bayern	27
Audi	10.01. 1994	Bayern	62
...			
BMW	07.01. 1994	ALL	69
BMW	08.01. 1994	ALL	37
Opel	07.01. 1994	ALL	48
Opel	08.01. 1994	ALL	67
Audi	07.01. 1994	ALL	55
Audi	08.01. 1994	ALL	52
Audi	09.01. 1994	ALL	27
Audi	10.01. 1994	ALL	62
BMW	ALL	Hessen	28
BMW	ALL	Bayern	37
BMW	ALL	Saarland	41
Opel	ALL	Hessen	48
...			
BMW	ALL	ALL	106
Opel	ALL	ALL	210
Audi	ALL	ALL	196
ALL	ALL	Bayern	240
ALL	ALL	Hessen	131
...			
ALL	ALL	ALL	512

    Data Cube

    (Group-By Operator berechnet nur einzelne Zeilen dieser Relation.)

    - Standard-SQL dafür ungeeignet (für "ALL" müsste jedes mal eine extra Relation gejoined werden)
- Range Queries
    - Aggregation über Elemente eines Hypercube in einem Intervall.
    - Intervall für numerische Attribute, bei kategorischen über einzelne Werte oder über ganzen Wertebereich.
    - Vgl. Abbildung auf F9.45
- Berechnung des Data Cubes
    - 

    - 

    - Man fängt zB mit B (Bundesland) Tabelle an. Dann Konstruiert man BM (=Bundesland x Marke), BD, …
    - Reduktionsfaktor: $\frac{Kardinalität\ (Input-Relation)}{Kardinalität\ (Output-Relation)}$
        - zB: Input-Relation hat 3 Dimensionen, Output nur 1 (zwei wurden "cubed by"), dann Reduktionsfaktor = 3/1 = 3
    - Parent: A ist Parent von B (A>B), wenn B aus A berechnet werden kann und A *genau* ein Attribut mehr hat als B.
- Operatoren für DataCube
    - CubeBy nicht so interessant für Benutzer, aber für benutzerfreundliche Oepratoren
    - Consolidation Paths
        - 

Marke	Datum	Bundesland	Anzahl
BMW	07.01. 1994	Hessen	28
BMW	08.01. 1994	Bayern	37
BMW	07.01. 1994	Saarland	41
Opel	07.01. 1994	Hessen	48
Opel	08.01. 1994	Bayern	62
Opel	08.01. 1994	Saarland	5
Opel	09.01. 1994	Saarland	95
Audi	07.01. 1994	Hessen	55
Audi	08.01. 1994	Bayern	52
Audi	09.01. 1994	Bayern	27
Audi	10.01. 1994	Bayern	62

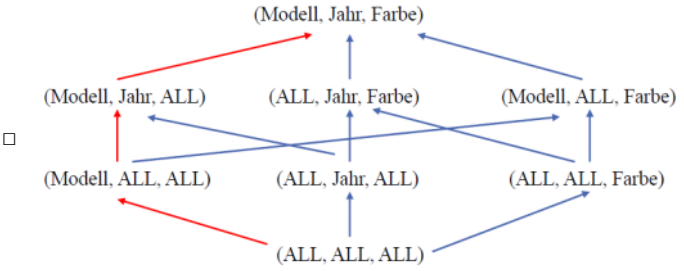
Marke	Bundesland	Anzahl
BMW	Hessen	28
BMW	Bayern	37
BMW	Saarland	41
Opel	Hessen	48
Opel	Bayern	62
Opel	Saarland	100
Audi	Hessen	55
Audi	Bayern	141

Bundesland	Anzahl
Hessen	131
Bayern	240
Saarland	141

Roll-up
(Drill-up)
Drill-dow

        - Schrittweise Vergröberung (Drill-up/down)
        - Konsolidierung nicht nur durch Ausblenden von Dimensionen, sondern auch Zusammenfassen von Werten
Typ-Perspektive:	Store -> Store Country -> Store State -> Store City -> Store Name
Instanzen-Perspektive:	. -> Canada -> CA -> …

        - Dimension Tables ermöglichen diese "Schrittweise Vergröberung": Verkäufe nach Lieferanten zusammenfassen, …
    - Symmetrische vs Asymmetrische Aggregationen
        - Symmetrische Aggregationen
            - Berechne Resultate aller Knoten (CUBE)
        - Asymmetrische (lineare) Aggregationen
            - Berechne nur Resultate der Knoten entlang eines Pfades (Rollup)



    - Slice & Dice
        - Nur 2D-Tabellen sind auf Bildschirm darstellbar.
        - Unterschiedliche Sichten sinnvoll, daher freier Zugriff auf Ausschnitte des Cubes
        - Dice: Drehen des Würfels. Slice: Aufschneiden des Würfels
    - Verknüpfung von Werten aus unterschiedlichen Dimensionen: ? (TODO, Folie nicht verstanden)
    - Weitergehende Ansätze
        - OLAP over Uncertain and Imprecise Data (OLAP=Online Analytical Processing)
            - Wsk-Verteilungen statt Measure-Werten. I.Allg. hat man keine konkreten Werte, sondern gröbere Daten.
        - Graph OLAP
            - Abstraktionsmöglichkeiten für Kanten (zB Datum der Interaktion) und für Knoten (zB Knoten nach Ort zusammenfassen. Alle Karlsruher werden zum Knoten "KA", dann zum Knoten "BaWü")
            - Ähnlich zu drill-up/down
            - Elementare Zellen enthalten statt measure-Werten (Kennzahlen von) ganze Graphen
    - Algebra für multidimensionale Strukturen
        - Algebra, deren Operatoren den DataCube-Operatoren entsprechen
        1. Beschreibung der Abbildung eines Datenbestandes von einer Darstellung in andere
        2. Definition der Operatoren mithilfe des relationalen Modells
        - Selektion, Projektion, Umbenennung
        - unfold, fold
            - unfold macht aus einem/mehreren Measures eine Dimension, fold macht das Gegenteil.
- Operatoren in MS-SQL
    - Ziel: Bisherigen Ideen in SQL einbetten
    - Cube Operation
        - Andere Syntax: NULL statt ALL
        - GROUPING=True <=> Element ist ein ALL Wert

```
Select Modell, Jahr, Farbe, SUM(Stücke),
    GROUPING(Modell),GROUPING(Jahr),GROUPING(Farbe)
FROM Verkauf
WHERE Modell = 'T-Shirt'
    AND Jahr BETWEEN 1996 AND 1997
GROUP BY Modell, Jahr, Farbe WITH CUBE;
```

        - "original"

Modell	Jahr	Farbe	Stücke
T-Shirt	1996	rot	500
T-Shirt	1996	blau	300
T-Shirt	1997	rot	450
T-Shirt	1997	blau	400
T-Shirt	'ALL'	rot	950
T-Shirt	'ALL'	blau	700
T-Shirt	1996	'ALL'	800
T-Shirt	1997	'ALL'	850
T-Shirt	'ALL'	'ALL'	1650

        - Version MS-SQL-Server

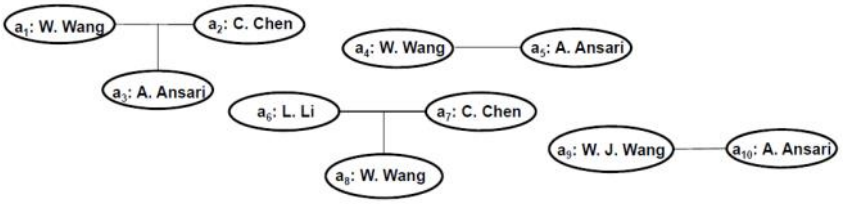
Modell	Jahr	Farbe	Stücke	Grouping (Modell)	Grouping (Jahr)	Grouping (Farbe)
T-Shirt	1996	rot	500	FALSE	FALSE	FALSE
T-Shirt	1996	blau	300	FALSE	FALSE	FALSE
T-Shirt	1997	rot	450	FALSE	FALSE	FALSE
T-Shirt	1997	blau	400	FALSE	FALSE	FALSE
T-Shirt	NULL	rot	950	FALSE	TRUE	FALSE
T-Shirt	NULL	blau	700	FALSE	TRUE	FALSE
T-Shirt	1996	NULL	800	FALSE	FALSE	TRUE
T-Shirt	1997	NULL	850	FALSE	FALSE	TRUE
T-Shirt	NULL	NULL	1650	FALSE	TRUE	TRUE

    - RollUp: "WITH ROLLUP"
    - Drill-Down
    - Herkömmliche Datenbankmethoden reichen im Prinzip, aber neuer Operationen bieten höheres Abstraktionsniveau
        - Zu umständlich mit normalen Operatoren
        - Übersichtlichkeit bei großen komplexen Anwendungen
        - Bessere Performance durch effiziente Operatorimplementierung
        - Anfrageoptimierung
- ROLAP vs MOLAP
    - OLTP vs OLAP
        - OLTP: Online transaction processing
            - Transaktionsorientierte Datenzugriffe. Erfassung von Daten und Lesezugriffe.
            - "Tagesgeschäft bedienen"

        - OLAP: Online analytical processing
            - Konsolidierung, Viewing und Analyse der Daten gemäß mehrerer Dimensionen
            - "Entscheidungen unterstützen"
    - ROLAP vs MOLAP
        - ROLAP: Relational OLAP
            - Aggregate, zB Resultat vom CUBE wird in relationaler Datenbank gespeichert
        - MOLAP: Multidimensional OLAP
            - spezielle Speicherstruktur für Inhalte multidimensionaler Datenbank
            - Schlechte Speichernutzung bei sparse Data. Abhilfe: Matrixkomprimierung

Data Cleaning und Entity Matching

- Einleitung
    - Preprocessing: Daten aufzubereiten vor Analyse
    - *Data Cleaning, Data Integration*, Data Reduction
    - Probleme von Daten: unvollständig, noisy (unkorrekte Attributwerte), inkonsistent (Sr. und Senior)
    - Heterogenität
        - Gleicher Sachverhalt verschieden bezeichnet
            - Schemaebene: customer-id vs cust-id
            - Instanzebene: Steffi vs Stephanie
        - Unterschiedlicher Sachverhalten gleich bezeichnet
            - Übernachtungspreis mit/ohne Frühstück
        - => Data Integration
    - Duplikateliminierung
        - Zählen identischer Wörter reicht nicht
        - Duplikaterkennung vs Entity Matching
            - Entity Matching ist übergeordneter Begriff: Datenobjekte können aus versch. Relationen kommen
            - Duplikaterkennung: Nur Datenobjekte aus demselben Datenbestand
        - Entity Matching vs String Matching
            - Attribute konkatenieren und matchen? Domänenwissen nicht anwendbar!
        - Schwierigkeiten
            - Versch. Formatierungskonventionen/Benamungskonventionen
            - Abkürzungen/"Verstümmelungen"
            - Auslassungen
            - Spitznamen
            - Fehler im Datenbestand
            - Skalierbarkeit für große Datenbestände
        - Imputation: Zu Data Cleaning gehört auch "Schätzen fehlender Werte"!
            - Einfach konventionelle Data-Mining Techniken anwenden (Klassifikation)
- Regelbasiertes Matching
    - Entwickler formuliert händisch Regeln "Wann matchen zwei Tupel?"
    - Möglichkeiten
        - Lineare Summe von einzelnen Ähnlichkeitswerten
            - $\mathrm{sim}(x,y) = \sum_{i=1}^{n} \alpha_i \cdot \mathrm{sim}_i(x,y)$, gewichtete Summe von n Ähnlichkeitswerten der n Attribute
            - x und y matchen wenn Summe größer als Schwelle.
            - Beispiel
                - $sim(x,y) = 0.3 \cdot sim_{name}(x,y) + 0.3 \cdot sim_{phone}(x,y) + 0.1 \cdot sim_{city}(x,y) + 0.3 \cdot sim_{state}(x,y)$
                - $sim_{name}(x,y)$: basiert auf Jaro-Winkler.
                - $sim_{phone}(x,y)$: basiert auf Editierdistanz zwischen Telefonnummer von x ohne Vorwahl und der von y.
                - $sim_{city}(x,y)$: basiert auf Editierdistanz.
                - $sim_{state}(x,y)$: basiert auf exaktem Match; ja $\rightarrow$ 1, nein $\rightarrow$ 0.
            - Vorteile
                - Konzeptionell einfach
                - leicht zu implementieren
            - Nachteile
                - Erhöhung einer Summenkomponente vergrößert Summe gleich um Erhöhung*Gewicht.
                - Wenn Komponente bereits groß ist, sollte Erhöhung davon nicht so sehr ins Gewicht fallen.
        - Kombination auf logistischer Regression
            - Grenznutzen am Rand soll abnehmen. Nutze Regressionskurve
            - $sim(x,y) = \frac{1}{1+e^{-z}}, z = \sum_{i=1}^{n} \alpha_i \cdot sim_i(x,y)$
            - Gewichte sind nicht mehr auf [0,1] beschränkt und müssen in Summe nicht 1 ergeben.
        - Komplexere Regeln
            - Entscheidungsbaum-artige Regeln für komplexe Sachverhalte
            - Vorteile

        - Einfach aufzusetzen, konzeptionell einfach
        - Schnelle Laufzeit
        - Komplexe Zusammenhänge können erfasst werden
    - Nachteile
        - Arbeitsaufwendig, Zeitaufwendig gute Regeln zu formulieren
        - Schwierig, passende Gewichte zu finden
        - Struktur der Regeln evtl unklar
        - => Lernbasierte Verfahren
- Lernende Matching Verfahren
    - Supervised Learning (Klassifikation)
    - Matching Modell mit Trainingsdaten trainieren, dann auf neue Daten anwenden
    - Lerne die Regel sim(x,y) aus "Lineare Summe von einzelnen Ähnlichkeitswerten"
        - Quadrierten Fehler minimieren (TODO Formel, F10.35)
    - Oder: Entscheidungsbaum lernen
    - Vorteile
        - Regelbasiertes Vorgehen kann aufwendig sein, da per Hand
        - Lernbasierte Verfahren können sehr komplexe Regeln konstruieren
    - Nachteile
        - Großer Datenbestand notwendig (schwer zu erlangen)
        - Clustering hilfreich?
- Matching mithilfe von Clustering
    - Viele Clustering-Techniken nutzbar, zB k-Means oder (hier verwendet) agglomeratives hierarchisches Clustering
    - Berechnung des Ähnlichkeitswerts zwischen zwei Clustern:
        - Kanonischer Tupel: Repräsant jedes Clusters
        - Ähnlichkeit zweier Cluster ist Ähnlichkeit ihrer kanonischer Tupel
        - Auswahl des kanonischen Tupels: zB zufälliges Sample, n-letzten Werte, Mittelwert, syntaktisch, Benutzereingabe (je nach Fall)
    - Idee
        - Definiertes Problem: Konstruiere Entitäten (Cluster)
        - Iterativer Prozess
        - Merge matchende Tupel in Cluster, um sogenanntes "Entity Profile" zu konstruieren (s. später)
- Probabilistische Matching-Verfahren
    - Unwahrscheinlich, dass zwei Attributwerte in Kombination auftreten (Doktor + Türsteher) (Alle in Karlsruhe => Wohnort=KA nicht hilfreich)
    - Modellierung der Domäne mit Wsk-Verteilung
    - Vorteile
        - Theoretisch fundiertes Rahmenwerk, natürliche Berücksichtigung von Domänenwissen
        - Viele Techniken für Wsks können genutzt werden
        - Referenzpunkt für Benchmarking von anderen Ansätzen
    - Nachteile
        - Aufwendig zu berechnen
        - Schwer zu verstehen/debuggen
- Collective Matching
    - mit Clustering
        - Matching Entscheidungen sind oft korreliert. Nutze diese Korrelationen für bessere Matchings
        - 

            - Wenn a1 und a4 gematched sind, ist Matching von a3 und a5 wahrscheinlicher. Anderstrum aber genauso!
                - "Äquivalenz", also in beide Richtungen muss gelten. Sonst Problem: Wenn a1 und a4 kein Matching sind, aber gematched werden, ist Matching für a3 und a5 unangenessen hoch!
        - $\boldsymbol{sim}(\boldsymbol{A},\boldsymbol{B}) = \alpha \cdot \boldsymbol{sim}_{attributes}(\boldsymbol{A},\boldsymbol{B}) + (1-\alpha) \cdot \boldsymbol{sim}_{neighbors}(\boldsymbol{A},\boldsymbol{B})$
            - sim_attributes berücksichtigt nur (wie gewohnt) Attributwerte
            - sim_neighbors berücksichtigt strukturelle Beziehungen. Ändert sich im Laufe der Zeit, wenn Verfahren erste Matches etabliert hat.
            - $sim_{neighbors}(A,B) :=$ $Jaccard\big(N(A),N(B)\big) = |N(A) \cap N(B)|/|N(A) \cup N(B)|$
                - N(A) = Multimenge der Ids der Cluster aller Knoten mit Kante zu einem Knoten in A
                - Bsp:

        - ◇ x in Cluster X hat Kante zu y in Cluster Y
        - ◇ x' in Cluster X hat Kanten zu y' in Cluster Y und zu z in Cluster Z
        - ◇ N(X)={Y, Y, Z} (Multimenge, Elemente können öfters vorkommen)
    - Collective Classification
        - TODO: weggelassen

- Skalierbares Matching
    - Ziele
        - Minimiere Tupelpaare, für die Matching geprüft werden muss
            - □ Hashing, Sorting, Indexierung, Repräsanten benutzen, Kombination mehrerer Techniken
        - Minimiere Aufwand für Matchingüberprüfung eines Paars
    - Hashing
        - Hashfunktion bildet Tupel auf Buckets ab. Matching nur für Tupel im gleichen Bucket
        - Beispiel: Buckets für versch. PLZ, Matching nur auf Tupeln mit gleicher PLZ
    - Sortieren
        - Wähle Sortierschlüssel und sortiere. Vergleiche jeden Tupel nur mit Fenstergröße-1 Vorgängertupeln.
        - Beispiele Sortierschlüssel: Matrikelnummer, Nachname
        - Tradeoff Effizient vs Genauigkeit
    - Transitive Hülle, Sorted Neighbourhood Multi-Pass Ansatz
        - Nutze $a = b \; und \; b = c \Rightarrow a = c$ um äquivalente Datensätze zu erkennen.
        - Rechenmethoden für transitive Hülle sind bekannt
        -

VORNAME	MIDDLE-INITIAL	NAME
Diana	D	Ambrosian
Diana		Böhm
Diana		Dambrosian
Diana	W	Böhm

        - Problem: Bei Fenstergröße 2 werden Duplikate nicht erkannt. Sortierung nach Initialen findet ein Duplikat nicht, Sortierung nach Name findet anderes nicht.
        - Idee: Mehrfache Durchführung mit verschiedenen Sortierschlüssel. Berechnung der transitiven Hülle über Ergebnisse liefern Gesamtergebnis.
        - Vorteile
            - □ Gleiche Genauigkeit kann mit niedrigerer Miss-Rate erreicht werden
            - □ Weniger False-Positives bei gleicher Erkennungsrate (Experimentelles Ergebnis)
    - Indexierung und Canopies
        - Indexierung: Wie Sortierung. Sortierschlüssel <-> Indexschlüssel
        - Canopies: Verwende billiges Ähnlichkeitsmaß, um Tupel in Canopies (überlappende Cluster) einzuordnen. Dann jedes Paar in jedem Canopy mit teurem Ähnlichkeitsmaß abgleichen.
    - Repräsentanten
        - Clustering, sodass matchende Tupel in demselben Cluster sind.
        - Erstelle Repräsentanten für jedes Cluster. Neuen Tupel nur mit Repräsentanten vergleichen.
        - Repräsentant wird durch Auswahl oder Tupelmerges erstellt.
    - Kombination der Techniken
        - Bsp: PLZ als Hashfunktion, Sortierung in Bucket nach Straße. Dann Sliding Window.
    - Short Circuiting
        - Reduzierung des Zeitaufwandes
        - Wenn bis hierhin bestimmter Score sehr hoch, überspringe die restlichen Berechnungen.
        - (oder wenn sehr niedrig, genauso)
    - Paralellisierung