Analyse & Visualisierung
Zusammenfassung

Bernhard Fischer
Andreas Rain

23. Januar 2015

Bernhard Fischer
Andreas Rain

Inhaltsverzeichnis

Teil I

Allgemeines

Teil II

Datafoundations

1 Datatypes

Data Type	Operation	Beispiel	Bemerkung
Nominal	$==, != $	Haar Farbe, Namen, Klassen	-
Ordinal	$==, != , <, >$	Schulnote, Bewertung	Abstand spielt keine Rolle
Numerisch(Intervall)	$==, != , <, >, +, -$	Datum, Temperatur in Celsius	Abstand spielt hier eine Rolle. Beispiel: 20 Grad Celsius sind nicht doppelt so warm wie 10 Grad Celsius.
Numerisch(Verhältnis)	$==, != , <, >, +, -, *, /$	Temperatur in Kelvin, Größe, Alter	Abstand spielt auch hier eine Rolle, allerdings gibt es einen festen Nullpunkt (naturgegeben).

2 Datenklassen

Es gibt unterschiedliche Arten von Daten, darunter fallen:
D-dimensionale Datensätze, Vectoren, Hierarchische Daten (Bäume, bzw. Graphen), Metadaten (diese geben Zusatzinformationen zu den Daten).

3 Data Preprocessing

Das Daten-Preprocessing teilt sich in die Schritte, Data cleaning, Normalisierung, Segmentierung und Data reduction auf.

3.1 Messung und Fehler

$$X = T + e_r + e_s$$

Hierbei ist T der Wert, e_r ein Wert für zufällige Fehler und e_s ein Wert für den systematischen Fehler.

- Random Error - Wird als Noise (Rauschen) bezeichnet. Der Durchschnitt wird nicht beeinflusst, allerdings die Varianz.
- Systematic Error - Wird als Bias (verzerrt, verschoben) bezeichnet. Der Durchschnitt wird verzerrt.

3.2 Missing und Empty Value

Missing Value
Ein fehlender Wert, ist ein Wert, der durchaus ein Gegenstück in der Realität hat, bei der Messung oder auch später verloren gegangen ist. Man versucht Daten hinzuzufügen, die die Statistiken der Daten und deren Informationsgehalt nicht verändern.
Empty Value
Kein realer Wert vorhanden.

3.2.1 Behandlung fehlender Werte

- Ignorieren
- Manuell Wert hinzufügen (ähm jo..)
- Globale Konstante
- Mittelwert
- Wahrscheinlichster Wert (Regression, Interpolation, Entscheidungsbaum)

3.3 Data cleaning

3.3.1 Noisy Data

Daten mit Varianz oder zufälligem Fehler.

- Fehlerhafte Messung
- Problem beim Eintragen
- Probleme beim Übertragen
- Unzureichende Technologie, bzw. Beschränkung durch Technologie (Messintervall passt nicht, Skalierung zu groß (mm, cm, m))
- nicht-einheitliche Benennung und Nichteinhaltung einer Namenskonvention

3.3.2 Binning

Binning glättet (smooth) die Daten oder stuft sie zu einer gewissen Genauigkeit ab, um Varianz zu verringern.

- Equi-Depth - N Intervalle, mit ungefähr gleicher Anzahl an Elementen

- Equi-Width - N Intervalle, gleicher Größe (max - min / N, von allen Daten!)

Smoothing

- Man kann nach Mittelwert glätten
- Man kann auch Randzuweisung glätten (Nähe zum Rand bestimmt welcher Wert genommen wird)

3.3.3 Approx. VS Interpol.

Approximation
$| f(x) - f_i(x_i) | \leq \epsilon$
Hierunter fallen Regressionen

Interpolation
$| f(x) - f_i(x_i) | = 0$
Hierunter fallen polynominelle, stückweise polynominelle, orthogonale poylnominelle und trigonometrische Funktionen.

3.3.4 Regression

Versucht die Parameter einer Funktion zu finden, sodass sie die Daten möglichst genau trifft. Sie minimiert die Summer der Fehlerquadrate. Lineare Regression wird in zwei Teilen durchgeführt:

Hier findet man bei der linearen Gleichung $f(x) = a + bx$ die Variable b

$$b = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

Hier findet man bei der linearen Gleichung $f(x) = a + bx$ die Variable a

$$a = \overline{y} - b\overline{x}$$

3.3.5 Clustering

3.3.6 Überprüfung von auffälligen Werten durch Menschen

3.4 Normalisierung

Versuch die Datenpunkte zwischen 0 und 1 zu Mappen.

$$f_{lin}(v) = \frac{v - min}{max - min},$$

auch lineares Mapping genannt.

$$f_{\surd}(v) = \frac{\surd(v) - \surd(min)}{\surd(max) - \surd(min)},$$

auch Qudratwurzel-Mapping genannt.

$$f_{ln}(v) = \frac{ln(v) - ln(min)}{ln(max) - ln(min)},$$

auch logarithmisches Mapping genannt.

Bei Quantil-Normailisierung wird darauf Wert gelegt, dass sehr große und sehr kleine Werte keinen starken Einfluss haben.

3.5 Segmentierung

Problem: Es soll für einen d-dimensionalen Datensatz eine natürliche Partitionierung in Cluster und Rauschen gefunden werden. **Manuelle Segmentierung** und **automatische Segmentierung** werden hierbei betrachtet.

Bei automatischer Segmentierung werden Clustering Algorithmen auf die Daten angewendet wie z.B. k-Means und Linkage-based Methoden etc.

3.6 Daten-Reduktion

Subsetting
Datensätze werden aufgeteilt betrachtet (Sampling, Querying).

Reduktion der Dimensionen
Hier wird z.B. PCA angewendet (Datenkompression). Irrelevante und schwache, sowie redundante Attribute oder Dimensionen werden entfernt.

3.6.1 Sampling

Weshalb?
Datenmengen sind oft zu groß um sie in einem angemessenen Aufwand zu verarbeiten. Sampling soll eine möglichst repräsentative Teilmenge der Daten bilden.

Typen des Sampling: Bei *nicht wahrscheinlichkeitsbasiertem Sampling* wird auf einer nicht zufallsbasierten Basis ausgewählt.

Bei *wahrscheinlichkeitsbasiertem Sampling* gibt es folgende Methoden

- Einfaches zufallsbasiertes Sampling
  Einfache Zufallsauswahl aus den Daten
- Systematisches zufallsbasiertes Sampling
  Es wird darauf geachtet gewisse Vorbedingungen zu erfüllen
- Schichtweises zufallsbasiertes Sampling
  Gruppen werden gebildet, innerhalb derer zufällig gesamplet wird
- Cluster zufallsbasiertes Sampling
  Es werden Cluster erzeugt, die zufällig gesamplet werden
- Verzerrtes (Biased) Sampling

3.6.2 Dimensions-Reduktion

Probleme:

- eine große Anzahl an features repräsentiert ein Objekt
  **Ein feature ist ein Attribut/Eigenschaft*
- Die Daten sind schwer darzustellen
- irrelevante features können eine die Genauigkeit der Algorithmen verringern

Wie? Durch Projektion versucht man die Dimensionen zu verringern. Dabei muss man die wichtigsten features eines Objekts identifizieren, um

- die Verarbeitung zu vereinfachen, ohne Qualität zu verlieren
- die wichtigsten zwei bis drei features direkt zu visualisieren

3.6.3 Principal Component Analysis

Das Ziel ist es versteckte Faktoren, die die Daten erklären zu finden. Weiterhin können die Dimensionen der Daten verringert werden. Sie ist ähnlich zu Cluster-Mittelpunkten.

Indem man die unwichtigsten Eigenvektoren weglässt, reduziert man die Dimension im Ergebnis. Auf einer PCA können außerdem nur numerische Daten verwendet werden.

Wann wird sie benutzt? Wenn man Daten mit einer möglichst geringen Korrelation darstellen möchte. Die Varianz wird demnach maximiert.
Für welche Fälle wäre PCA nicht sinnvoll? Daten mit gleichmäßiger Abweichung in alle Richtungen, d.h. mit gleichmäßiger Relation sind nicht gut für PCA.
Robust gegen Outlier? Nein (Kreis, + Punkt)
Robust gegen Noise? Ja ist robust gegen noise (solange die Dichte des Noise nicht zu hoch ist).
Robust gegen kleine Rotationen im gesamten Datenraum? Ja, da durch eine Rotation Korrelation der Daten nicht verändert wird und die PCA die Daten sowieso rotiert.

Teil III

Classification and Prediction

4 Auswertung von Klassifikatoren

Ansätze zum Auswerten von Klassifikatoren sind

Train-and-Test

- Zunächst werden Trainingsdaten verwendet um Klassifikatoren aufzubauen.
- Daraufhin werden Testdaten verwendet um diese zu validieren.

m-fold cross validation

- Datenmenge in m Teilmengen der gleichen Größe aufteilen.
- m Klassifikatoren trainieren indem jeden Klassifikator m-1 der Teilmengen als Trainingsdaten verwendet wird. Als Testdaten wird die übrig gebliebene Teilmenge verwendet.
- Die Auswertung der m Klassifikatoren wird nun kombiniert.

Kriterien für Auswertung

- Genauigkeit der Klassifikation
- Nachvollziehbarkeit, evtl. Deutbarkeit
- Effizienz, Model-Konstruktion und -Anwendung
- Muss sich auch für große Datenmengen eignen
- Robustheit (gg. Noise, unbekannte Attribute etc.)

Klassifikations-Genauigkeit Die Genauigkeit der Klassifikation kann durch folgende Formel bestimmt werden

$$G_{TE}(K) = \frac{|\{o \in TE | K(o) = C(o)\}|}{|TE|}$$

Im Prinzip also nur die Anzahl der korrekt klassifizierten Objekte geteilt durch die Gesamtanzahl an getesteten Objekte.

Klassifikations-Fehler Der Fehler der Klassifikation kann durch folgende Formel bestimmt werden

$$F_{TE}(K) = \frac{|\{o \in TE | K(o) \neq C(o)\}|}{|TE|}$$

Im Prinzip also nur die Anzahl der falsch klassifizierten Objekte geteilt durch die Gesamtanzahl an getesteten Objekte.

Confusion-Matrix

	Predicted as positive	Predicted as negative
Actually positive	True Positive	False Negative
Actually negative	False Positive	True Negative

Man definiere für eine gegebene Klasse zwei Maße wie folgt:

$$Precision(K) = \frac{|\,TP\,|}{|\,TP\,| + |\,FP\,|} \text{ und}$$
$$Recall(K) = \frac{|\,TP\,|}{|\,TP\,| + |\,FN\,|}$$

Beide Maße stehen in einem umgekehrten Verhältnis zueinander (trade-off).
Nun wird zudem das F-Measure von K definiert:

$$F - Measure(K) = \frac{2 \cdot Precision(K) \cdot Recall(K)}{Precision(K) + Recall(K)}$$

Ein möglichst hoher F-Measure-Wert ist erwünscht.

True Classification Error

Es werden n Elemente zufällig aus der Gesamtdatenmenge ausgewählt und klassifiziert. Der Anteil falscher Klassifikationen ist der True Classification Error.

5 Entscheidungsbäume

Wird durch Vergleiche von Features (Attributen) aufgebaut.
Blätter entsprechen den gegebenen Klassen.

Man kann den Baum entweder auf Basis von Traingsdaten und der Top-Down Strategy aufbauen. Indem man von der Wurzel bis zur Klasse durchläuft kann man Datensätze klassifizieren.

Construction Algorithmus:
Alle Daten gehören zur Wurzel. Man wählt dann das wichtigste Attribut und führt einen sinnvollen Split dieses Attributs durch. Die Trainingsdaten werden gemäß des Splits aufgeteilt. Dies wendet man auf alle weiteren Attribute innerhalb der Teilmengen an.

Der Algorithmus terminiert wenn es keine Attribute zum Splitten mehr gibt oder die meisten Trainingsdaten eines Knoten zu der selben Klasse gehören.

Typen von Splits

- Kategorisch $(==, !=)$, viele Teilmengen möglich
- Numerisch $(==, != , <, >, \leq, \geq)$, viele Splits möglich

Qualität von Splits
Gegeben ist eine disjunkte, abdeckende Partitionierung von T. p_i ist die Häufigkeit der Vorkommen einer Klasse c_i in T.
Man möchte nun ein Maß für die Unreinheit der Menge S unter Berücksichtigung von Class Labels. Ein Split von T soll diese Unreinheit minimieren → INFORMATION GAIN, GINI-INDEX

5.1 Information Gain

Entropie

$$entropy(T) = -\sum_{i=1}^{k} p_i \cdot log_2 p_i$$

Information Gain
Wird definiert durch

$$InformationGain(T, A) = entropy(T) - \sum_{i=1}^{m} \frac{\mid T_i \mid}{\mid T \mid} \cdot entropy(T_i)$$

5.2 Gini-Index

$$gini(T) = 1 - \sum_{j=1}^{k} p_j^2$$

Kleiner Gini-Index → geringe Unreinheit
Großer Gini-Index → Hohe Unreinheit

Gini-Index für ein bestimmtes Attribut in Abhängigkeit der Klasse:

$$gini_a(T) = \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot gini(T_i)$$

5.3 Overfitting

Hat man eine zu hohe Klassifikationsgenauigkeit, so ist zwar das Ergebnis auf Traingsdaten gut, wird aber auf Testdaten schlechter.

Wie verhindern?

- Entfernen fehlerhafter Trainingsdaten
- Auswahl geeigneter Trainigsdaten-Größe
- Auswahl eines geeignetem Minimums-Support (Minimale Anzahl von Traingsdaten die zu einem Blatt gehören)
- Auswahl einer geeigneten minimalen Confidence (Meist vorkommende Klasse soll geringen Anteil an Blattknoten haben)
- Anschließendes Säubern des Entscheidungsbaums (Zweige weglassen)
- Cross-Validation (Daten aufteilen in Trainingsdaten und Testdaten (9 zu 1 bspw.))
- Anzahl betrachteter Attribute verringern

5.4 Säuberung zum verringern von Fehlern

- Train-and-Test
- Abschneiden von Zweigen
- Wieder testen
- und so weiter
- und so fort

5.5 Minimaler Aufwand zum Säubern der Daten

Cross-Validation: Sogar anwendbar, wenn nur wenige gelabelte Daten vorhanden sind.
Pruning: Man säubert den Entscheidungsbaum mit den Trainingsdaten und kann den Klassifikationsfehler nicht als Qualitätsmaß nutzen.

Kleinere Entscheidungsbäume sind tendenziell besser für noch nicht da gewesene Daten.

5.6 Entscheidungsbaum-Algorithmen im Vergleich

C4.5 vorteile gegenüber ID3

- Man kann auch mit Zahlenwerten arbeiten
- Modifizierte Split-Kriterien (Gain-Ratio)
- Regeln extrahieren
- Stopped das verarbeiten von Nodes die keinen Gewinn bringen
- Nutzen von post-pruning Methoden
- Windowing

5.7 Gain Ratio

Informationsgehalt ist durch Tests mit viele möglichen Ergebnissen verzerrt. Die Gain Ratio versucht dies zu beheben (oder überbrücken).

$$SplitInformation(S, A) = -\sum_{i=1}^{c} \frac{\mid S_i \mid}{\mid S \mid} log_2 \frac{\mid S_i \mid}{\mid S \mid}$$

$$GainRatio(S, A) = \frac{InformationGain(S, A)}{SplitInformation(S, A)}$$

6 Bayesian Classification

Für jedes zu klassifizierende Objekt wird die Wahrscheinlichkeit berechnet.
Der Optimale Bayes Klassifikator versucht genau dies umzusetzen und ist wie folgt definiert:

$$argmax_{c_j \in C} \sum_{h_i \in H} P(c_j \mid h_j) \cdot P(h_i \mid o)$$

Kennt man die zu wählende Hypothese, kann man direkt den Maximum-Likelihood-Classifier nutzen, der vom optimalen Bayes-Classifier abgeleitet ist.

6.1 Bayesian Networks

Knoten sind Zufallsvariablen, Kanten sind Abhängigkeiten. Jede Zufallsvariable ist bedingt unabhängig von allen anderen Variablen die nicht erfolgreich sind. Für jeden Knoten wird eine Tabelle der bedingten Wahrscheinlichkeiten erzeugt.

Trainieren eines solchen Netzwerks geht mit Vorwissen und auch ohne Vorwissen.

6.2 Vor- und Nachteile

- + Optimalitätseigenschaft (Wird zum Vergleich mit anderen Klassifikatoren verwendet)
- + Hohe Klassifikationsgenauigkeit
- + Incrementality

- + Integration of domain knowledge
- - Anwendbarkeit
- - Ineffizient

7 Neurale Netzwerke

7.1 Vor- und Nachteile

- + generell hohe Klassifikationsgenauigkeit
- + Robust gegen rauschen
- + Effizientes Anwenden
- - Aufbau dauert lange
- - Schwer nachzuvollziehen
- - Kein bekanntes Wissen kann genutzt werden, da nicht nachzuvollziehen

8 Nearest-Neighbor Classification

- Mean-Vektoren für alle Objekte einer Klasse berechnen
- Zu klassifizierendes Objekt der Klasse zuweisen, von welcher der Mean-Vektor dem Objektvektor am nächsten ist.
- Es können auch mehr als nur 1 Nachbar berücksichtigt werden.
- Klassen können gewichtet werden.
- benötigt wird eine Distanz-Funktion
- benötigt wird k, die Anzahlder berücksichtigten Nachbarn
  zu klein → sehr anfällig gegen Outlier
  zu groß → es werden zu viele Objekte von anderen Klassen beachtet
  mittlere k → meist beste Ergebnisse

8.1 Vor- und Nachteile

- + Lokale Methode (??)
- + Hohe Klassifikationsgenauigkeit
- + Inkrementell
- + kann für Vorhersagen benutzt werden
- - teure Anwendung (prüfen, welcher der nächste Nachbar ist)
- - Generiert kein genaues Wissen über die Klassen

9 Support Vector Machines

- Sucht die Hyperebene, welche den Datenraum am besten zerteilt
- Hyperplane mit maximiertem Abstand zu den nächsten Trainingsobjekten wird ausgewählt

9.1 Vor- und Nachteile

- + Starke mathematische Grundlage
- + Findet das globale Optimum
- + Skaliert gut zu sehr hochdimensionalen Daten
- + Hohe Klassifikationsgenaujigkeit
- - Ineffizienter Modellaufbau
- - Modell kann kaum interpretiert werden (lernt ausschließlich Gewichte)
  Gewichte tendieren dazu, gleichmäßig verteilt zu sein

Teil IV

Clustering

10 Einführung

10.1 Definition

Was ist Clustering?
Clustering identifiziert eine endliche Menge an Kategorien, Klassen oder Gruppen, wobei Objekte innerhalb eines Clusters ähnlich sein sollen und jene die sich außerhalb dieses Clusters befinden, möglichst wenig gemeinsam haben.

Eigenschaften

- unterschiedliche Größe, Form, Dichte
- Hierarchie
- Überlappend oder auch Disjunkt

10.2 Ziele der Cluster-Analyse

- **Data Understanding** - Finden natürlicher Cluster
- **Data Class Identification** - Finden nützlicher und passender Gruppierungen
- **Data Reduction** - Repräsentanten homogener Gruppen finden
- **Outlier Detection** - Beachte lokale und globale Outlier!
- **Noise Detection**

10.3 Basis Definitionen

Distanz-Funktionen Größer null, bei Gleicheit 0, Symmetrie und Dreiecksungleichheit.

$$\text{Lp-Metric: } dist(x,y) = \sqrt[p]{\sum_{i=1}^{d} \mid x_i - y_i \mid^p}$$

Alle anderen Metriken (Euklid p=2 und Manhatten p=1) werden aus dieser Metrik abgeleitet. Es handelt sich bei x, y um d-dimensionale Vektoren.

11 Custering Methoden

11.1 Patritionierungs Methoden

Soll disjunkte Partitionierung in k Clustern unter minimalen Kosten finden.

Lokale Optimierungsmethode

- k Clusterrepräsentanten wählen
- Repräsentanten iterativ optimieren
- jedem Objekt den Ähnlichsten Repräsentanten zuweisen

Typen von Clusterrepräsentanten

- Mittelwert
- Median
- Wahrscheinlichkeits-Dichtefunktion

11.1.1 Forgy 1965 (Kompaktheit)

Objekte im Euklidischen Vektorraum.
Kompaktheit bestimmen mittels: $TD^2 = \sum_{i=1}^{k} TD^2(C_i)$, wobei $TD^2(C) = \sum_{p \in C} dist(p, \mu_c)^2$

11.1.2 k-Means (+ Medoid)

1. Objekte in k nichtleere Mengen aufteilen
2. Jeder Partitionierung den Centroid zuweisen
3. Jedem Objekt den nächsten Centroid zuweisen
4. Zwei und drei wiederholen, bis es nicht mehr geht oder Schleife.

Vor- und Nachteile

- + recht effizient $\mathcal{O}(tkn)$, t Iterationen
- + einfach zu implementieren

- - Terminiert möglicherweise an lokalem Optimum (nicht wirklich das Optimum).
- - Nur anwendbar wenn Mittelwert berechenbar
- - k festlegen
- - kann nicht mit noise und outliers umgehen
- - sucht immer konvexe Cluster

ISODATA Verbesserung des k-Means Verändert die Anzahl k im k-Means, falls ein Cluster keiner bestimmten Größe entspricht und teilt sie auf, falls die Standardabweichung einen Wert übersteigt. So passt sich das k an falls es nicht gut gewählt ist. Nachteil ist dass mehr Parameter angegeben werden müssen.

11.1.3 Wahl repräsentativer Punkte (Kompaktheit)

Kompaktheit bestimmen mittels: $TD = \sum_{i=1}^{k} TD(C_i)$, wobei $TD(C) = \sum_{p \in C} dist(p, m_c)$ und m_c ein repräsentatives Element des Clusters ist.

Komplexität $\mathcal{O}(n^k)$.

11.1.4 PAM (Partitioning Around Medoids)

k-Repräsentaten wählen, aus allen Kombinationen Kompaktheit (Kosten) berechnen und Kombination auswählen, welche geringste Kosten hat. Zum Schluss alle Objekte den nächsten Medoiden zuweisen.

Nachteile: Kann bei lokalem Optimum terminieren.
Komplexität schlecht.

11.1.5 Clara (CLustering LARge Applications)

Wie PAM, nur Medoiden werden auf einer Teilmenge (Sample) berechnet.
Wird für mehrere Samples wiederholt, bestes wird benutzt.

11.1.6 Clarans (CLustering LARge Applications based upon RANdomized Search)

Wie Clara, nur Anzahl der Iterationen und Anzahl der geprüften Nachbarn pro Iteration sind begrenzt.

Vorteile: In der Praxis meist eine Laufzeit von $\mathcal{O}(n^2)$.

11.1.7 Expectation Maximization (EM)

Generalisierung von k-Means: die Zuweisung der Punkte basiert auf Wahrscheinlichkeiten.

Idee:
Schätzen der Parameter der k Normalverteilungen.
Optimieren der Wahrscheinlichkeiten, dass der Mix der parametrisierten Normalverteilungen zu den Daten passt

Modifikation: Jedes Objekt dem Cluster hinzuzufügen, zu welchem die höchste Wahrscheinlichkeit besteht.

Vorteile:
Komplexität liegt in $\mathcal{O}(n \cdot k \cdot \#iterations)$

Nachteile:
Kann bei lokalen Minima terminieren.
Clustering-Ergebnis und Laufzeit hängt stark vom Initial-Cluster und einer guten Wahl des Parameters k ab.

!!!kommt das dran??!!!

11.2 Linkage Based Methods

11.2.1 Single Linkage, Complete Linkage, Average Linkage, Centroid Linkage

Ziel:
- Erstellung eines Dendrogramms (Cluster-Hierarchie)
- Die jeweils ähnlichsten Cluster sollen gemerged werden.

Verfahren:
- Top-down (Dividing): Alle Objekte initial in einem Cluster, dann Splits
- Bottom-up (Agglomerating): Jedes Objekt initial ein eigenes Cluster, dann mergen

Bottom-up-Clustering:
1. Jedes Objekt initial ein eigenes Cluster
2. Merging von jeweils den 2 Clustern mit der geringsten Distanz.
3. Berechne Distanz zwischen dem neuen Cluster und allen anderen Clustern
4. Stop, wenn alle Objekte zu einem Cluster gehören, sonst zu Schritt 2.

Distanzfunktionen:

- Single-Linkage: $dist_{single}(C_1, C_2) = min\ dist(p, q)$
- Complete-Linkage: $dist_{complete}(C_1, C_2) = max\ dist(p, q)$
- Average-Linkage: $dist_{avg}(C_1, C_2) = \frac{1}{\#C1 \cdot \#C2} \sum_{p \in C_1} \sum_{q \in C_2} dist(p, q)$
  Bei Avg-Linkage wird für jeden Punkt aus C_1 zu jedem Punkt aus C_2 die Distanz berechnet und aufsummiert. Es wird der Mittelwert berechnet, indem durch das Produkt von C_1 und C_2 geteilt wird.
- Centroid-Linkage: $dist_{mean}(C_1, C_2) = dist[mean(C_1), mean(C_2)]$

  Vorteile:

  - k muss nicht vorher bestimmt werden
  - Hierarchische Cluster werden gefunden
  - Es können beliebige Clusterings gefunden werden, indem man das Dendrogramm an verschiedenen Stellen horizontal durchschneidet.

  Nachteile:

  - Merges und Splits können nicht widerrufen werden
  - Nicht robust gegen Noise (unter Umständen kann eine Kette mehrere Cluster miteinander verbinden)
  - Ineffizient (mindestens $\mathcal{O}(n^2)$)

11.2.2 BIRCH

Es werden Zusammenfassungen von „Mikro-Clustern“ bestimmt. Auf die unten liegenden Cluster in dem CF-Tree kann dann ein beliebiger Clustering Algorithmus angewandt werden.

11.2.3 CF-Tree

CF steht für Clustering Feature einer Menge C aus Punkten X_i: $CF = (N, LS, SS)$, wobei

$$N = \mid C \mid \text{ Anzahl der Punkte in C,}$$
$$LS = \sum_{i=1}^{N} X_i \text{ lineare Summer der N Punkte,}$$
$$SS = \sum_{i=1}^{N} X_i^2 \text{ quadratische Summe der N Punkte}$$

CF's sind hinreichend um Centroide, Kompaktheitsmaße und Distanzfunktionen zu berechnen.

Additions-Theorem CF's von zwei disjunkten Clustern sind additiv: $CF(C_1 \cup C_2) = CF(C_1) + CF(C_2) = (N_1 + N_2, LS_1 + LS_2, QS_1 + QS_2)$, können inkrementell berechnet werden.
Ein CF-Tree ist ein höhenbalancierter Baum zum Speichern von Clustering Features.

Eigenschaften

- Jeder innere Knoten hat höchstens B Einträge
- Ein Blatt hat höchstens L Einträge
- Jedes Blatt hat zwei Pointer, next & prev
- Der Durchmesser eines Eintrags in einem Blatt überschreitet T nicht.

Aufbau eines CF-Tree

- Transformiere Objekt p in $CF_p = (1, p, p^2)$
- Einfügen des CF in das nächst Blatt
- Wenn Durchmesser T verletzt wird, versuche einen neuen Eintrag im Blatt zu machen. Wird hierbei L verletzt teile das Blatt auf.

Einfüge-Algorithmus für Punkt X

1. Finde das nächste Blatt b
2. Falls x in b passt, füge x in b ein; ansonsten teile b auf
3. Modifiziere den Pfad für b
4. Falls Baum zu groß → Verkleinere den Baum, indem die nächsten Blätter gemerged werden (ähnlich wie abschneiden, da Parent die Summe der unten liegenden CF's)

Vor- und Nachteile

- + CF-Tree Größe / Kompressions-Faktor hängt vom User ab
- + Effizienz (O(n) - O(n log n)...)
- - Nur für numerische Daten
- - Ergebnisse sind von der Reihenfolge der Datenobjekte abhängig

11.3 Dichtebasiertes Clustering

Idee: Cluster sind dichte Gebiete in einem d-dimensionalen Datenraum. Gebiete werden durch niedrigere Dichten unterteilt.

Anforderungen Für jedes Cluster, muss die lokale Dichte eine gewisse Grenze überschreiten. Die Objekte eines Clusters müssen räumlich verbunden sein.

Stärken Custer können beliebige Form haben. Robust gegen noise und effizient.

11.3.1 Basics

Core Point
Objekt o hat mindestens MinPts Nachbarn in seiner Nähe (nicht weiter entfernt als ε)

Border Point
Alle die keine Core Points sind.
directly density-reachable Point
Ein Nachbar eines Core Point.
density-reachable Point
Transitiv erreichbarer Punkt über mehrere Core Points.
density-connected points
p und q sind density-connected, falls beide zu einem dritten Punkt density-reachable sind.
Cluster
Cluster müssen Maximalität und Konnektivität haben.
Maximality: $\forall p, q \in D : if\ p \in C$ und q density-reachable von p ist, dann ist $q \in C$
Connectivity: $\forall p, q \in C : p$ ist density-connected zu q
Noise Alles was keinem Cluster zugeordnet werden konnte ist noise.

Wahl der Parameter
Es gibt die Parameter MinPts und ε, diese müssen passend gewählt werden.
Problematisch sind hier unterschiedliche Dichten von Clustern (+ hierarchische Cluster).

11.3.2 DBSCAN

Für jeden Punkt bestimmt DBSCAN die ε-Umgebung und prüft ob genügend Objekte enthalten sind. DBSCAN nutzt Index-Strukturen um die ε-Umgebung zu bestimmen. Hierdurch können beliebige Cluster-Formen gefunden werden.

Vor- und Nachteile

Vorteile:

1. Kann sehr gut mit verschiedenen Formen von Clustern umgehen (siehe z.B. Folie 05.99).
2. Ist bei geeigneter Parameterwahl gegen Noise und Outlier robust (falls die Cluster eine zueinander ähnliche Dichte haben und der Noise weniger dicht ist).
3. Ist effizient, da man eine Indexstruktur für die Berechnung der Nachbarn in Reichweite nutzen kann.

Nachteile:

1. Parameterwahl gestaltet sich unter Umständen schwierig. Werden ϵ und MinPts nicht adäquat gewählt, können völlig unerwünschte Clustering-Ergebnisse entstehen.

2. Cluster mit unterschiedlichen Dichten können nur sehr begrenzt geclustert werden, da z.B. Noise mit in ein Cluster genommen wird, nur um ein anderes mit geringerer Dichte auch erkennen zu können.

11.4 Hierarchical Density-Based Clustering (Optics)

Für konstante MinPts-Werte werden kleine ε density-based Clusters von welchen mit größerem ε absorbiert.

Clusterings für unterschiedliche Dichte-Parameter können mittels eines einzigen Scans bestimmt werden:
zunächst Cluster mit hoher Dichte finden, dann mit niedrigerer Dichte.

Es wird kein Dendrogramm erzeugt, jedoch eine grafische Visualisierung der Hierarchie.

11.4.1 Basics

Core Distance

$$CoreDistance_{\varepsilon,MinPts}(o) = \begin{cases} ^{undef,\ |N_\varepsilon(o)|<MinPts} \\ MinPtsDistance(o) \end{cases}$$

Einfach gesagt, ist die CoreDistance die geringste Distanz, sodass o noch ein Core Object ist.

Reachability Distance

$$ReachabilityDisance_{\varepsilon,MinPts}(p,o) = \begin{cases} ^{undef,\ |N_\varepsilon(o)|<MinPts} \\ max\{CoreDistance(o), dist(o,p)\} \end{cases}$$

Geringste Distanz, sodass p noch directly density-reachable zu o ist.

Vor- und Nachteile

Vorteile:

1. Im Gegenteil zu DBSCAN können mit OPTICS auch hierarchische Cluster gefunden werden (Cluster mit unterschiedlicher Dichte).
2. Ist effizient, da man eine Indexstruktur für die Berechnung der Nachbarn in Reichweite nutzen kann.

Nachteile:

1. OPTICS ist sehr abhängig von der Wahl der Parameter. Allerdings sind die Ergebnisse gut, wenn man eher zu große Werte verwendet.

11.5 Kernel-Density Estimation

Wie EM.

11.5.1 DENCLUE

Kann bei entsprechender Parametrisierung verschiedene Cluster-Verfahren generalisieren, wie dichtebasierte, partitionierende oder hierarchische Clustering-Verfahren. Dafür wird eine Dichtefunktion verwendet, welche den Einfluss eines Objektes auf seine Nachbarn beschreibt. Die gesamte Dichte

des Datenraums ergibt sich aus den Dichtefunktionen aller Objekte. Lokale Maxima dieser Dichtefunktion ergeben die einzelnen Cluster.

Vorteile:

1. DENCLUE ist robust gegen Noise.
2. Unregelmäßig geformte Cluster werden gut gefunden.
3. Hochdimensionale Daten können verarbeitet werden.

Nachteile:

1. Die Parametrisierung ist problematisch und muss vom Anwender gewählt werden.

Teil V

Association Rules

Man möchte Muster und häufig auftretende Beziehungen zwischen Einträgen in den Daten erkennen.

12 Basic concepts

12.1 FPA

Frequent pattern: Ein Muster das oft im Datensatz aufkommt
Motivation?: Man möchte Regelmäßigkeiten in den Daten feststellen, z.b. welche Produkte werden zusammen gekauft?
Anwendungsbereiche: Warenkorb-Analyse, Cross-Marketing uvm.

12.2 Warum FPA?

Kristallisiert wesentliche und wichtige Eigenschaften aus den Daten heraus.
Ist Grundlage für viele wichtige data mining aufgaben:

- Beziehung, Korrelation, Ursache für Auftreten
- Aufgeteilte und strukturelle Muster
- Muster Analysie bei multimedia, Zeitserien, Streamingdaten..
- Assoziative Klassifikation
- FPA basiertes Clustering
- semantische Daten-Kompression

12.3 Definitionen

Itemset: Eine Menge an Items..
Support count(): Wie oft ein Itemset vorkommt
Support: Support count() / Gesamtzahl der Itemsets (Transaktionen).
Frequent Itemset: Ein Itemset das ein ε (bezeichnet mit minsup) übersteigt.

12.4 Was ist eine Association Rule?

Itemset X $\rightarrow$ *Itemset Y*

Rule Evaluation Metrics (Maße zum bewerten von Assoziations-Regeln:

- Support(s)
  - wie oben beschrieben.
- Confidence(c)
  - Itemset A durch das Itemset B teilen, dass das Itemset A \ Itemset B impliziert.

12.5 Ziel von AR

Man möchte alle Regeln finden die für alle Transaktionen vorhanden sind.

Zwei Schritte:
1. Generiere Itemset mit Support $\geq \varepsilon$.
2. Regeln generieren $\rightarrow$ nur die nehmen mit hoher Confidence.

Ist nach wie vor recht viel zu verarbeiten.

13 Frequent itemset mining

Methoden hierfür sind: Apriori, FP Growth usw.

Beim Brute-Force-Ansatz wäre jedes Itemset ein candidate frequent itemset. Man würde einfach jeden Support zählen, indem man durch die ganze Database durchgeht. Komplexität ist zu hoch.

Abhilfe:

- Reduzieren der Anzahl der candidates (z.B. Pruning)
- Reduzieren der Anzahl an Transaktionen
- Reduzieren der Anzahl der Vergleiche (Indexstrukturen, man braucht nicht alle candidates mit allen Transaktionen vergleichen)

13.1 Apriori-Algorithmus

Prinzip:
Wenn ein Itemset frequent ist, dann müssen auch alle Teilmengen davon frequent sein.
Wenn eine Teilmenge nicht frequent ist, soll die Obermenge erst gar nicht getestet werden.

Methode:

1. Database nach L_1 frequent 1-itemsets scannen
2. Generieren von k-candidate-itemsets von den k-1-frequent-itemsets
3. Generierte candidates gegen Database testen
4. Ende, wenn keine frequent oder candidate itemsets mehr generiert werden können

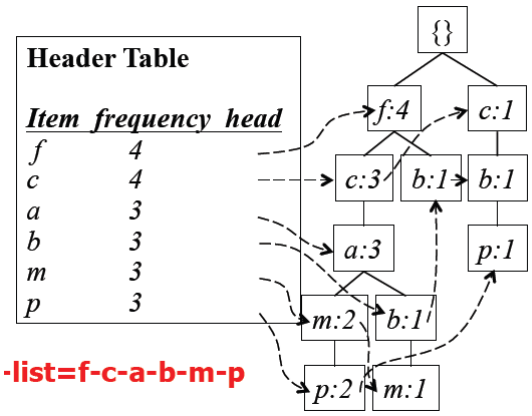
Zur **Generierung** der k-itemsets kann gejoint werden, wenn sich die itemsets bis auf das letzte Element gleichen.
Zum **Pruning** von $i \in C_k$ können alle in i k-1-elementigen Teilmengen geprüft werden, ob sie in C_{k-1} enthalten sind. Wenn nein $\rightarrow$ löschen.

13.2 FP-Tree

Vorgehensweise:

1. Frequents aus der Datenbank bestimmen (für 1-Elementige Teilmengen) (wrt. MinSup).
2. Frequent Itemsets nach Häufigkeit sortieren ({f,a,c,b} $\rightarrow$ {f,c,a})
3. DB wieder scannen und den FP-Tree konstruieren

Man sollte zudem die Header Table machen, sodass die heads auf den Tree zeigen:



Konstruktion: *(Min Support ist bestimmt, und nur die, die den MinSup erfüllen werden aufgenommen)*
Baum aufbauen, indem anhand der sortierten frequent itemsets Pfade konstruiert werden. Kommt man bei einem Item mehrmals vorbei, wird gezählt wie oft.

Vorzüge eines FP-Trees:

- **Vollständigkeit**:

  - Man behält alle Informationen für FP-Mining
  - Man macht keine längeren Muster von Transaktionen kaputt

- **Kompaktheit**:
  - Irrelevante Informationen fallen weg (nicht-frequente items..)
  - Absteigend nach Frequency
  - Ist nicht größer als die DB

Aufstellen der Conditional Patternbase
Man schreibt für jedes Item in der Header Table die Conditional Pattern Bases separat auf, indem für das jeweilige Item, jedes Vorkommen im FP-Tree betrachtet wird und der Pfad von oben bis zu diesem Vorkommen aufgeschrieben wird. Die Anzahl entspricht dann der Anzahl des Vorkommens selbst.

Aufstellen der Conditional FP-Trees für bestimmte Items aus der Header Table
Beachte: Für den FP-Tree gilt auch MinSup, wie beim vorherigen.

- Für jedes frequent Item wird aus der Conditional Pattern Base ein Conditional FP-Tree aufgebaut, solange es in der zugehörigen Patternbase mindestens 1 Item gibt, welches minsup erfüllt.
- Für jeden erhaltenen Baum wird rekursiv wieder eine Patternbase erstellt (für p-conditional jetzt ein px-conditional und dann ein pxy conditional) und der dazugehörige Conditional FP-Tree für die entsprechenden Conditional Patterns aufgebaut.
  Beachte: Für jeden Baum muss es auch einen Header-Table geben

13.2.1 FP-Growth

Methode:
FP-Tree aufbauen, und rekursiv alle tieferen conditional-FP-Trees aufbauen. Stoppen wenn nur noch ein Pfad vorhanden ist oder der resultierende FP-Tree leer ist.

Vorteile:

- Divide and conquer
  - Trennt das Mining vom Database-Management, abhängig von den schon ermittelten Trees.
  - Verringert die jeweils anzuschauenden Daten, je spezifischer die Itemsets, welche man vergleichen will, werden.
- Andere Faktoren
  - Es müssen keine Candidates generiert und getestet werden
  - FP-Tree ist eine komprimierte, effiziente Datenstruktur

14 Mining various kinds of association rules

15 From association mining to Correlation analysis

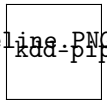
16 Constraint- based association mining

17 Zusammenfassung

Teil VI

Human Perception

18 KDD Pipeline



19 Warum muss man auf Wahrnehmung achten?

- Beispiel Cockpit (Wo ist was?)
- Brille, Hörhilfe..
- Film, Tv,...
- Medizin: Reparieren von kaputten Sinnesorganen und Nervensystemen

20 Gestalt Laws

- Gesetz der Nähe (Proximity)
- Gesetz der Geschlossenheit (Closure)
- Gesetz der Symmetrie
- Figur-Grund Abtrennung (Figure-Ground Segregation)
- Gesetz der guten Fortführung (Good Continuation)
- Gesetz der Ähnlichkeit (Similarity)

21 Was nimmt also Einfluß auf unsere Wahrnehmung?

- Kontrast

- Geometrische Effekte
- Gestalt Laws s.o.

Teil VII

Visualization Techniques

22 Darstellungsgrundlagen

22.1 Basics

22.1.1 Expressiveness& Effectiveness

Expressiveness: Die Visualisierung teilt alle Informationen mit und nur diese Informationen.
Effectiveness: Die zu teilenden Informationen für die Person sollen möglichst schnell interpretiert werden können.

22.2 Visuelle Variablen

- Position
- Farbe
- Helligkeit
- Form
- Ausrichtung
- Größe, Länge, Volumen
- Textur

22.2.1 Effekte visueller Variablen

Selektiv: Wenn Daten mit solchen selektiven Variablen codiert / visualisiert wurden, dann unterteilt der Mensch diese automatisch in Gruppen. • Größe • Helligkeit • Textur • Farbe • Ausrichtung	**Assoziativ:** Man stellt ähnliche Objekte in Beziehung. • Textur • Color, Helligkeit • Ausrichtung • Form	**Separation.** • Textur • Farbe • Ausrichtung • Form
Ordinal: Man nimmt Ordnung wahr. • Textur • Farbe, Helligkeit	**Proportional:** Man nimmt relative Größe wahr. • Größe • Ausrichtung • Helligkeit	