Analysetechniken für große Datenbestände 1 - Algorithmen

Lukas Bach - lbach@outlook.de - lukasbach.com

1.1 Nächster Nachbar auf Kd-Baum

Rekursive Traversierung über alle Knoten, angefangen bei Baumkopf. Elemente werden bei Traversierung in Warteschlange eingefügt, mit Distanz zur Anfrage als Kriterium. Erstes durch Schleife entdecktes Element wird zurückgegeben,

1.2 Entscheidungsbaum Split

Datenmenge wird rekursiv gesplittet, jeweils mit minimaler Splitentropie für Splitattribut und -Wert.

Splitentropie ist entsprechend gesplitteter Mengenanzahl gewichteter Mengenentropie.

1.3 Apriori

- Erzeugen der 1-elementigen FIsets
- Schleife unter k
  - Join, Prune, Support Count (Hashtree)
- Frequent Itemsets => Assoc Rules
  - Für I: Ist a => (I-a) Assoc Rule?

1.4 Sampling

Auf ~~Sample~~ werden k-elementige FIsets gesucht um Negative-Border (mit minsup) zu ermitteln, diese wird dann in einem DB-Scan geprüft. Aus Bereich um Negative Border werden dann interessante FIsets extrahiert, oder Berechnungen werden auf Sample aus Negative Bórder durchgeführt.

1.5 FP-Tree

- FIsets werden nach *pro Transaktion* nach Gesamthäufigkeit sortiert
- Überprüfung in Baumstruktur. Transaktionen werden als Äste in Baum gemergt mit Counting, Präfixpfade werden geprüft und daraus werden häufige Patterns extrahiert.

Projected Databases werden für Darstellung des Ergebnis verwendet. Mehrere PDBs, um jeweils bestimmte Menge an FIsets zu finden.

1.6 K-Means (Hill-Climbing)

- Bestimme k Medoiden (Seeds) aus Datenbestand
- Schleife: Ordne Datenpunkte dem nächstem Medoiden zu, berechne Medoid neu. Bis Verbesserung nicht mehr signifikant.

1.7 BIRCH

- Baue CF-Baum auf
- Neuanordnung der Subcluster in Blättern durch globalen Clustering-Algorithmus.
- Centroide der Cluster als Seeds nehmen, Datenpunkte neuverteilen

Mitgeführtes Cluster-Feature: CF=(N, LS, SS)

1.7.1 CF-Baum

Höhenbalancierter Baum, Blätter sind CFs (Elementarfeatures). Parameter: Fanout, Blattkapazität, Elementarcluster Radiusschwellenwert.

Einfügen: Punkt wandern zu Knoten mit geringstem Abstand zu Knotenschwerpunkt. Node-Splitting oder Merging falls Grenzen überschritten. Änderungen nur mit CFs.

1.8 Hierarchisches Clustering

Agglomerativ: triviales Vorgehen.

Divisiv: DIANA

- Suche abseitigstes Datenobjekt in Cluster C/nächstes Datenobjekt zu Splintergroup, dieses zu Splitergroup dazufügen. Objekte werden dann in nähere „Gruppe“ eingefügt.
- Wenn Abstand ausgeglichen: Aufteilen in Splintergroup/Rest

1.9 Projected Clustering

Wie kMeans, am Schleifenbeginn (bevor Datenobjekte Medoiden zugewiesen werden) werden Dimensionen zu jedem Medoiden bestimmt.

Dabei wird Locality bestimmt (Locality reicht bis zu nächstem Medoiden). Jetzt werden interessante Dimensionen bestimmt, für die Punkte nah an Medoid sind.

Für alle Punkte der Locality wird Durchschnittsdistanz zu Clusterpunkten über Dimensionen berechnet (Manhatten Segmental Distance). Diese wird von Durchschnittsdistanzen von konkreten Dimensionen abgezogen, um zu entscheiden, ob Dimension wichtig ist.

1.10 Linkbasiertes Clustering für kategorische Attr

Jaccard quantifiziert Ähnlichkeit von Transaktionen. Zwei Punkte sind Nachbarn, wenn Ähnlichkeit > Schwelle. #Links zw Punkte = #gemeinsame Nachbarn. Wiederholtes mergen von Punkten/Clustern mit maximaler Linkanzahl.

1.11 DBSCAN

- Für alle unbesuchten Punkte:
  - Falls Nachbarschaft<minpts: Noise
  - Sonst, füge Punkt zu Cluster dazu und gehe in Rekursion für alle ungeclusterten Nachbarn:
    - Clustere Punkt, falls Punkt unbesucht, markiere als besucht und falls dicht rekursiere über dessen ungeclusterten Nachbarn.

1.12 Optics

Core-Distance$_{MinPts}$(o): kleinster ε-Wert, sodass o dicht

Reachability-Distance$_{\varepsilon,minPts}$(p,o): Abstand zw p und o, falls größer als coreDist(o), sonst coreDist(o). Falls Abstand größer als **ε**: undef

Objekte sollen so sortiert werden, dass sie möglichst nah auf das ihnen zugeordnete dichte Objekt folgen. Objekte desselben Clusters direkt hintereinander.

Vorgehensweise wie iterativer DBSCAN, ControlList als PrioQueue mit ReachabilityDist als Kriterium, damit immer Nachbarschaft „durchgrasen“, danach zufällig neuen Punkt suchen.

1.12.1 Expectation Maximation

- Initiale Parameter raten
- Für jedes Datenobjekt Clusterwahrscheinl. Ausrechnen (Formel)
- Parameter der Cluster neu berechnen.
- Terminierung prüfen mit OverallLikelyhood, sonst zurück zu 2

Overall Likelyhood: $\sum_i (p_a \cdot \Pr[x_i|A] + p_B \cdot \Pr[x_i|B]$

1.13 Abstandsbasierte Outlierdetection

Indexbasiert: Indexstruktur, k-NN-Query für jeden Datenpunkt.

Nested-loop: klar

Zellenbasiert: Mehr als 1-p Objekte in L_1: Keine Outlier in Zelle. Weniger als 1-p Objekte in L_2: Nur Outlier in Zelle.Zellenbreite von $D/2 \cdot \sqrt{\#Dimensionen}$

1.14 Dichtebasierte Outlierdetection

Local Reachability Density: $lrd(p) = \frac{1}{\frac{\sum_{p' \in NN} d(p,p')}{2}}$.

Local Outlier Factor: $lof(p) = \frac{\sum_{p' \in NN} lrd(p')}{|p' \in NN|}$

Verwende Clustering, um Großteil der Punkte zu prunen.

1.14.1 Micro-Cluster

Ähnlich zu BIRCH. Elementarcluster mit ~CF => Obere, untere Schranke für lof.

1.15 Objektbasiertes Vorgehen

Hohe Dimensionalität, suche interessante Dimensionen.

Score/Outlierness. Punkt kann in nur konkreter Menge von Dimensionen ein nicht-trivialer Outlier sein.

Sample Subspace-Pool: Zufällige Teilräume der Attribute, bestimme Outlierness für alle Samples, für interessante Samples bestimme so interessante Attribute.

1.16 Ensembles: Bagging

N Instanzen werden aus Trainingsdatenbestand gesampled, Auf Samples wird trainiert, Model wird jeweils gespeichert.

Klassifizierung: Klasse wird durch jedes Modell bestimmt, häufigste Klasse wird ausgegeben.

1.16.1 Metacost

Datenbestand wird gerelabled entsprechend Conditional Risk, dafür Bagging. Relableing: Datenobjekte werden neuen Klassen zugeordnet).

1.17 Ensembles: Boosting

Wie Bagging, aber gewichtet.

Iteriertes Vorgehen: Trainiere Classifier mit Trainingsdatenbestand und multipliziere Classifier-Gewicht mit Fehler, jeweils alle Fehler normalisieren.

Klassifizierung: Klassen werden für jede Instanz bestimmt, die Klasse mit höchstem Gewicht wird ausgegeben.

1.18 Stacking

Unterstützung verschiedener Modelle.

Level-0 Modelle: Ausgangsbasis.

Level-1 Modelle: Metamodell. Holdout-Set um Level0-Modelle auszubessern.

1.19 (Statistische Tests)

- Chi-Quadrat-Test
  - Sind zwei Verteilungen unabhängig voneinander?
- Kolmogorov Smirnov Test
  - Stimmen zwei Wahrscheinlichkeitsverteilungen überein?
- Wilcoxon Mann Whitney Test
  - Ist Abweichung der Mediane zweier Verteilungen statistisch Signifikant?