



UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR
THEORETISCHE INFORMATIK

Datenbankanonymisierung auf Basis von k-means-Algorithmen

Finn Stoldt

19.12.2019

IM FOCUS DAS LEBEN



Einleitung

NETFLIX

Netflix Prize dataset

- 2006 veröffentlicht
- pseudonymisierte Filmbewertungen (mit Datum)
- von 500.000 Abonnenten
- Re-Identifizierung über öffentliche IMDb möglich

- Pseudonymisierung garantiert keine Anonymität.
- Daten müssen *anonymisiert* werden.
- Orientieren uns an Datenschutzmodell *k-Anonymität*.
- Kann durch *Mikroaggregation* erreicht werden.

Grundlagen

Datenbank $\mathcal{X} \in \mathbb{R}^{n \times m}$ mit n Elementen mit m reellwertigen Quasi-Identifikatoren

i -tes Element repräsentiert durch Vektor $\mathbf{x}_i = (x_1, \dots, x_m)$

Abstand zwischen $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$

$$d(\mathbf{x}, \mathbf{x}') := \sum_{i=1}^m (x_i - x'_i)^2$$

Abstand zwischen $\mathcal{X}, \mathcal{X}' \in \mathbb{R}^{n \times m}$

$$\text{sse}(\mathcal{X}, \mathcal{X}') := \sum_{i=1}^n d(\mathbf{x}, \mathbf{x}')$$

Schwerpunkt von $\mathcal{X} \in \mathbb{R}^{n \times m}$

$$c(\mathcal{X}) := \frac{1}{|\mathcal{X}|} \sum_{\mathbf{x} \in \mathcal{X}} \mathbf{x}$$

Abstand zwischen allen $\mathbf{x} \in \mathcal{X}$ und $c(\mathcal{X})$

$$\text{sse}(\mathcal{X}) := \sum_{\mathbf{x} \in \mathcal{X}} d(\mathbf{x}, c(\mathcal{X}))$$

\mathcal{X} ist *k-anonym*, wenn jeder Vektor $\mathbf{x} \in \mathcal{X}$ mindestens k mal in \mathcal{X} vorkommt.

Ein *Anonymisierungsalgorithmus* μ ist eine Abbildung $\mu : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m}$,

$$\mu : \mathcal{X} := \mathbf{x}_1, \dots, \mathbf{x}_n \mapsto \hat{\mathcal{X}} := \hat{\mathbf{x}}_1, \dots, \hat{\mathbf{x}}_n$$

Anonymisierung von \mathcal{X} durch *Mikroaggregation*:

1. k -Clustering \mathcal{C} von \mathcal{X} erzeugen.
2. Für alle $C \in \mathcal{C}$: Ersetze alle $\mathbf{x} \in C \subset \mathcal{X}$ durch $c(C)$

Anonymisierung

Anonymisierungsverzerrung bei Anonymisierung von \mathcal{X} durch μ

$$D_{\mu}(\mathcal{X}) := \text{sse}(\mathcal{X}, \mu(\mathcal{X})) = \text{sse}(\mathcal{X}, \hat{\mathcal{X}})$$

Diversität von \mathcal{X}

$$\Delta(\mathcal{X}) := \text{sse}(\mathcal{X})$$

Informationsverlust bei Anonymisierung von \mathcal{X} durch μ

$$L_{\mu}(\mathcal{X}) := \frac{D_{\mu}(\mathcal{X})}{\Delta(\mathcal{X})}$$

Algorithmus \varkappa -means

Eingabe: Datenbank \mathcal{X} , Clusteranzahl \varkappa

Ausgabe: Partition $\mathcal{C} = C_1, \dots, C_{\varkappa}$ von \mathcal{X}

1. Wähle \varkappa beliebige Elemente aus \mathcal{X} als Mittelpunkte $M = \mathbf{m}_1, \dots, \mathbf{m}_{\varkappa}$.
 2. Ordne jedem Punkt $\mathbf{x} \in \mathcal{X}$ ein Cluster C_i mit $i \in \{1, \dots, \varkappa\}$ zu, für das $d(\mathbf{x}, \mathbf{m}_i)$ minimal ist.
 3. Aktualisiere \mathbf{m}_i für alle $i \in \{1, \dots, \varkappa\}$, sodass \mathbf{m}_i der Gruppenschwerpunkt aller Punkte in C_i ist: $\mathbf{m}_i = c(C_i)$.
 4. Wiederhole Schritt 2 und 3 solange, bis sich \mathcal{C} nicht mehr verändert.
-

kAnonyMeans

Algorithmus `kAnonyMeans`

Eingabe: Datenbank \mathcal{X} , Anonymitätsparameter k , Initiale Clusteranzahl \varkappa

Ausgabe: k -anonyme Datenbank $\hat{\mathcal{X}}$

1. Partitioniere die gegebene Datenbank \mathcal{X} mittels \varkappa -means in \varkappa Cluster.
 2. Verschmelze durch Merge jedes Cluster mit weniger als k Elementen mit anderen Clustern.
 3. Teile mittels Split jedes Cluster mit $2k$ oder mehr Elementen in kleinere auf.
-

Merge

Verschiedene Reihenfolgen durch unterschiedlichen Aufbau von Merge:

- mit innerer Schleife
- mit äußerer Schleife
- ohne weitere Schleife

Verschiedene Eignungsmaße für Verschmelzungskandidaten:

- Mittelpunktabstand
- sse-Zuwachs

k -Clusterings müssen erzeugt werden \Rightarrow MDAV⁺

Verwenden MDAV⁺ da

- geringe Clustervarianzen
- wenige Elemente

MergeAndSplit während k-means

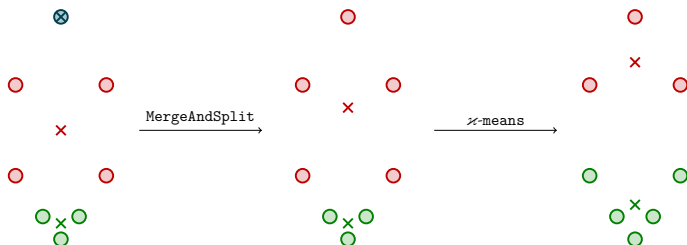


Abbildung: Beispiel-Situation für $k = 3$ in der es sinnvoll ist, nach MergeAndSplit wieder k -means auszuführen. Die Elemente des Datensatzes sind durch Kreise und die Clustermittelpunkte durch Kreuze dargestellt.

MergeAndSplit während k-means

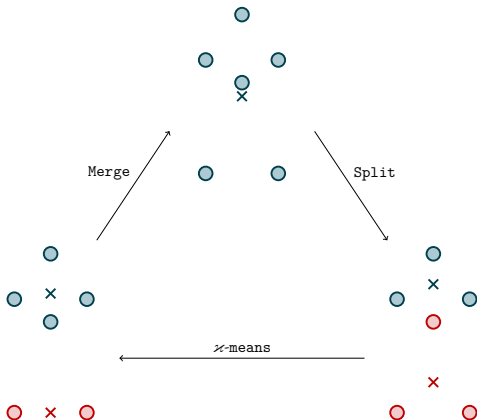


Abbildung: Dargestellt ist ein Beispiel einer Livelock-Situation verursacht durch Ausführung von MergeAndSplit während k -means für $k = 3$. Die Elemente des Datensatzes sind durch Kreise und die Clustermittelpunkte durch Kreuze dargestellt.

Algorithmus kAnonyMeans

Eingabe: Datenbank \mathcal{X} , Anonymitätsparameter k , Initiale Clusteranzahl κ

Ausgabe: k -anonyme Datenbank $\hat{\mathcal{X}}$

1. Partitioniere die gegebene Datenbank \mathcal{X} mittels κ -means in κ Cluster.
 2. Verschmelze durch Merge jedes Cluster mit weniger als k Elementen mit anderen Clustern.
 3. Teile mittels Split jedes Cluster mit $2k$ oder mehr Elementen in kleinere auf.
-

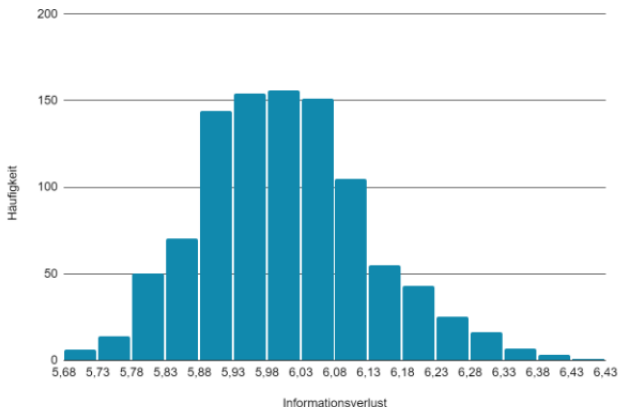


Abbildung: Histogramm der in 1.000 Durchläufen entstandenen Informationsverluste (in Prozent) bei Anonymisierung des Benchmarkdatensatzes CENSUS durch **kAnonyMeans** mit Standardkonfiguration und $k = 3$.

kAnonyMeans*

- Zufallsfaktor: inititiale Mittelpunkte
- „Je besser die Mittelpunkte, desto niedriger der Informationsverlust“
- Mittelpunkte evolutionär entwickeln

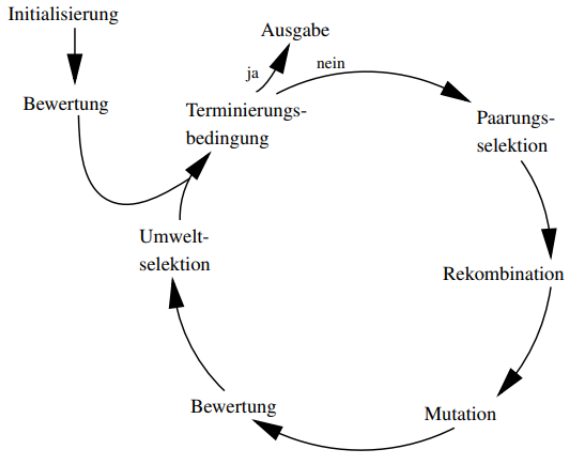


Abbildung: Schematische Darstellung des Zyklus bei evolutionären Algorithmen.

Algorithmus kAnonyMeans*

Eingabe: Datenbank \mathcal{X} , Populationsgröße p , Überlebende s , Mutationen m_c , Mutationsstärke m_s

Ausgabe: Anonymisierte Datenbank $\hat{\mathcal{X}}$

Erzeuge Startpopulation $\mathcal{M} = M_1, \dots, M_p$ mit $M \subseteq \mathcal{X}$ nach Forgy- oder κ -means++-Methode;

Berechne $L_{\text{kAnonyMeans}_M}(\mathcal{X})$ für alle $M \in \mathcal{M}$;

Solange \neg Abbruchbedingung

Sortiere \mathcal{M} aufsteigend nach $L_{\text{kAnonyMeans}_M}(\mathcal{X})$ mit $M \in \mathcal{M}$;

Entferne alle Individuen M_{s+1} bis M_p , sodass die s besten Individuen überleben;

Erzeuge aus den überlebenden Individuen (Eltern) $p - s$ neue Individuen (Kinder)
und füge diese an \mathcal{M} an;

Tausche von m_c Kindern jeweils m_s zufällige Mittelpunkte durch zufällige Elemente aus \mathcal{X} aus;

Berechne $L_{\text{kAnonyMeans}_M}(\mathcal{X})$ für alle Kinder $M \in \{M_{s+1}, \dots, M_p\} \in \mathcal{M}$;

Sortiere \mathcal{M} aufsteigend nach $L_{\text{kAnonyMeans}_M}(\mathcal{X})$ mit $M \in \mathcal{M}$;

Berechne $\text{kAnonyMeans}_{M_1}(\mathcal{X})$;

Verwendete Datensätze

- Etablierte Benchmark-Datensätze
 - CENSUS
 - EIA
 - TARRAGONA
- Synthetisch generierte Datensätze
 - SimU
 - SimC
- Sonstige Datensätze
 - Cloud1
 - Cloud2

Ergebnisse

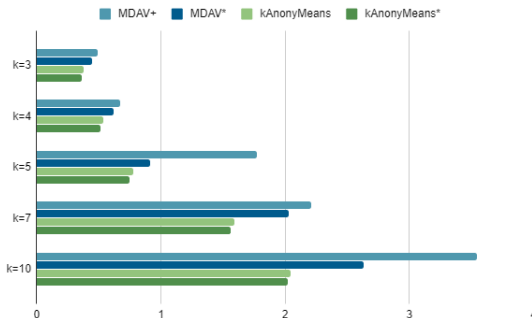


Abbildung: Informationsverluste (in Prozent) bei Anonymisierung des EIA Datensatzes durch verschiedene Algorithmen für verschiedene k .

Ergebnisse

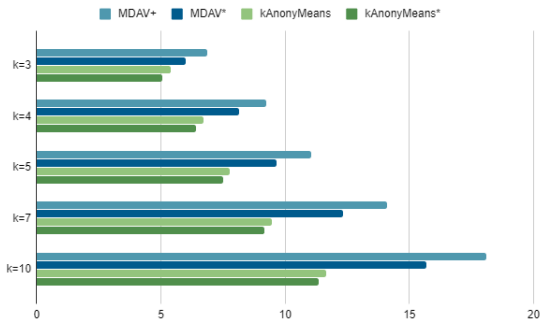


Abbildung: Informationsverluste (in Prozent) bei Anonymisierung des SimC Datensatzes durch verschiedene Algorithmen für verschiedene k .

Zusammenfassung und Ausblick

- kAnonyMeans* im Schnitt 17,4% besser als MDAV⁺
- kAnonyMeans* im Schnitt 11,6% besser als MDAV*
- kAnonyMeans* im Schnitt 3% besser als kAnonyMeans
- Stark auf geclusterten Daten
- Je größer k , desto stärker kAnonyMeans

- Zufallsabhängigkeiten minimieren
- optimale Parameterbelegung finden
- bessere Rekombinations- / Mutationsstrategien

Ausblick - Mikroaggregation

- Optimaler Algorithmus
- besserer Approximationsalgorithmus als $\mathcal{O}(k^3)$
- Linearzeitalgorithmus
- Mikroaggregation durch evolutionäre Strategie

Danke für Ihre Aufmerksamkeit!

Algorithmus MDAV^+

Eingabe: Datenbank \mathcal{X} , Anonymitätsparameter k

Ausgabe: k -anonyme Datenbank $\hat{\mathcal{X}}$

1. Berechne den Schwerpunkt $c(\mathcal{X})$ der Eingabedatenmenge \mathcal{X} .
 2. Wähle jenen noch nicht zugewiesenen Eintrag $\mathbf{x} \in \mathcal{X}$, der am weitesten von $c(\mathcal{X})$ entfernt ist.
 3. Bilde eine Gruppe um \mathbf{x} bestehend aus \mathbf{x} und seinen $k - 1$ nächsten nicht zugewiesenen Nachbarn (diese Elemente sind nun zugewiesen).
 4. Wenn mindestens k nicht zugewiesene Einträge übrig sind, gehe zurück zu Schritt 2, andernfalls weise alle noch nicht zugewiesenen Elemente der zu ihnen nächsten Gruppe zu.
-

Auswahl der initialen Mittelpunkte

- Forgy
- \mathcal{K} -means++

Laufzeit und Informationsverlust bei kAnonyMeans

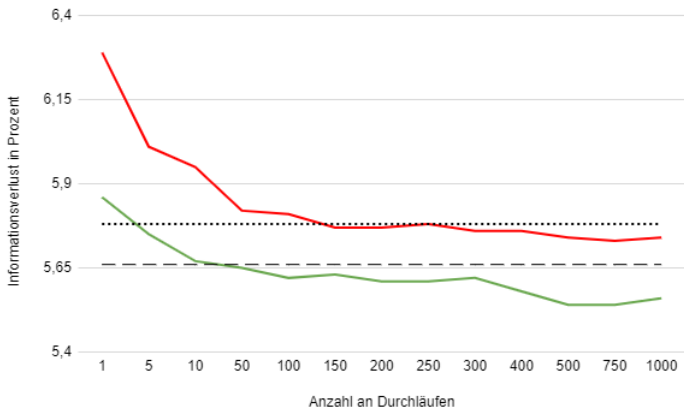


Abbildung: Minimaler (grüne Linie) und maximaler (rote Linie) Informationsverlust (in Prozent) bei fünfzigmal durchgeführter Anonymisierung des CENSUS-Datensatzes mittels kAnonyMeans für $k = 3$ bei verschiedener Anzahl an Durchläufen. Die gepunktete Linie gibt zum Vergleich den Informationsverlust bei Verwendung von MDAV* an, die gestrichelte Linie den von MDAV+.

Laufzeit und Informationsverlust bei kAnonyMeans*

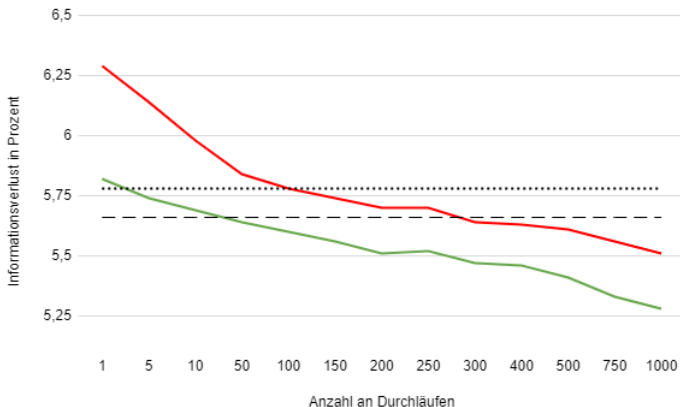


Abbildung: Minimaler (grüne Linie) und maximaler (rote Linie) Informationsverlust (in Prozent) bei fünfzigmal durchgeführter Anonymisierung des CENSUS-Datensatzes mittels kAnonyMeans* für $k = 3$ bei verschiedener Anzahl an Durchläufen. Die gepunktete Linie gibt zum Vergleich den Informationsverlust bei Verwendung von MDAV* an, die gestrichelte Linie den von MDAV+.

References I