Seminar Duplikaterkennung

Themenvorstellung

Sascha Szott

(FG Informationssysteme, HPI)

16. April 2008

Themenvorstellung

1. Yan et al.: Adaptive Sorted Neighborhood Methods for Efficient Record Linkage (dazu: Hernandez, Stolfo: Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem)
2. Chaudhuri et al.: Example-driven Design of Efficient Record Matching Queries
3. Chaudhuri et al.: Robust Identification of Fuzzy Duplicates
4. Ananthakrishna et al.: Eliminating Fuzzy Duplicates in Data Warehouses
5. Bilenko, Mooney: Adaptive Duplicate Detection Using Learnable String Similarity Measures
6. Bilenko et al.: Adaptive Blocking: Learning to Scale Up Record Linkage
7. Sarawagi, Bhamidipaty: Interactive Deduplication using Active Learning
8. Chaudhuri et al.: Leveraging Aggregate Constraints For Deduplication

❶ Hernandez, Stolfo: Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem

Sorted Neighborhood Method (SNM)

- Eingabe: Menge von Tupeln (Records), Eingabegröße *n*
- drei Phasen

1. **Schlüsselerzeugung**: Schlüsseldefinition und Zuweisung eines Schlüssels für jeden Datensatz
2. lexikographische **Sortierung** der so erhaltenen Schlüssel
3. Verschieben des Fensters „von links nach rechts“ (bzgl. der Ordnung)
    - **Vergleiche** nur die Tupel, die sich zu einem Zeitpunkt innerhalb des Fensters (der Breite *w*) befinden
    - Nutzung einer **Ähnlichkeitsfunktion**, um für ein Tupelpaar zu entscheiden, ob die beiden Tupel Duplikate darstellen oder nicht

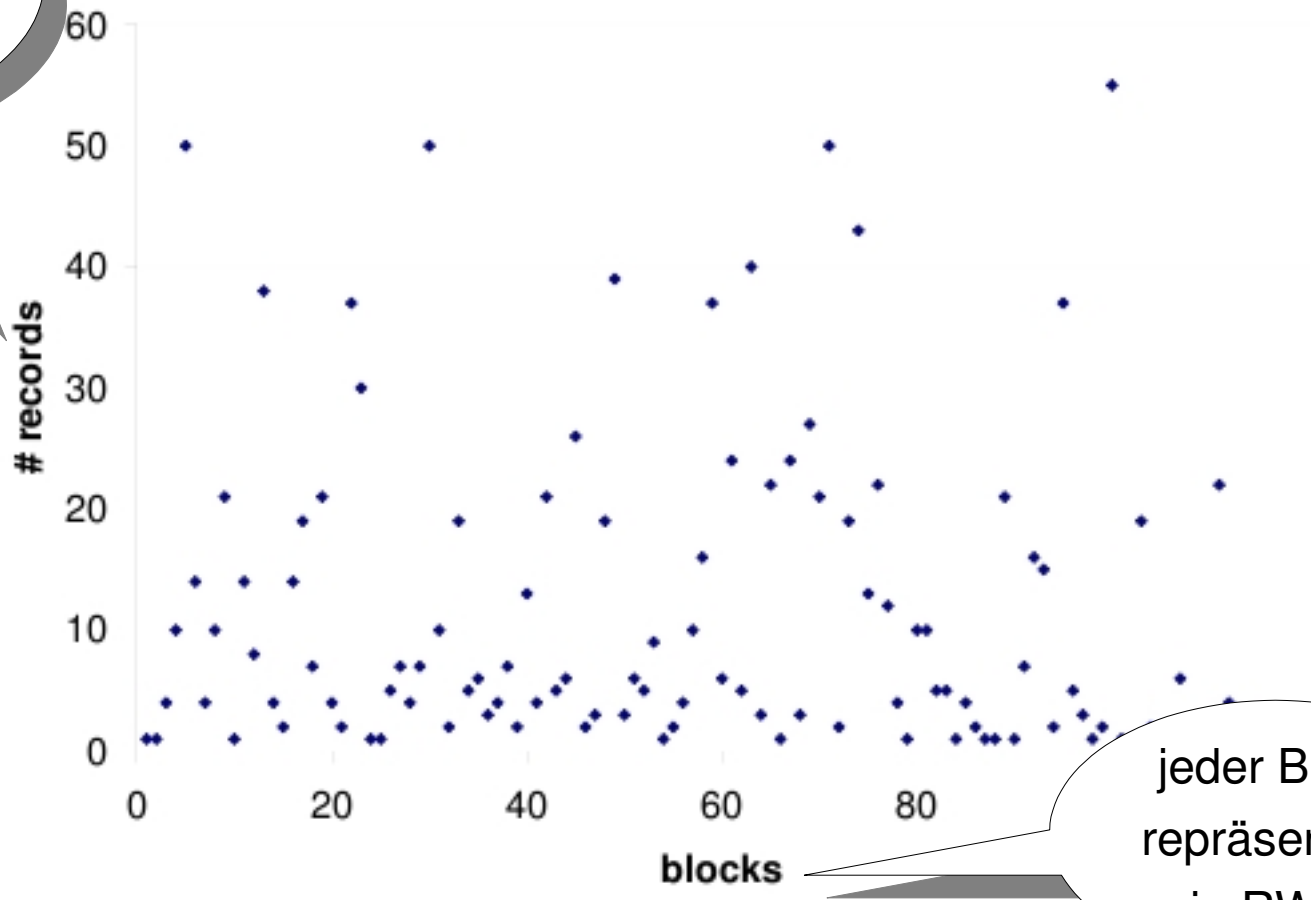
❶ Yan et al.: Adaptive Sorted Neighborhood Methods for Efficient Record Linkage

Beobachtungen bei klassischer SNM

- Fenstergröße *w* beeinflußt
  - Effektivität (*w* möglichst groß)
  - Effizient (*w* möglichst klein)
- Wahl einer geeigneten Fenstergröße schwierig (Expertenwissen nötig)
- für gewöhnlich gibt es auch nicht *die* optimale Fenstergröße

❶ Yan et al.: Adaptive Sorted Neighborhood Methods for Efficient Record Linkage

entspricht der optimalen Fenstergröße*



jeder Block repräsentiert ein RWE**

* unter der Annahme, dass Tupel eines Blocks in der Sortierung direkt aufeinander folgen

** Real-Welt Entität (*real world entity*)

Figure 2: Ideal block sizes in the Cora dataset (116 blocks, alphabetically ordered).

❶ Yan et al.: Adaptive Sorted Neighborhood Methods for Efficient Record Linkage

Ausweg: adaptive und dynamische Anpassung der Fenstergröße zur Laufzeit

Umformung des Problems:

- Erzeuge eine **disjunkte** Partitionierung der sortierten Liste
  - bei klassischer SNM: überlappende Partitionierung (gleicher Größe)
  - jetzt: Partitionen können unterschiedlich groß sein
- dafür notwendig: Bestimmung der Partitionsgrenzen
  - Vorstellung von zwei Techniken, die weniger als $n - 1$ Vergleiche erfordern
- in Phase 3 dann nur noch Vergleiche innerhalb der Partitionen

❷ Chaudhuri et al.: Example-driven Design of Efficient Record Matching Queries

Standpunkt: Record Matching kann als Anfrage betrachtet werden

- aber: Anfrageformulierung ist schwierig (und domänenabhängig)
- erstellen eine *initiale* Record Matching-Anfrage
  - Vorgehen: Benutzer spezifiziert für einige Tupelpaare, ob Duplikateigenschaft vorliegt oder nicht
  - Erzeugung der Anfrage mittels *machine learning*-Techniken
  - anschließende Verfeinerung durch Programmierer möglich
- Qualitätskriterien
  - effizient: Ausnutzung von (hochoptimierten) DBMS-Operatoren
  - verständlich: ermöglicht die nachträgliche Anpassung

❷ Chaudhuri et al.: Example-driven Design of Efficient Record Matching Queries

Enquiries: **R**

ID	Name	Address	City	State	Zip	Mothers Name	Fathers Name	Phone
1	Gail Smith	10 Main St	Chicago	Illinois	60602	Mary Smith	*NULL*	163-1234
2	Kevin J	#1344 Mont Ave	*NULL*	Wisc	53593	Stephanie Joule		608-234-0098
3	Sandra C	#245 First Ave		Texus		NULL	Charles Calvin	332-1288
...	...	...	...	...	...	...	...	...

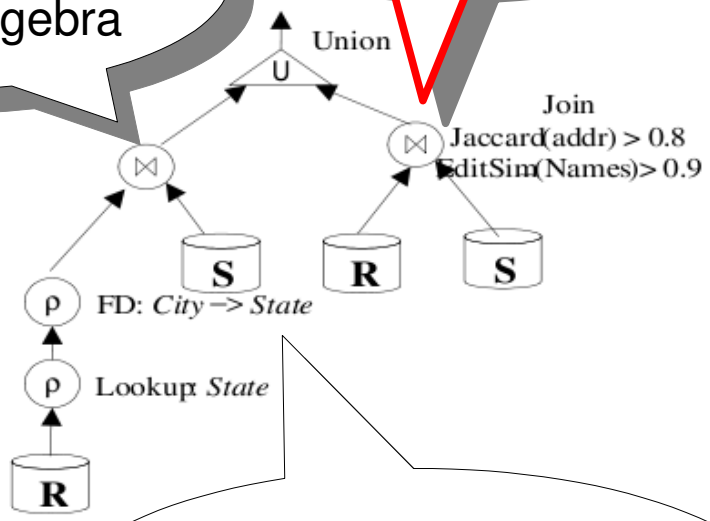
Evacuees: **S**

ID	Name	Address	City	State	Zip	Mothers Name	Fathers Name	Phone
1	Gershwin K	35 Forest Dr	NULL	WI	53593	Georgia K	Ben Kirsten	608-1123445
2	Mary Green	24 Second Ave	Verona	WI	NULL	Emma Green	Anthony Green	
3	Ted Johnson	412 Madison St	Verona	NULL	53593	Olivia J	Ethan Johnson	
4	C. Larson	18 Main St	Fitchburg	WI	53593	Ashley Larson	Michael Larson	
5	G. Smith	135 Dayton St	NULL	WA	98052	Mary Carlton	Tom Smith	234-0098
6	G Smith	Main Street	Chicago	IL	60608	M Smyth	Bart Smith	...

Figure 1: A record matching example & query

Operatoren der Relationenalgebra

similarity join



attribute value transformations (Referenzlisten, FAen)

Join predicate	Precision	Recall
Baseline—any 1 similarity predicate	>95%	0%
Jaccard(Names, FatherName) > 0.4 && Jaccard(Address) > 0.7	>95%	11%
Jaccard(Names, FatherName) >= 1.0 && Jaccard(Phone) > 0.5	>95%	30%

Figure 2: Example queries & accuracy

sich ergebende Recallwerte unter der Bedingung, dass Precision > 0.95

❸ Chaudhuri et al.: Robust Identification of Fuzzy Duplicates

- vorherige Technik: supervised learning
- Probleme:
  - hoher Aufwand für die Erstellung der Trainingsdaten
  - Subjektivität von Entscheidungen
  - ausreichende Repräsentativität der Trainingsdaten
- „Abhilfe“: unsupervised learning
  - Nutzung von Clustering-Algorithmen zur Partitionierung der Eingaberelation (jedes Cluster repräsentiert ein RWE)
  - anschließend werden nur noch die Tupel innerhalb einer Partition auf Duplikateigenschaft hin untersucht (Ähnlichkeitsmaß)

❸ Chaudhuri et al.: Robust Identification of Fuzzy Duplicates

Unzulänglichkeit bei der Nutzung von Clustering-Algorithmen

- für das Identifizieren von Duplikatgruppen zusätzlich Betrachtung von *lokalen Struktureigenschaften* wichtig
- Autoren stellen zwei neue Kriterien vor, die lokale Struktureigenschaften von Daten erfassen
  - **compact set**
    Tupel innerhalb einer Duplikatgruppe sind *näher* zueinander als zu Tupeln außerhalb der Gruppe (bzgl. eines gegebenen Maßes)
  - **sparse neighborhood**
    die lokale *Nachbarschaft* der Gruppentupel ist leer oder dünn besetzt

❸ Chaudhuri et al.: Robust Identification of Fuzzy Duplicates

Ziele:

- Tupel, die Kriterien erfüllen, können dennoch innerhalb einer Gruppe liegen, obwohl sie nur geringe Ähnlichkeit aufweisen (weniger falsche Negative)

- Tupel, die Kriterien nicht erfüllen, können in unterschiedlichen Gruppen eingeteilt werden, obwohl sie eine große Ähnlichkeit aufweisen (weniger falsche Positive)

ID	ArtistName	TrackName
1*	The Doors	LA Woman
2*	Doors	LA Woman
3*	The Beatles	A Little Help from My Friends
4*	Beatles, The	With A Little Help From My Friend
5*	Shania Twain	Im Holdin on to Love
6*	Twian, Shania	I'm Holding On To Love
7	4th Elemynt	Ears/Eyes
8	4th Elemynt	Ears/Eyes - Part II
9	4th Elemynt	Ears/Eyes - Part III
10	4th Elemynt	Ears/Eyes - Part IV
11	Aaliyah	Are You Ready
12	AC DC	Are You Ready
13	Bob Dylan	Are You Ready
14	Creed	Are You Ready

Table 1: Examples from a media database.

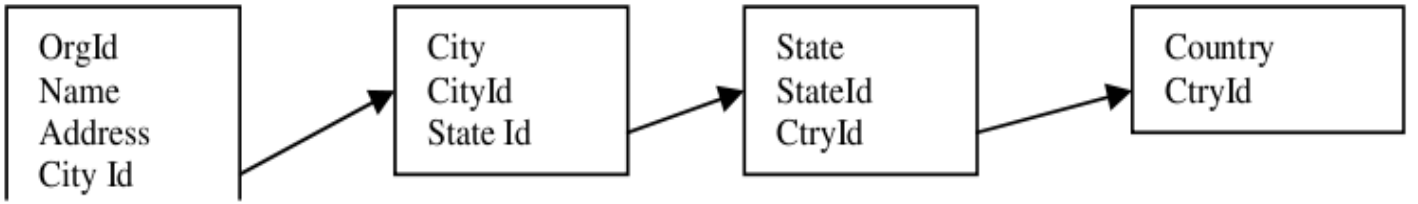
Ähnlichkeitsfunktion basierend auf der Levenshtein-Distanz

*Duplikate: {1, 2}, {3, 4} und {5, 6}

❹ Ananthakrishna et al.: Eliminating Fuzzy Duplicates in Data Warehouses

DELPHI: **D**uplicate **El**imination in the **P**resence of **Hi**erarchies

- Duplikaterkennung in Dimensionstabellen (in Data Warehouses)
  → Ausnutzung von *dimensionalen Hierarchien*



OrgId	Name	Address	CityId
O1	Clintstone Assoc.	#1, Lake View Blvd.	C1
O2	Compuware	#20, Main Street	C2
O3	Compuwar	#20, Main Street	C3
O4	Clintstone Associates	#1, Lake View	C4
O5	Ideology Corp.	#10, Vancouver Pl.	C5
O6	Victoria Films	#5, Victoria Av.	C6
O7	Ideology Corporation	#10, Vanc. Pl.	C7
O8	Clark Consultants Ltd.	#8, Cherry Street	C8
O9	Clark Consultants	#8, Cherr St.	C9

Organization (at Level 1)

CityId	City	StateId
C1	Joplin	S1
C2	Jopin	S2
C3	Joplin	S4
C4	Joplin	S3
C5	Victoria	S5
C6	Victoria	S6
C7	Vancouver	S5
C8	Aberdeen	S7
C9	Aberdeen	S8

City (at Level 2)

StateId	State	CtryId
S1	MO	1
S2	MO	2
S3	MO	3
S4	Missouri	3
S5	BC	4
S6	British Columbia	4
S7	Aberdeen shire	5
S8	Aberdeen	5

State (at Level 3)

CtryId	Country
1	United States of America
2	United States
3	USA
4	Canada
5	UK

Country (at Level 4)

Figure 1: An Example Customer Database

❹ Ananthakrishna et al.: Eliminating Fuzzy Duplicates in Data Warehouses

- weisen jeder CtryID *c* die Menge *s*(*c*) der StateIDs zu, die auf diese CtryID verweisen
- für zwei CtryIDs c_1 und c_2 gilt: je größer die Schnittmenge von $s(c_1)$ und $s(c_2)$, umso wahrscheinlicher, dass c_1 und c_2 Duplikate sind
- auch Ausschluss von falschen Positiven möglich
  - Edit-Distance zwischen USA und UK gering, aber *s*('USA') und *s*('UK') besitzen keine gleichen Elemente
  - **Folge:** USA und UK werden nicht als Duplikate angesehen

StateId	State	CtryId
S1	MO	1
S2	MO	2
S3	MO	3
S4	Missouri	3
S5	BC	4
S6	British Columbia	4
S7	Aberdeen shire	5
S8	Aberdeen	5

CtryId	Country
1	United States of America
2	United States
3	USA
4	Canada
5	UK

❹ Ananthakrishna et al.: Eliminating Fuzzy Duplicates in Data Warehouses

Beobachtung: SNM für bestimmte Szenarien ungeeignet

- z. B. Erkennung des Duplikatpaars (UK, Great Britain) im vorherigen Beispiel
- **aber:** wir wollen auch nicht alle Paare miteinander vergleichen (quadratische Laufzeit)
- daher: Verwendung einer **Gruppierungsstrategie** unter Ausnutzung der dimensionalen Hierarchie
  - z. B. Vergleiche zwei Tupel aus Relation *State* nur dann, wenn sie
    - gleiches Tupel in der Relation *Country* referenzieren oder
    - zwei Tupel in *Country* referenzieren, die bereits als Duplikate erkannt wurden
  - paarweise Vergleiche von Tupeln nur innerhalb der Gruppen

❺ Bilenko, Mooney: Adaptive Duplicate Detection Using Learnable String Similarity Measures

- **Probleme** beim Einsatz klassischer Ähnlichkeitsfunktionen für Duplikaterkennung
  - Ähnlichkeit ist *domänenabhängig* (Bsp.: Papier vs. Artikel)
  - Ähnlichkeit ist *attributabhängig* (Bsp.: Nachname vs. Vorname)
- **Idee**: Nutzung von *trainierbaren* Maßen für den Ähnlichkeitsvergleich von Zeichenketten (für jedes Attribut in Bezug auf eine vorgegebene Domäne)
- Vorstellung von zwei Maßen
  - Erweiterung einer erlernbaren affine-gap Edit-Distance
  - Vektorraum-basiertes Maß, das Support Vector Machine zum Training verwendet

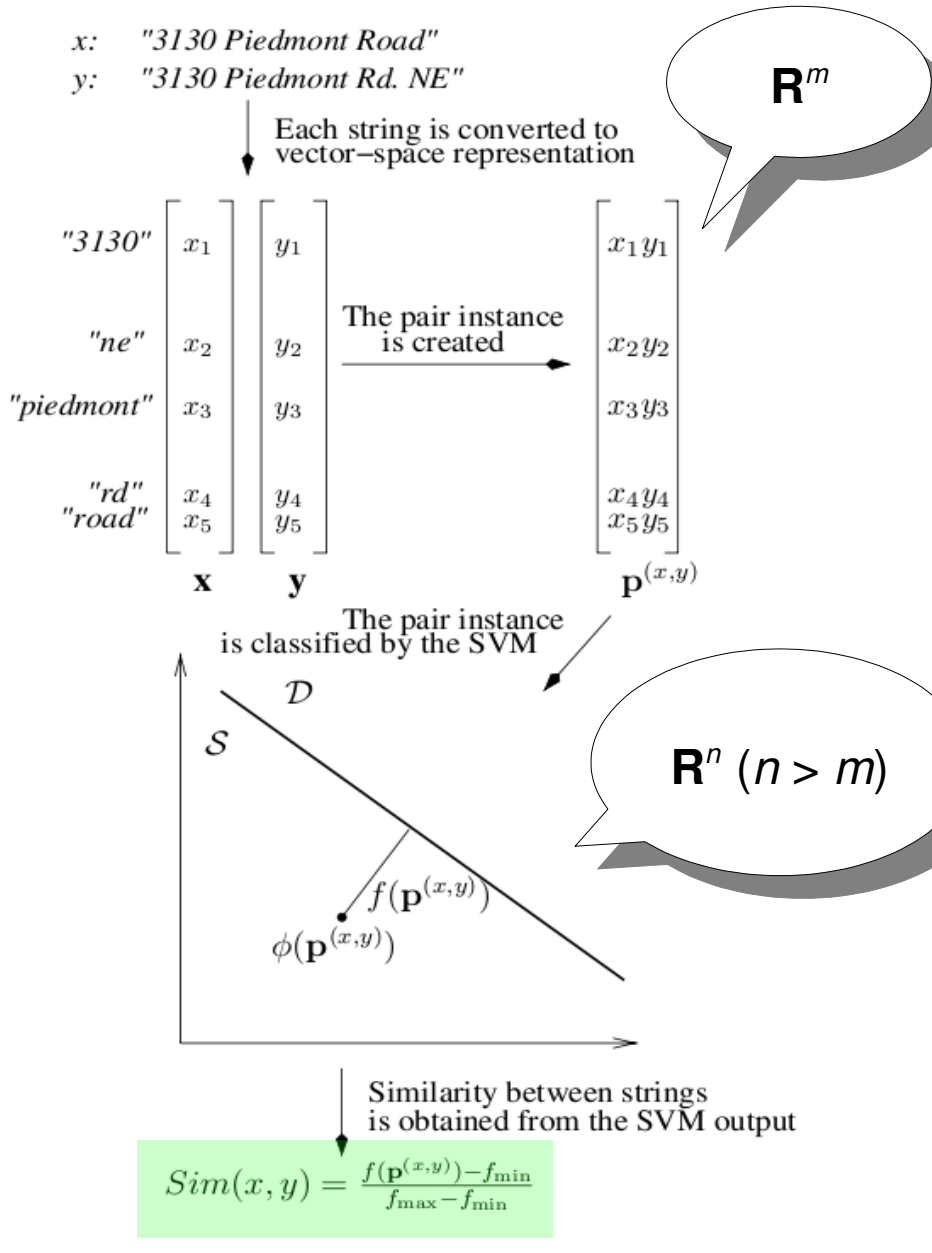
... für einzelne Wörter

... für textwertige Attribute

❺ Bilenko, Mooney: Adaptive Duplicate Detection Using Learnable String Similarity Measures

Vorgehen

- Lernen für jedes Attribut der Eingangsrelation ein Ähnlichkeitsmaß
- anschließend wird ein Prädikat *isDuplicate* unter Nutzung dieser attributbasierten Ähnlichkeitsmaße erlernt
  - dazu wird ebenfalls SVM genutzt
  - Entscheidung über Duplikateigenschaft zweier Tupel



❻ Bilenko et al.: Adaptive Blocking: Learning to Scale Up Record Linkage

- Blockingmethoden erlauben die Reduktion der ursprünglichen Problemgröße von $O(n^2)$
- Beispiele
  - SNM (Blocking implizit durch Schlüsseldefinition und Fensterbreite)
  - disjunkte Partitionierung
- dadurch: Skalierbarkeit / Effizienz gesichert (n kann sehr groß werden)
- Probleme bei bekannten Blockingmethoden
  - domänenabhängig
  - Konstruktion und Feintuning ist nicht-trivial und teilweise sehr aufwendig

❻ Bilenko et al.: Adaptive Blocking: Learning to Scale Up Record Linkage

- nicht optimale Blockingmethode führt zu
  - schlechter **Effektivität**, wenn Tupelpaare mit hoher Ähnlichkeit durch Reduktion aus dem Suchraum entfernt werden (Blocking zu streng)
  - schlechter **Effizienz**, wenn Tupelpaare mit niedriger Ähnlichkeit durch Reduktion nicht aus dem Suchraum entfernt werden (Blocking zu tolerant)
- **Idee**: Erlernen von **optimalen Blockingmethoden**
  - Finden einer optimalen Blockingmethode kann als Instanz des Rot-Blau Mengenüberdeckungsproblems (*red-blue set cover problem*) betrachtet werden
  - ***NP***-schweres Problem: Entwicklung eines Approximationsalgorithmus

❼ Sarawagi, Bhamidipaty: Interactive Deduplication using Active Learning

- Schwierigkeit beim *überwachten* Lernen
  - Trainingsdaten müssen repräsentativ sein: große Abdeckung (Spezialfälle, schwierige Fälle)
- Abhilfe: *aktives* Lernen
  - überwachtes Lernen: statischer Trainingsdatensatz
    - Menge von Paaren der Form $[(t_1, t_2), x]$ mit x = 'yes' oder x = 'no'
  - aktives Lernen:
    - Auswahl einer Menge von Paaren (t_1, t_2) durch Lernalgorithmus
    - Auswahl der Paare nach größtem Informationsnutzen
    - Benutzer entscheidet anschließend für jedes Paar (t_1, t_2), ob Duplikate vorliegen oder nicht

einfach

schwierig

❼ Sarawagi, Bhamidipaty: Interactive Deduplication using Active Learning

Vorgehen

- gleichzeitige Konstruktion verschiedener redundanter Funktionen
- Ausnutzung der Unstimmigkeiten zwischen diesen Funktionen, um neue Inkonsistenzen in Datensatz aufzuspüren
- **Resultat:** *deduplication function* (Klassifikator)

❽ Chaudhuri et al.: Leveraging Aggregate Constraints For Deduplication

- Ausnutzung von *Aggregations-Constraints* (ACs) für Duplikaterkennung
- Annahme: Duplikate meist nicht exakt, sondern nur ungefähr gleich sind
- Folge: Verstöße gegen ACs
- Ausnutzung von ACs für Duplikaterkennung
  - Ziel: Verringerung der Anzahl solcher Verstöße (nach der Duplikateliminierung)
  - schließlich: Ausnutzung von erfüllten ACs für das Zusammenführen von Duplikaten (Erfüllung einer vorher verletzten AC als Indiz für Duplikateigenschaft)
  - Relaxierung in der Praxis: nicht exakte Erfüllung von ACs, sondern auch beinahe Erfüllung (*maximum constraint satisfaction*)

❽ Chaudhuri et al.: Leveraging Aggregate Constraints For Deduplication

Beispiele

berechnet

Member	Fees stored	Fees derived
John Doe	100	130
J. Doe	40	10
...	...	...

berechnet

Member	Fees stored	Fees derived
John Doe	100	100
J. Doe	40	10
...	...	...

AC: SUM(Fees_stored) = SUM(Fees_derived)

- durch Deduplikation wird AC erfüllt
- **Konklusion**: Duplikatpaar

- erstes Tupel erfüllt AC
- durch Deduplikation würden wir AC verletzen
- **Konklusion**: kein Duplikatpaar