FACHHOCHSCHULE
ERFURT UNIVERSITY
OF APPLIED SCIENCES
Angewandte
Informatik

Masterarbeit
in der Angewandten Informatik
Nr. AI-2021-MA-014

Generisches Clustern hoch-komplexer Produktdaten

Hannes Dröse

Abgabedatum: 29.04.2022

Prof. Dr. Ines-Kerstin Rossak
Dipl.-Inform. Robert Queck

Zusammenfassung

Die Kurzfassung wird zumeist als letzter Abschnitt geschrieben. Sie soll auf einer Seite den Hintergrund bzw. die Motivation sowie die zentralen Ergebnisse der Arbeit zusammenfassen. Der Zweck dieses Texts ist es, einem Recherchierenden oder Informationssuchenden zu einem bestimmten Thema zu signalisieren, ob er in diesem Werk für ihn relevante Informationen finden wird: Lohnt es sich, die Arbeit zu bestellen oder zu kaufen? Was ist neu?
Die Kurzfassung ist das sichtbare Aushängeschild der Arbeit und dient der Aufnahme in Referenzdatenbanken, in Online-Literatur-Shops u. Ä. Schauen Sie sich die Abstracts in der Fachliteratur (etwa in Fachbüchern, am Anfang von Zeitschriftenartikeln, in Konferenz- Proceedings, in ACM- oder IEEE-Literaturdatenbanken, im OPAC der Bibliothek usw.) an, um ein Gefühl dafür zu entwickeln, was hineingehört (siehe Abschnitt 2.4 Materialsammlung, S. 2).
Was definitiv nicht hineingehört, sind Gliederungsübersichten, chronologische Arbeitsberichte u. dgl.

Abstract

The abstract has to be provided in english as well.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Hintergrund

Die adesso SE ist ein deutsches IT-Beratungs- und Dienstleistungsunternehmen. Sie ist 1997 gegründet worden mit Hauptsitz in Dortmund. Seit der Gründung ist die Firma sehr kontinuierlich gewachsen. Mittlerweile sind über 5800 Mitarbeiter an 44 Standort in 10 verschiedenen europäischen Ländern hier beschäftigt.

Als seine Mission versteht das Unternehmen: “Kerngeschäftsprozesse optimieren durch gezielten Einsatz moderner IT”. Es ist in diversen Branchen mit IT-Consulting und -Lösungen etabliert. Dies sind z.B. die Automobil-, Versicherungs- oder Energiebranche. adesso teilt sich selbst dazu in sog. LoBs (Lines of Business) ein. Die LoB Automotive ist also die Abteilung, die sich mit Consulting und Entwicklung von IT für die Automobil-Branche beschäftigt.

Über die Jahre sind nicht nur die Mitarbeiterzahlen gewachsen, sondern auch die abgedeckten Branchen, in denen das Unternehmen tätig ist. In den letzten Jahren sind zunehmend Projekte im E-Commerce-Sektor umgesetzt worden, allerdings ohne eine zugehörige dedizierte LoB. Mit Beginn von 2022 hat sich dies nun geändert und die LoB Retail wurde ins Leben gerufen. adesso möchte also fortan das Thema E-Commerce bearbeiten und vorantreiben.

In diesen Bereich fällt die Programmierung und Betreuung von Online-Shops sowie das umfangreiche Thema des Product-Information-Managements (PIM). Dabei geht es um das Verwalten und Aufbereiten von Produktdaten für verschiedene Anwendungen wie Warenhaltung, Marketing oder Bestellabwicklung. adesso erwägt in Zukunft ein eigenes Product-Information-Management-System zu entwickeln. In Vorbereitung dessen steht die Evaluierung der Machbarkeit verschiedener Ansätze rund um das Product-Information-Management an. Dies ist auch der Aufhänger für diese Masterarbeit.

1.2 Themenfindung

Eine Idee zum Product-Information-Management, die im Raum steht, ist die Nutzung von Verfahren der Clusteranalyse. Dabei handelt es sich um die Einteilung von Objekten in Gruppen (sog. Cluster), sodass sich die Objekte in der gleichen Gruppe ähnlicher sind als Objekte in den anderen Gruppen. [vgl. [King15]; Kap. 1.1 What Is a Cluster?]

In einem ersten Brainstorming sind verschiedene Anwendungen der Clusteranalyse diskutiert worden. Es folgte eine intensive Literaturrecherche zu den besprochenen Anwendungen, die im Folgenden kurz diskutiert werden.

TODO

- Auto Kategorisierung => Quellen suchen und recherchieren
- recommednations => bsp Paper und Anwendungen anreißen
- suche => [KoLo12]
- parallele Recherche in Akeneo-PIM als typisches PIM-System (open source) für Datenstrukturen

1.3 Kernfrage und Ablauf

Parallel zu der beschriebenen Recherche ist ebenfalls ein Blick in typische PIM-System wie z.B. Akeneo-PIM geworfen worden. Ziel war es zu verstehen, wie die Produktdaten in solchen System typischerweise vorliegen. Zum einen weisen sie einen hohen Grad an Struktur auf. Zum anderen kommen sehr verschiedenartige Datentypen vor. Die klassische Clusteranalyse arbeitet allerdings nur mit numerischen Daten. Somit war abzusehen, dass das Clustern selbst kein einfacher Prozess sein wird.

Dadurch verschob sich der Fokus von möglichen Anwendungen der Clusteranalyse von Produktdaten zu der neuen Kernfrage:

Ist das Clustern hoch-komplexer Produktdaten überhaupt möglich und wenn ja, wie?

Diese Fragestellung war der Ausgangspunkt weiterer Recherchen, Experimente und Versuche. Sie werden in den folgenden Kapiteln erläutert.

Kapitel 2 gibt einen Überblick über die wissenschaftlichen Grundlagen und Erkenntnisse des Themenfeldes.

Auf dieser Basis wird in *Kapitel 3* eine Konzeption dargelegt, wie das Clustern durchgeführt werden kann. Ebenso werden Versuche geplant, die zur Überprüfung des Konzeptes dienen.

Kapitel 4 beschreibt die Umsetzung des dargelegten Konzeptes. Thema dabei sind die Implementierung nötiger Software-Komponenten, Tools und Techniken, die zur Überprüfung der Fragestellung angewendet worden sind.

Die Auswertung und Evaluation der Versuche erfolgt in *Kapitel 5*.

Im finalen *Kapitel 6* wird das Fazit gezogen, ob die Kernfrage beantwortet werden kann. Ebenso wird ein Ausblick für weitere Versuche und Fragestellungen in der Zukunft gegeben.

2 Clusteranalyse

2.1 Begriff und Einordnung

maschinelles Lernen (machine learning)

unüberwachtes Lernen (unsupervised learning)

Clusteranalyse (cluster analysis)

Vorgehen: CRISP Modell

2.2 Distanz- und Ähnlichkeitsmaße

2.2.1 Definition

Clustering erfolgt über Bestimmung der "Nähe" (engl. proximity) der Objekte zueinander. [vgl. [KaRo09]; Kap. 1.2 Types of Data and How to Handle Them]

Notation:

- Objekte oder Datenpunkte sind Vektoren aus numerischen und/oder nominalen Attributen: $x = (i|i \text{ is nominal or numerical attribute})$
- Zugriff auf Attribut-Wert eines Datenpunkts über Attribut-Index: x_i
- Menge aus Datenpunkten (= Datenset) mit Großbuchstaben angegeben z.B. $x \in X$ oder $y \in Y$
- Menge N sind stets alle Datenpunkte also gesamtes Datenset
- mit N_i wir die Menge aller Werte eines Attributs im Datenset beschrieben
- Cluster sind spezielle Teilgruppen $C \subset N$
- Cluster können andere Cluster oder Vektoren enthalten: $C = x|x = C_i \vee x \in X$
- leere Cluster gibt es nicht $C \neq \emptyset$
- i.d.R. $n = |N|$, $k = |C|$, $i =$ Attribut-Index, $j =$ zweiter Attribut-Index

Bestimmung der Nähe mittels Abstands- bzw. Distanzmaßen. Distanzmaß wird über Funktion d dargestellt [vgl. [KaRo09]; Kap. 1.2 Types of Data and How to Handle Them]

Tabelle 2.1: Eigenschaften der Abstandfunktion d

1.	$d(x, y) \geq 0$	Distanzen sind steht positiv
2.	$d(x, x) = 0$	zwei gleiche Objekte haben immer einen Abstand von 0
3.	$d(x, y) = d(y, x)$	die Distanzfunktion ist kommutativ bzw. symmetrisch
4.	$d(x, z) \leq d(x, y) + d(y, z)$	Distanzen geben stets den kürzesten Weg an

Statt Distanzmaße auch Verwendung von Ähnlichkeitsmaßen $s(x, y)$ (engl. similarity) möglich. Ähnlichkeit i.d.R. im Interval $[0, 1]$ angegeben, wobei $s(x, x) = 1$. Wenn Distanz z.B. durch Normalisierung ebenfalls im Interval $[0, 1]$ liegt, dann gilt: $d(x, y) = 1 - s(x, y)$ [vgl. [KaRo09]; Kap. 1.2 Types of Data and How to Handle Them]

Distanz und Ähnlichkeit dadurch beliebig austauschbar, deshalb diese Verwendung empfehlenswert.

(Frage: Erwähnung der Proximity Matrix? ist eigentlich nicht wichtig für diese Arbeit)

2.2.2 Numerische Attribute

durch (rationale) Zahlen dargestellt [vgl. [KaRo09]; Kap. 1.2 Types of Data and How to Handle Them], mit stetigen (engl. continuous) Werten [vgl. Huan98]. Umfasst damit sowohl Daten in Intervall- und Verhältnisskalen (engl. interval and ratio data) [vgl. [Bosl12]; Kap. 1 Basic Concepts of Measurement]

Minkowski-Familie

Datenpunkte damit numerischen Vektoren also Punkte. Bestimmung des Abstand der Punkte => verschiedene Maße, gehören alle zur Minkowski-Familie [vgl. [Cha07]; und [King15]; Kap. 1.2 Capturing the Clusters]:

Tabelle 2.2: Übersicht gängiger Maße aus der Minkowski-Familie

Name	p-Norm	Formell
Minkowski	allgemein	$d(x, y) = \sqrt[p]{\sum_{i=1}^{n} \lvert x_i - y_i \rvert^p}$
Manhattan	$p = 1$	$d(x, y) = \sum_{i=1}^{n} \lvert x_i - y_i \rvert$
euklidisch	$p = 2$	$d(x, y) = \sqrt{\sum_{i=1}^{n} \lvert x_i - y_i \rvert^2}$
Chebyshev	$p = \infty$	$d(x, y) = \max \lvert x_i - y_i \rvert$

höhere p-Norm bedeutet i.d.R. robustere Bestimmung des Abstands. Manhattan => Winkel zwischen zwei Punkten, euklidisch => Länger der Geraden durch die Punkte [vgl. [King15]; Kap. 12.3 Which Proximity Measure Should Be Used?]

verschiedenste Versionen und Abwandlungen dieser Maße [siehe Cha07]

- aber die meisten basieren auf Minkowski-Familie
- oder lassen sich auf Minkowski abbilden (z.B. Maße basierend auf der Pearson Korrelation nutzen im Kern euklidischen Abstand zur Berechnung)
- liefern ähnliche Ergebnisse wie klassische Minkowski
- [siehe Cha07] Clustering mit verschiedenen Maßen und dann Ergebnisse geclustert. (eigene Erkenntnis aus dem Paper) im Kern jeder Cluster Gruppe war auch ein Vertreter der Minkowski-Familie

Normalisierung

manche numerischen haben logarithmischen Zusammenhang (10 zu 20 ist gleichbedeutend mit 100 zu 200), dann $x'_i = \log x_i$ für gleichmäßige Abstände

Projektion auf $[0, 1]$ => $x' = \frac{x - \text{avg } X}{\max X - \min X}$; empfehlenswert, wenn Attribute gleichgewichtet sein sollen

manchmal ist Normalisierung nicht sinnvoll, da dadurch verschiedene "Gewichtung" der Attribute bestimmtes Wissen dargestellt wird => erfordert Domänenwissen

Eventuell sogar arbeiten mit Gewichtsvektor für die Attribute => eher spezielle Anwendungsfälle

gesamter Abschnitt [vgl. [KaRo09]; Kap. 1.2 Types of Data and How to Handle Them]

2.2.3 Kategorische Attribute

statt Zahlen bestehen Datenpunkte aus ihnen zugeordneten Kategorien oder Labels [vgl. [Bosl12]; Kap. 5 Categorical Data]

Manchmal gibt es eine sinnvolle Reihenfolge der Labels (z.B. xs, s, m, l, xl), dennoch keine Aussage über Verhältnis oder Intervall zueinander möglich. mit Reihenfolge als ordinal bezeichnet, ohne als nominal [vgl. [Bosl12]; Kap. 1 Basic Concepts of Measurement; Kap. 5 Categorical Data]

Ordinale Attribute

zwei Möglichkeiten:

Reihenfolge ignorieren und als nominal verarbeiten => siehe folgende Abschnitte

aber eher empfohlen: Umwandlung und Verarbeitung als numerische Attribute:

- Nummerierung der Labels nach ihrer Reihenfolge von 1 bis n
- dann: $x' = \frac{x-1}{n-1}$
- dadurch im Interval $[0, 1]$ in $n - 1$ gleichmäßige Abschnitte eingeteilt

ganzer Abschnitt [vgl. [KaRo09]; Kap. 1.2 Types of Data and How to Handle Them]

Nominale Attribute

dieser Teil aus [vgl. [KaRo09]; Kap. 1.2.5 Nominal, Ordinal, and Ratio Variables]

Umwandlung in sog. binäre Attribute:

1. bei 2 Ausprägungen => 0 = erste Ausprägung, 1 = zweite Ausprägung z.B. yes/no
2. ab 3 =>

- entweder in 2 Ausprägungen komprimieren

- oder ein neues Attribut pro Kategorie definieren: 0 = gehört nicht zur Kategorie, 1 = gehört zur Kategorie

folgende aus [[KaRo09]; Kap. 1.2.5 Binary Variables]

für 1. => sog. "symmetrische" binäre Attribute, 2. => "asymmetrische"; Verarbeitung mit speziellen Ähnlichkeitsmaßen:

Simple matching $s(x,y) = \frac{|x \cap y| + |\bar{x} \cap \bar{y}|}{|x \cup \bar{x} \cup y \cup \bar{y}|}$

- jeder Match (beide Datenpunkte haben Label und beide Punkte haben ein Label **nicht**) wird gezählt
- nur für symmetrische geeignet
- gibt Varianten davon mit unterschiedlichen Gewichten für Matches usw. => lassen sich alle auf Simple matching abbilden

Jaccard-Koeffizent $s(x,y) = \frac{|x \cap y|}{|x \cup y|}$

- nur die tatsächlich vorhandenen Attribute (mit 1 codiert) werden miteinander verglichen
- nicht-vorhandene Attribute bei beiden Datenpunkten werden ignoriert
- gleiches Bsp:
  - 3 Farben (rot, grün, blau)
  - x ist rot; y ist blau
  - Simple matching würde $\frac{1}{3}$ ergeben, weil ja beide gemeinsam haben **nicht** grün zu sein
  - Jaccard hingegen sagt Ähnlichkeit $\frac{0}{2}$, weil keines von zwei vorhandenen Attributen matched

2.2.4 String-Attribute

freie Textfelder

Tokenization

!!!QUELLEN!!! [CRF03]

Bei nominalen Attributen mit vielen Ausprägungen (bsp Produkttitel => jeder ist anders)

- Zerlegen in einzelne Wörter => Tokenization
- Rückführung der Wörter auf ihren Stamm (z.B. Porter-Stemming)
- Entfernen von Füllwörtern => Stop-Word Removal

Am Ende erhalten wir eine feste Menge an Tokens oder Keywords => Menge kategorischen Daten => asymmetrisch binär verarbeiten

oder auch Überführung in Vektor-Space ...

String-Metrics (eventuell weglassen??)

!!!QUELLEN!!! [CRF03] [RaRa11]

Alternativ kann Abstand auch mittels String-Metrics ermittelt werden.

String-Metrics messen die Ähnlichkeit zweier Strings

Bsp:

- Levenshtein: Anzahl an Buchstaben eingefügt/geändert/gelöscht um x auf y umzuwandeln
- Hamming: Anzahl an **ungleichen** Buchstaben => beide Strings müssen gleich lang sein
- Jaro: Verhältnis gleicher zu verschiedenen Buchstaben, performant mit guten Ergebnissen
- Jaro-Winkler: Erweiterung von Jaro mit höherem Gewicht auf den Anfang (Prefix) der Wörter

Verwendung:

- entweder String-Metrics selbst als Distanzmaß nutzen
- oder "ordinale" Reihenfolge mittels String-Metrics bestimmen und dann als ordinal betrachten

2.2.5 Gemischte Attribute

[vgl. [KaRo09]; Kap. 1.2.6 Mixed Variables]

verschiedene Ansätze:

Umwandlung in numerische Werte

- siehe ordinal
- siehe nominal => String-Metrics

Umwandlung in kategorische Werte

- ordinal als nominal betrachten (siehe Ordinale Attribute)
- numerisch Umwandeln durch Diskretisierung
  - Bildung von festen Bändern/Gruppen z.B. Alter => $10-19$, $20-29$ etc.

Separates Clustern je Attribut-Klasse und Subspace-Clustering

mehrmaliges Clustern mit jeweils nur Attributen eines Skalenniveaus

anschließend Vergleich der verschiedenen gebildeten Cluster

noch weitere komplexere Ansätze, wo verschiedenste Kombinationen an Attributen für das Clustern ausgewählt werden und verglichen werden => "Subspace-Clustering" [siehe JiCh17]

Kombinierte Distanzfunktion

Bewertung jedes Attributs mit geeignetem Distanzmaß. Anschließend zusammenrechnen mit Gleichgewichtung

ab hier aus [vgl. Huan98]

z.B. bei k-Prototype werden numerische und kategorische mit jeweils geeigneten Distanzmaßen verrechnet:

$d(x,y) = wd_numerical(x,y) + (1-w)d_categorical(x,y)$, also z.B. euklidisch und simple matching oder Manhattan und Jaccard etc.

w soll eine Gleichgewichtung erzeugen also z.B. $\frac{Anzahl numerischer Attribute}{Anzahl aller Attribute}$

2.3 Clustering-Verfahren

2.3.1 Partitionierendes Clustering

Überblick

Minimierungsproblem: initiale Cluster-Zuordnungen (Partitionen), dann Veränderungen der Cluster-Zuordnungen bis ein lokales Minimum gefunden worden ist. [vgl. [King15]; Kap. 4.1 Introduction]

Verschiedene Algorithmen und Varianten: initiale Selection der Cluster, Wahl der Distanzmaße, Vorgehen während der Minimierung [vgl. [King15]; Kap. 4.1 Introduction] und für welche Arten von Attribute geeignet, dazu später mehr [siehe Huan98]

Effizienz:

- neigt zu lokalen Minima => mehrmaliges Wiederholen des Clustering mit verschiedenen zufälligen Startpunkten
- dennoch sehr effizient lösbar: $\mathcal{O}(n \cdot k \cdot l)$
- n Datenpunkte, k Cluster, l Wiederholungen des Clusterings
- k und l kleine Werte, viel kleiner als n
- also: $\mathcal{O}(n)$
- [vgl. Huan98]

Aber: k muss vorher bekannt sein => verschiedene k probieren oder z.B. mit hierarchischem Verfahren k abschätzen [vgl. [King15]; Kap. 4.1 Introduction]

k-Means

klassischster Vertreter des Clusterings und weit verbreitet und genutzt [vgl. Huan98]

Jedes Cluster wird über einen Schwerpunkt (Mittelpunkt) repräsentiert [vgl. StKaKu00]. Es gilt die Datenpunkte so den Clustern zuzuordnen, dass die Summe der Abstände der Datenpunkte zum Mittelpunkt so klein wie möglich sind. Schwerpunkt wird über Mittelwert (engl. mean) der Clustermitglieder bestimmt. [vgl. [King15]; Kap. 4.5 K-Means Algorithm]

Funktioniert daher nur mit numerischen Werten [vgl. Huan98]

Ablauf [vgl. [King15]; Kap. 4.5 K-Means Algorithm]:

1. Wahl der initialen k Startpunkte
2. Zuordnen aller Datenpunkte zum nächstliegenden Schwerpunkt
3. Neuberechnung der Schwerpunkte mittels Durchschnittswert (engl. mean) der Datenpunkte
4. ab 2. wiederholen, solange bis keine/kaum noch Änderungen der Schwerpunkte

Für Wahl der initialen Schwerpunkte [vgl. [King15]; Kap. 4.3 The Initial Partition]:

- ersten k Datenpunkte
- oder gleichmäßig aus der gesamten Liste
- oder zufällig über gesamte Liste
- und weitere

z.B. mehrmals mit verschiedenen random Startpunkten durchführen und bestes Ergebnis nehmen => verteilte Berechnung möglich

Statt die Schwerpunkte nur einmal am Ende eines Durchlaufes neuberechnen (Forgy's Method) auch permanente Neuberechnung (mit jedem neu zugeordnetem Datenpunkt) möglich (MacQueen's Method) und weitere Abwandlungen [vgl. [King15]; Kap. 4.5 K-Means Algorithm]

k-Medoids

[ArVa16]

- Variante des k-Means
- median statt mean als Clusterschwerpunkte
- median Berechnung komplexer, da Abstand aller Punkte eines Clusters zueinander berechnet werden muss
- aber laut Autor bessere Ergebnisse, schnellere Konvergenz und dadurch trotzdem kürzere Rechenzeit

k-Modes

[Huan98]

Variante des k-Means für kategorische (Attribute)

Änderungen zum k-Means:

- Simple matching als Distanzmaß (allerdings auch andere z.B. Jaccard möglich)
- statt Durchschnittswertes (engl. mean) wird Clusterschwerpunkt aus dem Modus (engl. mode) bestimmt
- Update der modes eines Clusters während des Clustering anhand ihrer Häufigkeit (Frequenz)

nach einem Durchlauf, Set erneut durchgehen und checken, ob einige Punkte nicht einem anderen Cluster zugeordnet werden müssen, da sich Modus des Datensets während des Clusterns ja häufig ändert. nach Reassign => Modus Frequenz in beiden Clustern anpassen

k-Prototype

[Huan98]

Kombi aus k-means und k-modes für gemischte Datensets

nutzt ein Distanzmaß für numerische und eins für kategorische Werte + Gewicht zur gleichgewichtung der Attribute siehe gemischte Attribute

Schwerpunkt der numerische Attribute ist der Durchschnitt (mean) und der nominalen ist der Modus (mode) => also Kombi aus beiden

2.3.2 Hierarchisches Clustering

Überblick

!!!QUELLEN!!!

schrittweise Zuordnung zu Clustern => top-down (divisive) oder bottom-up (agglomerative)

Visualisierung: Dendogramm

Vorteil: k muss vorher nicht bekannt sein, sondern Hierarchie mit beliebiger Feinheit, Erkenntnisse aus Kind- und Vater-Clustern

Nachteile: einmal kombiniert/geteilt kein Reassignment der Cluster mehr, teilweise seltsame Ergebnisse, Fehler ziehen sich immer weiter durch

rechenintensiv: Vergleich von jedem Datenpunkt mit jedem anderen Datenpunkt, also mindestens $\mathcal{O}(n^2)$ meistens noch aufwendiger, da Datenset k mal durchsucht werden muss. Da hierarchisch ist $k \approx n$, also meistens $\mathcal{O}(n^2 \log n)$ (mit Optimierungen) bzw. $\mathcal{O}(n^3)$

Agglomeratives Clustering (Bottom-Up)

!!!QUELLEN!!!

Ablauf:

1. alle Datenpunkte in einem eigenen Cluster
2. Ermittlung der beiden am nächsten gelegenen Cluster
3. Fusion dieser beiden Cluster zu einem größeren
4. ab 2. wiederholen bis alle Punkte in einem Cluster sind

Theoretisch könnte auch vorher abgebrochen werden, aber für den Anwender sind i.d.R. die obersten "größten" Cluster-Hierarchien am interessantesten

zentrale Frage: wie werden zwei Cluster mit mehreren Datenpunkten miteinander verglichen => Linkage

Bsp immer die beiden dichtesten Datenpunkt beider Cluster => Single-Link

Name	Formell	Beschreibung	Eigenschaften
Single-Link	$d(X,Y) = \min d(x_i, y_j)$	Abstand durch die beiden dichtesten Punkte definiert	neigt zur Bildung von langen Ketten (chaining effect), anfällig gegen Outlier
Complete-Link	$d(X,Y) = \max d(x_i, y_j)$	Abstand durch die beiden am entferntesten liegenden Punkte definiert	neigt zur Bildung von vielen sehr kleinen Clustern, anfällig gegen Outlier
Average-Link	$d(X,Y) = \text{avg } d(x_i, y_j)$	durchschnittlicher Abstand aller Punkte der beiden Cluster zueinander	ähnlich dem Mittelpunkt von k-Means, aber viel aufwendiger, da immerzu jeder Punkt eines Clusters mit jedem anderen Punkt der anderen Clustern verglichen werden muss
Mediod-Link	$d(X,Y) =$ ...	Median Punkt eines Cluster repräsentiert das Cluster	ähnliches Verhalten wie Average-Link, aber effizienter zu berechnen, da der Median eines neu gebildeten Clusters nur einmal bestimmt werden muss
k-Centroid-Link ???	$d(X,Y) =$ ...	vergleicht den Durchschnitt der k zentralsten Punkte der Cluster miteinander; allgemeine Form von Mediod-Link ($k = 1$) und Average-Link ($k = \|C\|$)	liegt je nach k irgendwo zwischen Average- und Mediod-Link; k ist ein weiterer Parameter, der festgelegt werden muss

Name	Formell	Beschreibung	Eigenschaften
Ward-Method	$d(X,Y) = \frac{d(\bar{x},\bar{y})^2}{\frac{1}{\lvert X\rvert}+\frac{1}{\lvert Y\rvert}}$	berechnet die Varianz für jedem potenziellen Merge, Merge mit höchster Reduktion der Varianz wird ausgewählt	sehr aufwendig zu berechnen, da immer wieder jeder Punkt mit jedem anderen verrechnet werden muss

Insgesamt alle sehr rechenintensiv. single- und complete-link bei guter Implementierung in $\mathcal{O}(n^2)$, alle anderen mindestens in $\mathcal{O}(n^2 \log n)$ oder $\mathcal{O}(n^3)$ (vor allem bei Average-Link und Ward-Method)

Diversives Clustering (Top-Down)

Überblick ergibt für Menschen intuitivere Cluster, da wir ebenfalls mental so vorgehen [vgl. [King15]; Kap. 3.3 Agglomerative versus Divisive Clustering]

theoretisch weniger aufwendig als agglomerativ, wenn nicht komplette Hierarchie generiert wird => weniger Durchläufe da i.d.R. $k < n - k$ (von 1 Cluster zu k weniger Schritte als von n zu k Clustern)

Problem: erstes Cluster riesig groß mit theoretisch $2^{n-1} - 1$ möglichen Arten des Splitterns (also $\mathcal{O}(2^n)$)

in der Praxis wird dieses Problem durch geschicktes Vorgehen umgangen:

DIANA [vgl. [KaRo09]; Kap. Divisive Analysis (Program DIANA)]

klassisches, erstes Verfahren, wie Abspaltung bei einer Partei

Ablauf:

1. Start mit allen Punkten in einem großen Cluster
2. Berechnung der durchschnittlichen Abweichung aller Punkte voneinander
3. Punkt mit größter Abweichung => Startpunkt einer “Splittergruppe”
4. Berechnung der durchschnittlichen Abweichung ohne Punkt mit größter Abweichung
5. Differenz der beiden Abweichungen größer geworden und am größten => Punkt der “Splittergruppe” zuordnen
6. wiederholen 4. bis Differenzen nur noch negativ
7. Durchmesser der Cluster berechnen $\emptyset = \max d(x, y)$
8. Cluster mit größtem Durchmesser verwenden und ab 2. wiederholen, bis gewünschte Abbruchbedingung erreicht ist

Abbruchbedingung kann bestimmter Durchmesser sein oder ein festes k, spätestens Schluss, wenn $k = n$

Laufzeit (eigene Überlegung; kleiner Kommentar dazu in [StKaKu00])

- Berechnung der durchschnittlichen Abweichung: jeder Punkt mit jedem anderen vergleichen => $\mathcal{O}(n^2)$
- da Cluster immer kleiner werden, wird Berechnung mit jedem Schritt einfacher

[RaRa11] Versuche mit agglomerativ und divisive nach DIANA => DIANA immer doppelt so schnell fertig für das gesamte Datenset

Bisecting k-Means [vgl. StKaKu00]

zur Teilung des größten Clusters wird k-Means mit $k = 2$ genutzt

- dadurch viel effizienter da k-Means in $\mathcal{O}(n)$
- k-Means wird zwar k mal aufgerufen, aber gleichzeitig wird die Anzahl der Datenpunkte je Cluster mit jeder Teilung kleiner

2.4 Cluster Validität

[[King15]; Kap. 8.1 Introduction] => fehlerbehaftet ... ???

unsupervised learning, dennoch Analyse, ob gefundene Cluster gut sind

[RAAQ11]

Clustering validation prüfen, ob Cluster eine gute/natürliche Partition sind, Prüfung durch indices

3 Ansätze:

1. Internal Indices: nur Verwendung der Daten des Datensets selbst
2. External Indices: Vergleich mit externen Daten z.B. extern festgelegten Clustern
3. Relative Indices: Vergleich der Ergebnisse verschiedener Clustering Durchläufe miteinander

[[King15]; Kap. 8.1 Introduction]

Wenn keine externen Daten verfügbar => z.B. Monte-Carlo-Simulationen zur Evaluation durchführen => nicht weiter betrachtet, da externe Daten vorhanden sind

2.4.1 Internal Indices

[StKaKu00]

- Overall Similarity ...
- $\sum d(C_x i, C_x j)/|C|$
- oder cosine() ?
- ...

[RAAQ11]

- Dunn index

- Silhouette index
- Bayesian Information Criterion
- ...

2.4.2 External Indices

[[King15]; Kap. 8.4 Indices of Cluster Validity]

- Rand Index => ähnlich simple matching nur aus Sicht der Cluster
- Jaccard => nur aus Sicht der Cluster
- Abwandlungen davon ...

[RAAQ11]

- Purity => Gegenstück zur Entropy
- Normalized Mutual Information (NMI) Measure

[RAAQ11] & [StKaKu00]

- Entropy => "Varianz der extern zugeordneten Klassen je Cluster"
- F-measure => Betrachtung der Cluster als "Such-Queries"

3 Konzeption

3.1 Überblick

- möglichst realistische Datenquellen => Akeneo-PIM + Icecat Datenkatalog
- Clustering mittels möglichst effizienten Verfahren
  - dabei verschiedene Attribute, Datentypen, Gewichtungen probieren
  - möglichst ohne "externe" Informationen clustern können
- Evaluation => ist clustering sinnvoll => Was ist sinnvoll?
- Überblick über das Vorgehen

3.2 Datenquellen und -sets

3.2.1 Akeneo-PIM

Überblick

- open source PIM-System
- weite Verbreitung
- typischer Funktionsumfang
- große Menge an Addons/Plugins (u.a. Icecat Importer)

Systemübersicht

- Microservice Cluster
- Frontend UI
- Rest-Api

Datenstrukturen

- viele Verschiedene: AUFZÄHLUNG...
- nur einige relevant => genauerer Blick

Product & Product Values

- Meta-Values
- "values" Eintrag ist Mapping: attribute.code => product_value

Category

- Baumstruktur für Kategorisierung

Family

- definiert Attribute pro Family und required/optional

Attribute & Attribute Group

- verschiedene Typen + Constraints

Channel, Currency, Locale

- alles lokalisierbar (Sprache und Währung)
- zusätzlich noch scopes => verschiedene Werte für gleiche Locale z.B. Web und Mobile

Measurement Family

- Maßeinheiten mit Umrechnung

3.2.2 Icecat Produkt-Katalog

- offener Katalog mit XXXXXXXX Produktdaten
- einigermaßen standardisiert nach UNSPEC-Taxonomy

3.2.3 Datenset

Anforderungen

- Produkte mit wenigen Attributen
- Produkte mit vielen Attributen
- unterschiedliche aber naheliegende Kategorien
- Produkte mit Variationen und verschiedenen Generationen
- Duplikate

Umsetzung

- Smartphone-Hüllen speziell für Samsung Galaxy Reihe
- Smartphones aus Samsung Galaxy Reihe S20 und S21 + Versionen und Farben
- evtl. noch weitere Smartphones
- evtl. ein paar Tablets
- evtl. noch was ganz anderes wie Ladegeräte o.ä.

3.3 Clustering

3.3.1 Clustering-Verfahren

Anforderungen:

- hierarchisch wünschenswert
  - mehr Informationen ableitbar
  - Anzahl an Clustern dynamisch ableitbar
- gute Laufzeit => potenziell hohe Menge an Produkten
- Verarbeitung verschiedenster Datentypen in den Attributen

Ansatz: Bisecting K-Prototypes mit Custom-Varianten von Distanzfunktionen

3.3.2 Distanzfunktion

Anforderungen

- Berechnung der Distanz + Berechnung des Mittelpunkts
- Kombi von verschiedenen Datentypen
- viele Null-Values

Tabelle 3.1: Verarbeitung der Akeneo-Attribut-Typen

Datentyp	Akeneo-Typ	Distanzmaß	Centroid
numerisch	Number, Metric, Price, Date	Minkowski	mean
kategorisch	Bool, SelectSingle, ReferenceDataSingle	Jaccard	mode
multi-kategorisch	SelectMulti, ReferenceDataMulti	Jaccard+	mode
strings	Text, Textarea	?	?
Datei	File, Image	-	-
Id	Identifier	-	-

Herleitung der Funktion

- $d(p_1, p_2) = \frac{v+n+c+m+?}{|p_1^{attr} \cup p_2^{attr}|}$
- $v = |p_1^{attr} \setminus p_2^{attr} \cup p_2^{attr} \setminus p_1^{attr}|$
- $n = \sum_{x \in p_1^{num_i},\ y \in p_2^{num_i}} |x - y|$
- $c = \sum_{x \in p_1^{cat_i},\ y \in p_2^{cat_i}} q(x, y)$
- $q(x, y) = \begin{cases} 0, & x = y \\ 1, & x \neq y \end{cases}$
- $m = \sum_{x \in p_1^{mul_i},\ y \in p_2^{mul_i}} 1 - \frac{|x \cap y|}{|x \cup y|}$

... weitere Erklärungen etc...

Umgang mit verschiedenen Versionen der Funktion?

3.4 Evaluation

- Ziel: sinnvolle Cluster finden => Was heißt das?

3.4.1 Stabilität der Cluster

- Hintergrund:
  - Clustering ist wertlos, wenn Zuordnung zufällig erfolgt
  - Clustering sollte stabil, reproduzierbar & weitestgehend deterministisch sein
- mehrmaliges Clustern (gleiche Settings):
  - K-Means mit random init => evtl. unterschiedliche Ergebnisse bei verschiedenen Durchläufen
  - gute Distanzfunktion => klare Separierung der Produkte
  - je größer die Ähnlichkeit zwischen zwei Clusterings, desto stabiler
  - Bewertung der Ähnlichkeit mittels Jaccard-Koeffizient
- Analyse der Auswirkung Settings
  - Clustern mit verschiedenen Attributen/Gewichtungen
  - Bewertung der Ähnlichkeit der Clusterings
  - ebenfalls Jaccard-Koeffizient

3.4.2 Internal Indices

- allgemeine Stats zum Clustering:
  - durchschnittlicher Error
  - Entwicklung über Hierarchien
  - Anzahl an Produkten

=> eher untergeordnete Rolle, eventuell weglassen ???

3.4.3 External Indices

- Hintergrund:
  - bisher nur Aussage wie ähnliche/verschieden die Clusterings sind
  - Frage ist noch: Wenn Clusterings unähnlich,welches ist “besser”
  - Ansatz => Inspiration aus möglichen Anwendungen:
    - Auto-Kategorisierung
    - Einordnung neuer Produkte zu bestehenden (transitive Erkenntnisse)
    - Finden von Alternativ-Produkten mit gleichen Eigenschaften
    - Finden von Duplikaten
- Vergleich mit Akeneo Families und Akeneo Categories
  - Families: ähnliche Attribute, sollten sich ähnlich sein, eher technischer Check
  - Categories: menschengemachte Labels, spiegeln die intuitive Zuordnung wieder

  - Analyse der Ähnlichkeit zwischen Clustering und Families/Categories

- speziellere Checks:
  - bewusst eingefügte Duplikate => solange wie möglich im gleichen Cluster (top-down clustering)
  - Produkte unterschiedlicher Kategorien, aber mit Bezug (z.B. Smartphone plus Hülle) => bleiben sie länger im gleichen Cluster als andere Produkte dieser Kategorien?

3.5 Vorgehen / Versuche ?

3.5.1 Set "Samsung Galaxy Cases"

- Smartphone Hüllen für Samsung Galaxy Reihe sowohl
- überschaubar wenig Attribute

Versuch: Basis-Prozess, Fehler finden

- clustern nach einander mit numerisch + kategorisch (+ multi-kategorisch)
- Prozess und "Pipeline" aufsetzen

Versuch: gleiche Modelle

- Clustering soll Hüllen für gleiches Smartphone-Modell finden
- werden in Akeneo Categories vorher hinterlegt, Vergleich damit
- numerisch & kategorisch => einzeln und zusammen
- (multi-kategorisch => hier noch nicht möglich, da solche keine Attribute)
- gleich-gewichtet vs menschen-gewichtet

3.5.2 Set "Samsung Galaxy S 20er-Reihen"

- Smartphones Samsung Galaxy S 20 und 21 Modelle
- verschiedenen Versionen (Größen) und Farben
- sehr viel Attribute
- enthält Duplikate

Versuch: gleiche Modelle finden

- Clustering soll gleiche Modelle/Baureihen finden
- werden in Akeneo Categories vorher hinterlegt, Vergleich damit
- vor allem interessant: nur required Attribute vs optionale hinzunehmen => wird es dadurch besser oder schlechter?
- multi-kategorisch ebenfalls auswerten => macht es einen Unterschied?
- gleich-gewichtet vs menschen-gewichtet

Versuch: Duplikate finden

- Duplikate sollten sehr lange im gleichen Cluster sein

- eventuell zusammen mit dem vorherigen Versuch evaluieren ???
- Hauptfrage: gleiche Settings für beide Aufgaben nutzbar?

3.5.3 Set “Samsung Galaxy Smartphones und Hüllen”

- Kombi beider Sets
- Clustering mit gefundenen Konfig(s)
- Frage: Assoziation zwischen Modellen sichtbar?

...

3.5.4 weiteres ?

- weitere Versuche?
- eventuell durch Hinzunehmen von Smartphones und Hüllen anderer Hersteller ?
- weitere Produktgruppe – ähnlich aber schon anders z.B. Tablets?
- weitere Produktgruppe – ganz anders z.B. andere Elektroartikel?

4 Implementierung

- Akeneo-PIM:
  - auf Server von adesso
  - Akeneo Icecat Connector
  - Import der Daten
- Python Modules:
  - Akeneo Client => holt Daten aus Akeneo-PIM und dumpt in json Dateien => “raw data”
  - Akeneo Parser => parsing von json in Python-Strukturen (dataclasses) => “cleaned data”
  - Akeneo Clustering => Filtern der Attribute, Vorverarbeitung fürs Clustering
  - Clustering => eigene Implementierung des Bisecting K-Means mit generischen Datenstrukturen
  - Evaluation => Hilfsmethoden zur Auswertung der Ergebnisse
- Jupyter Notebooks:
  - Akeneo Rest Api Exploration
  - Clustering Basis Example (2d Vektoren)
  - verschiedene Iterationen des Clusterings und Distanzfunktionen

...

5 Auswertung

...

6 Fazit und Ausblick

Quellenverzeichnis

[ArVa16] Arora, Preeti ; Varshney, Shipra ; u. a.: Analysis of k-means and k-medoids algorithm for big data. In: *Procedia Computer Science* Bd. 78, Elsevier (2016), S. 507–512

[Bosl12] Boslaugh, Sarah: *Statistics in a nutshell: A desktop quick reference* : O'Reilly Media, 2012

[Cha07] Cha, Sung-Hyuk: Comprehensive survey on distance/similarity measures between probability density functions. In: *City* Bd. 1 (2007), Nr. 2, S. 1

[CRF03] Cohen, William W ; Ravikumar, Pradeep ; Fienberg, Stephen E ; u. a.: A Comparison of String Distance Metrics for Name-Matching Tasks. In: *IIWeb*. Bd. 3 : Citeseer, 2003, S. 73–78

[Huan98] Huang, Zhexue: Extensions to the k-means algorithm for clustering large data sets with categorical values. In: *Data mining and knowledge discovery* Bd. 2, Springer (1998), Nr. 3, S. 283–304

[JiCh17] Jia, Hong ; Cheung, Yiu-Ming: Subspace clustering of categorical and numerical data with an unknown number of clusters. In: *IEEE transactions on neural networks and learning systems* Bd. 29, IEEE (2017), Nr. 8, S. 3308–3325

[KaRo09] Kaufman, Leonard ; Rousseeuw, Peter J: *Finding groups in data: an introduction to cluster analysis*. Bd. 344 : John Wiley & Sons, 2009

[King15] King, Ronald S: *Cluster analysis and data mining: An introduction* : Stylus Publishing, LLC, 2015

[KoLo12] Kou, Gang ; Lou, Chunwei: Multiple factor hierarchical clustering algorithm for large scale web page and search engine clickstream data. In: *Annals of Operations Research* Bd. 197, Springer (2012), Nr. 1, S. 123–134

[RAAQ11] Rendón, Eréndira ; Abundez, Itzel ; Arizmendi, Alejandra ; Quiroz, Elvia M: Internal versus External cluster validation indexes. In: *International Journal of computers and communications* Bd. 5 (2011), Nr. 1, S. 27–34

[RaRa11] Rajalingam, N ; Ranjini, K: Hierarchical Clustering Algorithm - A Comparative Study. In: *International Journal of Computer Applications* Bd. 19, Citeseer (2011), Nr. 3, S. 42–46

[StKaKu00] Steinbach, Michael ; Karypis, George ; Kumar, Vipin: A comparison of document clustering techniques (2000)

Anhang

- Link zum Repo
- Überblick zum Repo

Selbstständigkeitserklärung

Die Autoren versichern hiermit, dass die vorliegende Arbeit mit dem Thema

Generisches Clustern hoch-komplexer Produktdaten

selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmitteln angefertigt worden ist.

Erfurt, den 29.04.2022

.....
Hannes Dröse