HAW HAMBURG

BACHELOR THESIS
Theophile Teyou Soh

Analyse und Vergleich von verschiedenen Clustering - Verfahren auf Datensätze unterschiedlicher Dimensionalitäten

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Engineering and Computer Science
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

HAW HAMBURG

Bachelorarbeit

Theophile Teyou Soh

Analyse und Vergleich von verschiedenen Clustering - Verfahren auf Datensätze unterschiedlicher Dimensionalitäten

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department Computer Science

Theophile Teyou Soh

Analyse und Vergleich von verschiedenen Clustering - Verfahren auf Datensätze unterschiedlicher Dimensionalitäten

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Technische Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Michael Neitzke
Zweitgutachter: Prof. Dr. Marina Tropmann-Frick

Eingereicht am: 11.07.2022

Theophile Teyou Soh

Thema der Arbeit

Analyse und Vergleich von verschiedenen Clustering - Verfahren auf Datensätze unterschiedlicher Dimensionalitäten

Stichworte

Clustering, Dimensionsreduktion, CURE Clustering, spektrales Clustering, PCA, UMAP, T-SNE, maschinelles Lernen, Hochdimensionaler Datensatz

Kurzzusammenfassung

Clustering stellt einen wichtigen Teil des maschinellen Lernens dar, der nicht vernachlässigt werden kann und die Klassifizierung von Daten ermöglicht. Im Rahmen der vorliegenden Studie wird ein Gesamtüberblick über den Vergleich verschiedener Clustering-Methoden gegeben. Ebenso werden ihre Leistung auf mehrdimensionalen Datensätze analysiert. In dieser Arbeit wird ein System entworfen, welches aus Clustering-Algorithmen besteht. Nachdem der Datensatz ausgewählt und die Merkmale der Cluster identifiziert wurden, wird die Leistung der Methode analysiert. Da die Daten hochdimensional sind, umfasst diese Arbeit auch eine Faktorenanalyse der Daten sowie eine anschließende Reduktion der Daten auf niedrigere Dimensionen zum Zweck der Visualisierung.

. . .

Theophile Teyou Soh

Title of Thesis

Analysis and comparison of different clustering methods on data sets of different dimensionalities

Keywords

Clustering, dimensionality reduction, CURE clustring, spectral clustering, PCA, UMAP, T-SNE, machine learning, high dimensional dataset

Abstract

Clustering is an important part of machine learning that cannot be neglected and enables classification of data. This study provides an overall view of the comparison of different clustering methods and analyzes their performance on multidimensional datasets. A system is designed which consists of the composite of clustering algorithms. After selecting the dataset and identifying the characteristics of the clusters, the performance of the method is analyzed. Since the data is high dimensional, this work also includes factor analysis of the data and subsequent reduction of the data to lower dimensions for visualization.

. . .

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung **1**
- 1.1 Motivation . . . 1
- 1.2 Zielsetzung und Methodik . . . 2
- 1.3 Aufbau der Arbeit . . . 3
- 1.4 Zusammenfassung . . . 5

2 Grundlagen / Theoretisches Wissen **6**
- 2.1 Maschinelles Lernen . . . 6
  - 2.1.1 Lernstile . . . 7
- 2.2 Kategorien von Clustering-Verfahren . . . 10
- 2.3 Herausforderung beim hochdimensionalen Clustering . . . 15
  - 2.3.1 Distanzmaße und Ähnlichkeitsmaße . . . 15
  - 2.3.2 Ergebnisvalidierung . . . 18
- 2.4 Dimensionsreduktion . . . 22
  - 2.4.1 T-Distributed Stochastic Neighbor Embedding (T-SNE) . . . 22
  - 2.4.2 Principal Component Analysis (PCA) . . . 23
  - 2.4.3 Uniform Manifold Approximation and Projection (UMAP) . . . 24
- 2.5 Zusammenfassung . . . 24

3 Prozessablauf / Realisierung **25**
- 3.1 Ablauf . . . 25
- 3.2 Datensätze . . . 27
- 3.3 Beschreibung der Verfahren . . . 29
- 3.4 Zusammenfassung . . . 36

4 Analyse / Auswertung der Ergebnissen **37**
4.1 Bewertung und Visualisierung 37
4.1.1 Agglomerative Clustering 37
4.1.2 Expectation Maximization Clustering 41
4.1.3 Spectral Clustering 43
4.1.4 CURE Clustering 46
4.2 Zusammenfassung 50

5 Zusammenfassung und Ausblick **51**
5.1 Zusammenfassung 51
5.2 Ausblick 52

A Appendices **56**
Selbstständigkeitserklärung 57

Abbildungsverzeichnis

1.1 Klassifikationsmodell 2
1.2 Aufbau der Arbeit 3

2.1 Methoden des maschinellen Lernens 7
2.2 Ablauf des überwachten Lernens 8
2.3 Ablauf des unüberwachten Lernens 9
2.4 Beispielhafte Darstellung eines dichtebasierten Clusterings 11
2.5 Beispielhafte Darstellung eines partitionierenden Clusterings 12
2.6 Beispielhaftes Dendrogramm zur Darstellung eines hierarchischen Clusterings 14
2.7 UMAP-Schritte und -Komponenten 24

3.1 Ablauf der Realisierung der Analyse 26

4.1 Darstellung verschiedener Indexe auf die drei Datensätze für jedes agglomerative Verfahren 38
4.2 Visualisierung von agglomerativen Clustering mit „ward Linkage“: Darstellung mit den Algorithmen der Dimensionsreduktion (PCA,TSNE,UMAP) 40
4.3 Visualisierung von Expectation Maximization Clustering: Darstellung mit den Algorithmen von Dimensionsreduktion (PCA,TSNE,UMAP) 43
4.4 Darstellung verschiedener Indexe auf die drei Datensätze für jedes spektrale Verfahren 44
4.5 Spectral-Clustering mit „affinity nearest-neighbors“: Darstellung mit den Algorithmen zur Dimensionsreduktion (PCA,TSNE,UMAP) 46
4.6 Dendrogramme von den drei Datensätzen mit *„ward“* Methode 47
4.7 Visualisierung von CURE-Clustering: Darstellung mit den Algorithmen von Dimensionsreduktion (PCA,TSNE,UMAP) 49

Tabellenverzeichnis

3.1 Überblick über Beispieldatensatz von BMI 28
3.2 Überblick über Beispieldatensatz von BlogFeedback Data Set 29

4.1 Metriks Resultat: Agglomevatives Clustering 38
4.2 Metriks Resultat: Expectation Maximization Clustering 41
4.3 Metriks Resultat: Spectral Clustering 44
4.4 Metriks Resultat: CURE Clustering 46

1 Einleitung

Die vorliegende Arbeit wurde sich mit der Clusteranalyse von hochdimensionalen Datensätzen gesetzt, die im Einzelfall verhältnismäßig andersartig bezeichnet werden (*Cluster-Analysis*[1], *Automatic Classification*[2], *Dimension reduction*[3], usw.).
Das folgende Kapitel beinhaltet die Motivation, Zielsetzung, Methodik sowie den Aufbau der Arbeit. Dabei besteht das primäre Ziel darin, den Einstieg in das bestimmte Thema zu geben.

1.1 Motivation

Die kontinuierliche Entwicklung der Welt im Allgemeinen in der Wirtschaft und im Besonderen in der Industrie sowie im Handel führt zu verschiedenen Krisen, die nicht nur Online-Shops, sondern auch andere Industriezweige dazu zwingen, ihren Online-Handel zu optimieren.
Wie Online-Shops bieten auch viele andere Unternehmen anderen Händlern sowie privaten Kunden den Verkauf und Versand ihrer Produkte feil. Bei der Bearbeitung von Bestellungen und der Verwaltung von Lieferungen sowie aufgrund der Tatsache, dass sich Produktdaten ständig ändern, ist es nicht immer einfach, Produktinformationen in Form von Befehlen zu erhalten. Auch als Geschäftsstrategie sollten Unternehmen ihre Kundenlisten und die Artikel, an denen sie interessiert sind, kategorisieren, um ihnen in Zukunft ähnliche Produkte anbieten zu können.
In den letzten Jahren hat maschinelles Lernen zunehmende Aufmerksamkeit von Markt erhalten. Dies lässt sich durch die Entwicklung im Bereich der Künstlichen Intelligenz erklären. [25].S.10.

[1] Clusteranalyse
[2] Automatische Methode von Klassifikation
[3] Dimensionreduktion

1.2 Zielsetzung und Methodik

Die Klassifikation von Datensätzen wurde bereits in einer Vielzahl von Arbeiten behandelt. Die meisten Lösungen beschränken sich auf Nachrichten, Blogs und Online-Rezensionen. Die Datenklassifizierung stellt ebenfalls einen Bereich von zunehmenden Interesse dar. Darüber hinaus fördert sie die Forschung auf diesem Gebiet. Die meisten dieser Werke klassifizieren Methoden des maschinellen Lernens im kleinen Maßstab. Mit mehreren Klassen und Algorithmen haben sie sich als effektiv erwiesen und zeigen eine hohe Genauigkeit. Die Abbildung 1.1 zeigt die primäre Funktion von einem Klassifikationssystem bzw. Clustersystem.

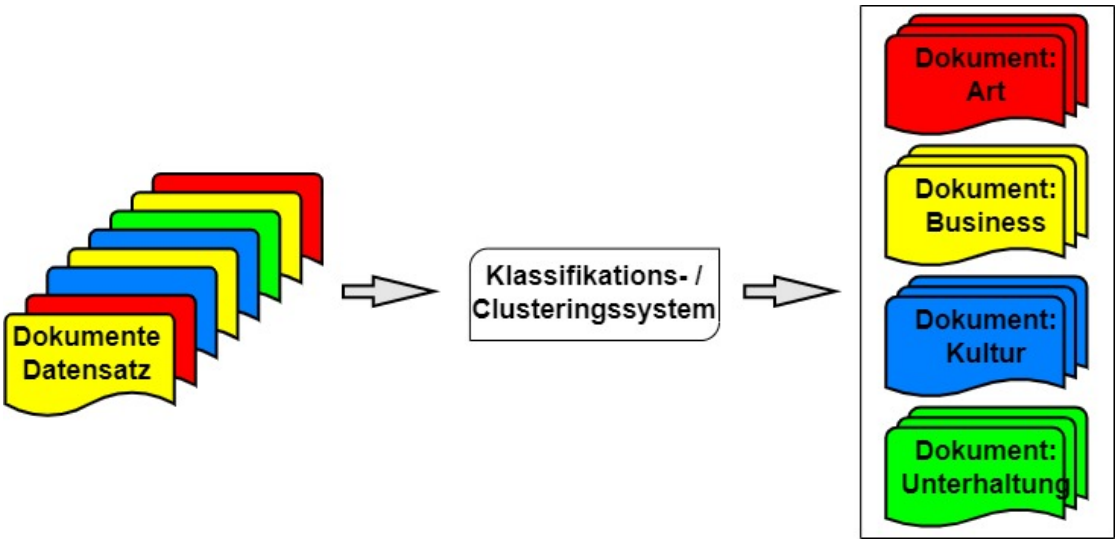


Abbildung 1.1: Klassifikationsmodell

Nicht allein im Privatsektor, sondern ebenso in Forschung und herstellendem Gewerbe werden jeden Tag Daten produziert. Darüber hinaus tragen ein stetig wachsendes Datenvolumen sowie Voraussetzungen hinsichtlich Kombination und Integration verschiedener Datensätze dazu bei, die Komplexität der Untersuchung zu erhöhen. Die leistungsfähige Anlieferung des Datenspeichervolumens kann z.B. von Seiten einer Organisation des Datenspeichers versprochen werden. Eine Begutachtung solcher Fakten erfordert dennoch ein erweiterbares Verfahren, das in der Lage ist, jene Datenmengen im Zuge eines akzeptablen Zeitraums zu analysieren. Zu diesem Zweck sind vollautomatische Analysen von

Datensätzen (*unsupervised analysis*) ohne Benutzereingriff vonnöten.
Eine vorstellbare Fragestellung innerhalb der Analyse von Datensätzen ist die Suche nach Gruppierungen von Daten, die ähnliche Charakterzüge aufweisen. Das Clustering ist eine Methodik zur Sammlung jener Ähnlichkeitsmaße von Datensätzen, im Zuge dessen mittels eines parametrisierten Algorithmus die erwähnten Gruppierungen in den Datensätzen entweder vollautomatisch oder mit Benutzerinteraktion identifiziert werden können. Somit besteht die Möglichkeit herauszufinden, wie gut dieses Verfahren für die Clusteranalyse bei den Hoch-Dimensionalen sind.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit ist in fünf Kapitel gegliedert. Diese orientieren sich an der vorgeschlagenen Zielsetzung. Wie in Abbildung 1.2 dargestellt ist, werden die fünft Kapitel durch drei Begriffe verbunden: Einführung (orange), Entwicklung (blau) und Ergebnisse (grün).

Abbildung 1.2: Aufbau der Arbeit

Kapitel 1 : Einleitung

In diesem Kapitel werden Motivation und Aufstellung dieser Arbeit beschrieben. Die Definition der Zielsetzung und Problemstellung wird ebenso angeschnitten. Die Planung, wie das Problem gelöst wird, und die entsprechende Vorgehensweise werden kurz beschrieben.

Kapitel 2 : Grundlagen / Theoretisches Wissen

Im zweiten Teil werden für die Zielsetzung jener Arbeit relevante Ausdrücke und Definitionen aus dem Themenfeld des maschinellen Lernens präsentiert. Wichtige Themen aus dem Bereich der Clusteranalyse sowie beträchtliche Verfahren, die in der Arbeit genutzt werden geschildert.

Kapitel 3 : Prozessablauf / Realisierung

Im dritten Kapitel wird zunächst der Ablauf des Konzepts durch Implementierung erläutert. Anschließend folgen die Beschreibung des Verfahrens sowie der dafür genutzte Datensätze.

Kapitel 4 : Analyse / Auswertung der Ergebnissen

Im Fokus des vierten Kapitels stehen die Bewertung der verschiedenen Experimente sowie ihre Visualisierung.

Kapitel 5 : Zusammenfassung und Ausblick

In Kapitel 5 wird das Fazit der Arbeit vorgestellt. Darüber hinaus wird ein Ausblick gegeben, wie die Analyse erweitert werden könnte.
Eine genauere Beschreibung der Unterkapitel wird jeweils zu Beginn des betreffenden Kapitels gegeben.

1.4 Zusammenfassung

In diesem Kapitel wurde ein Überblick über Motivation, Zielsetzung und Aufbau der Arbeit gegeben. Damit wurde das Ziel verfolgt, den Einstieg in das eigentliche Thema zu erleichtern. Im anschließenden Kapitel werden für die Arbeit relevante Begriffe und Definitionen vorgestellt.

2 Grundlagen / Theoretisches Wissen

In diesem Kapitel werden die Grundlagen und Lernmethoden des maschinellen Lernen vorgestellt, die für ein Verständnis dieser Arbeit relevant sind. Darüber hinaus werden alle im Zusammenhang mit dieser Arbeit stehenden CLustering-Verfahren in abgekürzter Form eingeführt. Nachfolgend werden die Herausforderungen des mehrdimensionalen Clusterings im Sinne von Ähnlichkeiten und Ergebnisvalidierung, welche in dieser Arbeit eingesetzt werden, beschrieben.

2.1 Maschinelles Lernen

Vor einer Erklärung des maschinelles Lehnens ist zunächst festzuhalten, wie das Lernen an sich definiert ist. Auch wenn der Begriff selbst jedem geläufig ist, kommen nach dem Autor des Buches [27] jene drei Definitionen in Betracht:

> *Jede Form von Leistungssteigerung, die durch gezielte Anstrengung erreicht wurde.*
> *Jede Verhaltensänderung, die sich auf Erfahrung, Übung oder Beobachtung zurückführen lässt.*
> *Durch Erfahrung entstandene, relativ überdauernde Verhaltensänderung bzw. Möglichkeiten.*

Im Prinzip gilt gleiches für die Maschine. Aus diesem Grund könnte maschinelles Lernen (engl. Machine Learning) unkompliziert als ein Themenbereich der Künstlichen Intelligenz definiert werden, der es Systemen erlaubt, auf Grundlage von Trainingsdaten eigenständig zu lernen und eigene Fähigkeiten zu erweitern. Maschinelles Lernen befasst sich mit der Realisierung lernfähiger Systeme und Algorithmen und verwendet unterdessen Konzepte und Methoden der Statistik und der Informationstheorie. Die Technik und Algorithmen des maschinellen Lernens können, wie unten dargestellt, in zwei bzw. drei Bereiche unterteilt werden: "Überwachtes Lernenünd Ünüberwachtes Lernen"je nach Art

des Lernens (eine weitere Variante ist "Bestärkendes Lernen"), sowie nach der Funktionsweise oder der Art der Modellierung. Abbildung 2.1 gibt einen Überblick der wichtigsten Methoden, gruppiert nach Art des Lernverfahrens.

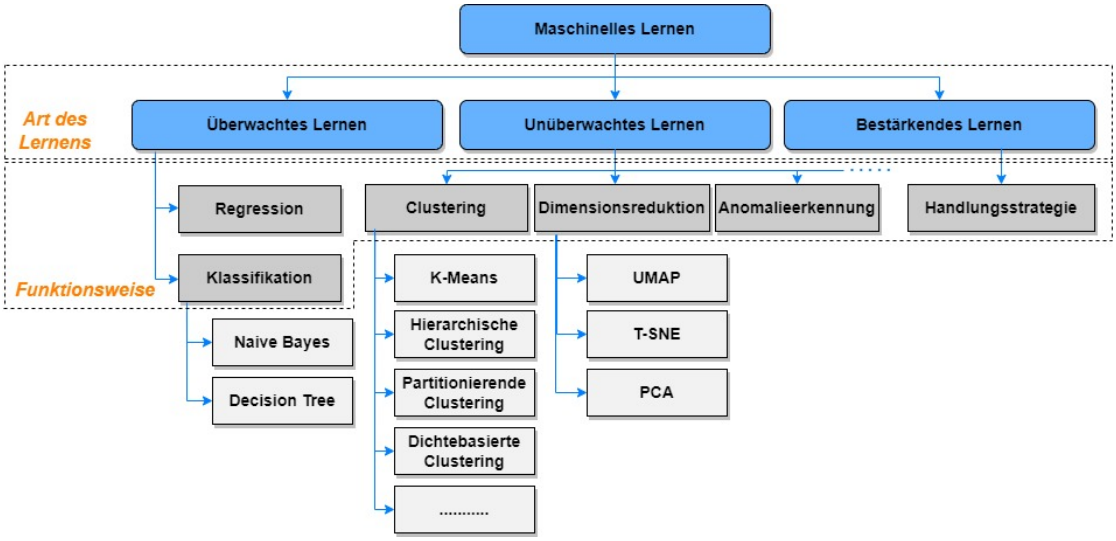


Abbildung 2.1: Methoden des maschinellen Lernens

2.1.1 Lernstile

Die Lernalgorithmen des maschinellen Lernens lassen sich im Wesentlichen in drei Kategorie einteilen: *Überwachtes Lernen*, *Unüberwachtes Lernen* und *Bestärkendes Lernen*, die jeweils einen anderen Prozess durchlaufen. Tatsächlich beinhalten all diese Typen das Erstellen (Lernen) einer mathematischen Funktion.

$$f : X \to Y$$

Der Unterschied besteht darin, welche Mengen X und Y beteiligt sind und wie die Daten beschaffen sein müssen, um diese Funktion zu lernen. Es ist wichtig, sich daran zu erinnern, dass das grundlegende Merkmal von Funktionen in der Mathematik darin besteht, dass einem Element X genau ein Element Y zugeordnet ist.

Überwachtes Lernen (supervised learning) bedarf eines Lehrers. Eine zugängliche Auswertung erfolgt für jeden Datensatz der Eingabedaten, d.h. die Daten wurden bereits in Gruppen eingeteilt. Der Ablauf ist in Abbildung 2.2 veranschaulicht und wird im Folgenden erläutert: Die Merkmale der zu trainierenden Eingabedaten wurden extrahiert und in Trainings- und Testdaten unterteilt. Ein Vorhersage-Modell wird via eines adäquaten Algorithmus auf Basis der Trainingsdaten erstellt, dessen Güte anhand von Leistungsindikatoren[1] bestimmt wird. Dieser Prozess wird in dieser Art mehrfach wiederholt, bis das Modell die angestrebte Leistung erreicht hat. Folgend wird es für die Vorhersage bzw. Klassifizierung von neuartigen Datensätzen genutzt.

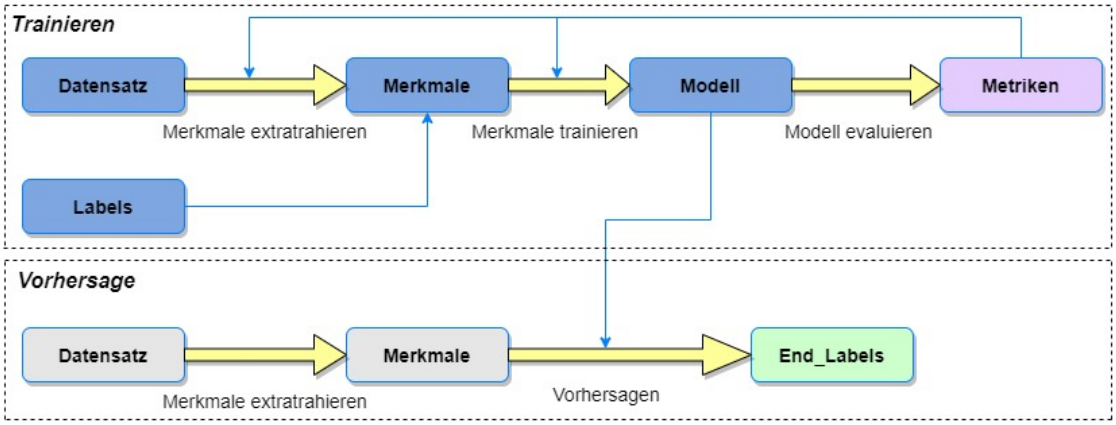


Abbildung 2.2: Ablauf des überwachten Lernens

Bestärkendes Lernen (reinforcement learning) In manchen Situationen ist nicht klar definiert, was richtig oder falsch ist. Daher stellt die Kennzeichnung von Informationen eine Herausforderung dar. So ist z.B. nicht erkennbar, was die perfekte Strategie ist, um ein Haus in kurzer Zeit und mit wenig Energie zu säubern. Bekannt ist hingegen, welches Ergebnis gewünscht und welches unerwünscht ist. In diesem Zusammenhang setzt bestärkenden Lernen intelligente Agenten[2] ein, die mit ihrer Umgebung interagieren und Strategie erlernen, um die erhaltenen Belohnungen zu maximieren. Beim Lernen mithilfe Verstärkung müssen die Eingabedaten nicht wie oben beim überwachten Lernen bewer-

[1] Ergebnisvalidierung wie zum Beispiel : Vertrauenswahrscheinlichkeit, Genauigkeit, Erfolgsquote
[2] Computersoftwaresystem, das in der Lage ist, separat zu handeln, um gezielte Ziele zu erreichen und auf Menschen oder Geschehnisse zu reagieren, die um es herum passieren.

tet werden stattdessen werden **Markovsche Entscheidungsprobleme**[3] verwendet, die mathematische Basis für verstärkendes Lernen sind. Sie werden in zusätzlichen Disziplinen analysiert und von Seitenalgorithmen der dynamischen Programmierung gelöst.

Unüberwachtes Lernen (unsupervised learning) Während im Fall von überwachtem Lernen dem Algorithmus eine Sammlung von Zielwerten zur Auswahl stehen und im Fall von bestärkendem Lernen die Lösungen eines Verhaltens positiv oder negativ bewertet werden können, existieren Fälle, in denen weder das eine noch das andere möglich ist. So liegt beispielsweise eine bestimmte Datenmenge vor, innerhalb derer mit unüberwachtem Lernen verborgene Strukturen in unmarkierten Daten identifiziert werden sollen. Der Lernprozess soll seitens das Erkennen von Mustern in den Datensatz ein zweckfreies Modell anlegen. Dafür sind Clusteranalysen geeignet, ein Verfahren zur Erkennung von Klassen ähnlicher Beobachtungen *(Clustern)* in einem Datensatz. Abbildung 2.3 verdeutlicht den Ablauf dieses Lernverfahrens des maschinellen Lernens.

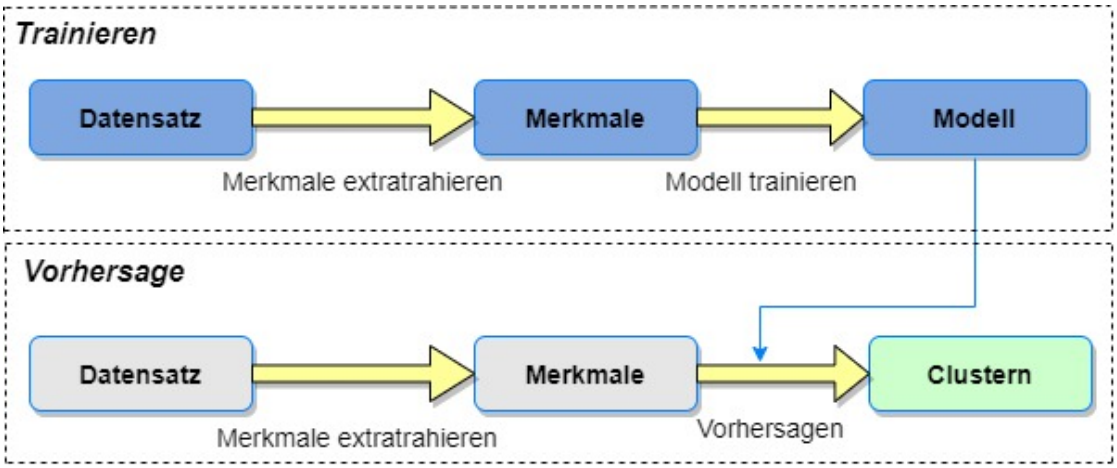


Abbildung 2.3: Ablauf des unüberwachten Lernens

[3] Probleme, binnen denen der Vorteil eines Agenten von einer Folge von Entscheidungen für Zustandsübergängen abhängig ist. d.h. die Wahrscheinlichkeit einen Zustand A von Zustand B aus zu erreichen

2.2 Kategorien von Clustering-Verfahren

Nachdem die grundsätzliche Definition von maschinellem Lernen und seine wichtigen Lernstile vorgestellt wurden, muss für eines besseres Leseverständnis dieser Arbeit ein Blick auf die Methoden bzw. Kategorien von Clustering-Verfahren geworfen werden. Bei der Clusteranalyse werden Distanz und Ähnlichkeit als Basis verwendet. Die Distanz dient dazu, die Beziehungen zwischen andersartigen Datenpunkten für die Quantität den Merkmale festzustellen, während für die qualitativen Merkmalen die Ähnlichkeit bevorzugt wird. Jeder verläuft über mehrere Etappen wie folgt:

- **Merkmalsextraktion und Merkmalsauswahl:** Zunächst werden die Merkmale für den zu analysierten Datensatz extrahiert und/oder ausgewählt.
- **Entwurf des Clustering-Algorithmus:** Dann wird der Algorithmus entsprechend den Eigenschaften der Problemstellung durchgeführt.
- **Ergebnisauswertung:** Außerdem, werden die erhaltenen Cluster evaluiert und die Validierung des Algorithmus beurteilt.
- **Ergebnisinterpretation:** Schließlich wird die praktische Erklärung für das Ergebnis des Clustering erläutert.

So existiert keine Algorithmus, der am besten funktioniert. In zahlreiche Fällen geht es darum, nach Testen von verschiedenartigen Parametern des Verfahrens eine Entscheidung zu Gunsten eines Verfahrens zu treffen. Untenstehend wurden die Clustering-Algorithmus kategorisiert.

Dichte-basierte Verfahren

Die Grundidee ist, dass Dateneingaben von einer Zone mit hoher Datendichte als zum selben Cluster gehörig betrachtet werden. Ein bekanntestes Verfahren der Dichte-basierten Clusteralgorithmen ist **DBSCAN** (Density-Based Spatial Clustering of Application with Noise) [10]. Dies ist als direkt dichtverbunden definiert (directy density connected) und mit folgender Ungleichung repräsentiert ($dist(p,q) \leq Eps$), wobei p und q zweier markierter Objekte miteinander verbunden sind. p wird als Kernpunkt (core point) bezeichnet, wenn eine Mindestanzahl an Punkten ($MinPts$) um p dichteverbunden sind. N_{Eps} (Eps-neighborhood) ist wie folgt definiert:

$$N_{Eps} = q \in D | dist(p, q) \leq Eps$$

Außerdem sind die Punkte p und q dichte-erreichbar (density reachable), wenn ein weiterer Punkt von beiden Punkten direkt dichteverbunden ist.

$$p \in N_{Eps}(q) \text{ und } |N_{Eps}(q)| \geq MinPts \text{ (core point condition)}$$

In diesem Verfahren entsteht ein Cluster mithilfe eines Kernpunkts und seiner Umgebung, die dichte-erreichbar ist. Abbildung 2.4 [28] stellt ein Beispiel eines dichte-basierten Verfahrens dar.

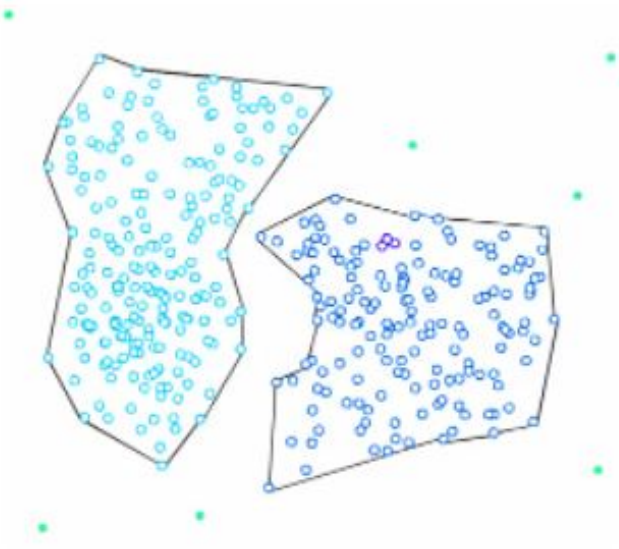


Abbildung 2.4: Beispielhafte Darstellung eines dichtebasierten Clusterings

Partitionierende Verfahren

Beim partitionierenden Clustering-Verfahren wird der Datenraum entweder aktiv oder passiv unterteilt. Das erfolgt passiv via Abgrenzung mit Hilfe Clustersdarstellung und aktiv von Seiten Integration von mehrdimensionalen Ebenen. Vom jeweilige Algorithmus werden diese Cluster (Die Anzahl an Cluster (K) wird zu Beginn abgemacht) bestimmt

und die Metrik der Datenpunkte zu ihrem nächstgelegenen Cluster minimiert. **K-Means** und **K-Mediods** zählen zu den verbreitesten dieser Algorithmen. Ein gefundenes Cluster C_i wird per virtuell gebündeltem Punkt (Centroid) m_i und den Punkten repräsentiert, die exakt an diesem Centroid anschließenden [1]. Die Kernidee ist, das Zentrum des Clusters per iterativer Berechnung unaufhörlich zu aktualisieren. Das Verfahren beschränkt sich auf zwei Schritte. Diese werden wiederholt, bis ein Konvergenzkriterium getroffen wird.

- Das Cluster wird durch Zuweisung aller Instanzen zu einem Centroid, der von einem Datenpunkt zugewiesen wird, erstellt.
- Der Centroid m_i wird durch neu hinzukommende Instanzen aus dem vorherigen Schritt via nachfolgender Gleichung neu kalkuliert.

$$\forall d \in D : m_i^{(d)} = \frac{1}{|C_i^{(d)}|} \cdot \sum_{X \in C_i} X^{(d)}$$

Die alternative Ausführungen des K-Means Algorithmus ist bspw.: **K-Medians**, **Fuzzy K-Means**, **Intelligent K-Means** (IK-Means) und **Kernel K-Means**, die in dieser Arbeit nicht behandelt werden. Ein Beispiel von partitionierenden Verfahren wird in Abbildung 2.5 [28] dargestellt.

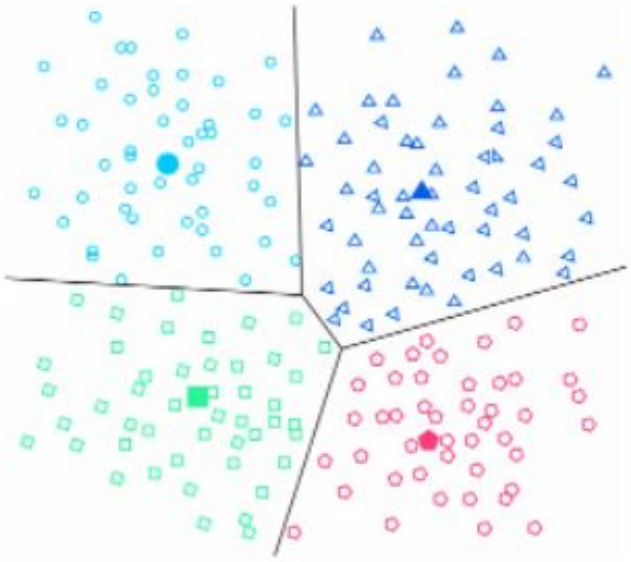


Abbildung 2.5: Beispielhafte Darstellung eines partitionierenden Clusterings

Hierarchische Verfahren

Die Zielsetzung jeder Art von Clustering-Algorithmen besteht darin, die hierarchische Beziehung zwischen den Daten zu rekonstruieren, um sie zusammenzustellen. Um dieses Ziel zu erreichen, wird entweder die Gesamtheit aller Datenpunkte als ein einziges großes Cluster betrachtet (**top-down**), oder die Menge aller Datenpunkte wird als ein einziges Cluster für jeden Datenpunkt angenommen (**bottom-up**). Diese Beschreibung unterscheidet zwei Sorten von hierarchischen Verfahren.

- **Divisive Clusterverfahren**: Zu Beginn sind sämtliche Bestandteile in einem großen Cluster. Daraufhin wird ein solches Cluster Schritt für Schritt verkleinert. Letztendlich wird jedes Cluster genau ein Objekt beinhalten.

- **Agglomerative Clusterverfahren**: Im Gegensatz zum divisiven Clustering entspricht anfänglich jedes Objekt einem speziellen Cluster. Dann bilden schrittweise zwei Cluster nach Ähnlichkeit eine neues Cluster. Das Prinzip des agglomerativen Verfahrens lässt sich in drei Schritten darstellen : [7]
  1. Jedes Objekt aus der Menge $I = I_1, ..., I_N$ stimmt in einem Cluster überein, d.h. die Anfangspartition ist $C^{(0)} = I_1, ..., I_N$.
  2. Die Partition $C^{(v)}$ mit $(v \geq 1)$ wird durch Zusammenlegung von zwei Gruppen aus der Partition $C^{(v-1)}$ gebildet (v bezeichnet die Anzahl von Schritten der Iteration).
  3. Der Schritt 2 wird iteriert, bis nur noch ein Cluster besteht $(C^{(v)} = I)$.

Diese Kategorie von Clusterverfahren umfasst noch weiteren Algorithmen wie z.B: **DIANA**, **CURE**, **CHAMELEON**, **BIRCH**, **ROCK**. In Abbildung[4] 2.6 wird ein sogenanntes Dendrogramm[5] für die beide Verfahren veranschaulicht.

[4] *https : //archiv.ub.uni − heidelberg.de/volltextserver/*20182/1/*dissertation*$_m$*aximilianhoecker.pdf*
[5] Grafische Darstellung für Zusammenführen von Elementen einer hierarchischen Clustering-Analyse in einer baumartige Struktur, die Elemente als Punkte auf einer Achse und Skalar des Distanzindex auf den andere Achse hat.

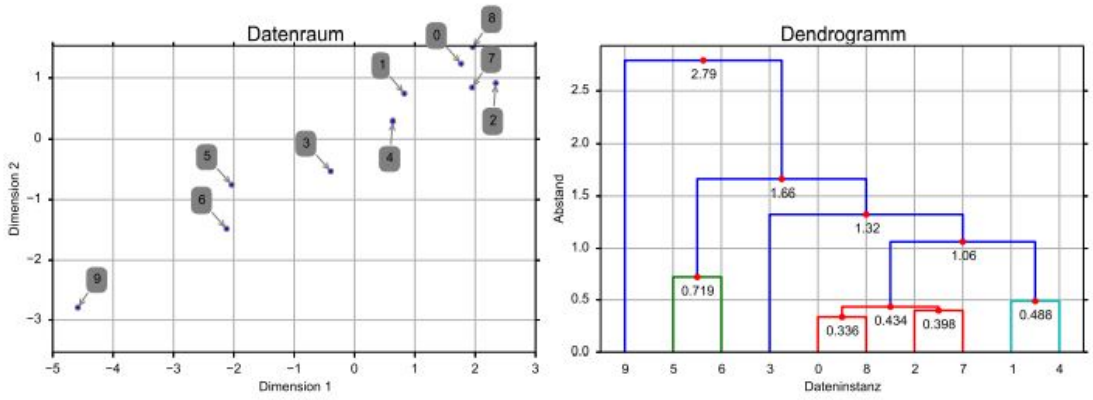


Abbildung 2.6: Beispielhaftes Dendrogramm zur Darstellung eines hierarchischen Clusterings

Wahrscheinlichkeit-basierte Verfahren

Wie die oben beschriebenen Algorithmen zielen wahrscheinlichkeitsbasierte Verfahren ebenfalls darauf ab, Datenpunkte zu gruppieren (*Clustern*). Aber es geht in diesem Fall von der Wahrscheinlichkeit aus, dass ein bestimmtes Objekt zu einem bestimmten Cluster gehört. Das Verfahren zeichnet sich zudem durch einen iterativen Ansatz aus und kann nur auf numerische Daten angewendet werden. Somit entspricht ein Cluster einer Kombination von parametrisierten Wahrscheinlichkeitsdichtfunktionen[6] (Propability Density Function, PDF), die durch folgende Gleichung zum Ausdruck kommt.

$$Pr[a \leq x \leq b] = \int_a^b f(x)\mathrm{d}x \text{ mit } \int_{-\infty}^{\infty} f(x)\mathrm{d}x = 1$$

Die Menge aller Cluster stellt eine Liste parametrisierter Wahrscheinlichkeitsdichtfunktionen dar, die häufig erforderlich sind, um ein flexibles Modell anzupassen, das einen Algorithmus verwendet, um die Parameter der Cluster zu lernen. Modelle in Form von Kombinationen der o.g. Wahrscheinlichkeitsdichtfunktionen bestehen aus sogenannten Mischverteilungsmodellen[7] (Mixture Models, MM), die aus der Summe der Komponenten bestehen. Eine einzelne Komponente kann auch durch eine bestimmte Verteilung dargestellt werden, deren Parameter von Komponenten abhängen, und jede Komponente

[6] Wahrscheinlichkeit, dass der Wert einer stetigen Zufalls-variablen in einen Bereich fällt

[7] Probabilistische Modell, das die Erzeugung von sämtlichen Datenpunkte aus einer Mischung einer endliche Anzahl an Verteilung mit fremdartigen Parameter voraussetzt

kann durch ihre eigene Verteilung dargestellt werden. Die bekannten Modelle sind das **Gauss'schen Mischverteilungsmodell** (Gaussian Mixture Modell) sowie das **Bernoulli Mischverteilungsmodell** (Bernoulli Mixture Modell).
Die Modelle werden anhand des **Expectation Maximization Algorithmus** [8]. (EM) zufällig initialisiert und mithilfe des Algorithmus in dieser Art verbessert, dass jede Komponente des Mischverteilungsmodells ein Cluster darstellen kann. Beim EM Algorithmus schwankt die Komplexität der Berechnung je nach implementierter Wahrscheinlichkeitsdichtfunktion. Dank der Vielseitigkeit der Mischverteilungsmodellen ist es mit entsprechendem Rechenaufwand möglich, ebenso mehrdimensionale Cluster zu berechnen.

2.3 Herausforderung beim hochdimensionalen Clustering

Hochdimensionale Daten werden nicht ausschließlich im Industriebereich, sondern ebenso in de Wissenschaft gebraucht. In wissenschaftlichen Forschungsfeldern liegen z.B. Finanzdaten, Sensordaten, Informationen aus Gesundheitswesen oder Netzdokumenten vor. Der Begriff hochdimensionale Daten wird, als Datensatz verstanden, innerhalb dessen die Anzahl der Wesensmerkmale P größer ist als die Menge der Beobachtungen N ($P> \ N$) [9].S.649. Die Algorithmen, die mit hochdimensionalen Daten wirken, gehen davon aus, dass die Menge der Dateninstanzen kleiner als die Zahl der Dimensionen ist. Problematisch ist, dass ein großer theoretischer Suchraum existiert, in dem Cluster gefunden werden können. So gibt es z.B. in einem Suchraum mit 100 Dimensionen bestehend aus einer jeweils diskreten Skalar und Abstufung denkbaren Werten bereits 100^{100} verschiedene Suchpunkte. In dem Bereich der Mustererkennung wird eine detaillierte Disputation dieses Effekts in [11].S.33. beschrieben. So werden in den anschließenden Abschnitten über die differenzierten anderen Problemstellungen, die das Clustering von hochdimensionalen Daten liefert, debattiert.

2.3.1 Distanzmaße und Ähnlichkeitsmaße

Die Distanzmaße und Ähnlichkeitsmaße sind definiert, um die Ähnlichkeit inmitten von Objekten bzw. nebst Mengen zu ermitteln. Der Unterschied liegt darin: Je ähnlicher sich zwei Objekte oder zwei Mengen sind, desto größer ist der Wert des Ähnlichkeitsmaßes, wohingegen der Wert des Distanzmaßes desto kleiner sein sollte.

Clusterdistanz

Die Autoren von [10].S.38. beschreiben eine Clusterdistanz als *eine Funktion dist:* $\epsilon \times \epsilon \rightarrow \mathbb{R}_+$*, welche nicht von der Reihenfolge der Objekte in den Datensätzen abhängt, d.h. für alle Datensätze* $X, Y \in \epsilon$ *und alle Permutationen* $\sigma \in S_n, T \in S_m$ *gilt*:

$$dist((x_1, ..., x_n), (y_1, ..., y_m)) \qquad dist((x_{\sigma(1)}, ..., x_{\sigma(n)}), (y_{\tau(1)}, ..., y_{\tau(m)})).$$

Sicherlich existieren viele Optionen, den Abstand nebst Clustern bzw. Datensätzen zu definieren. Die bekanntesten Clusterdistanzen bestimmen den Abstand von Clustern des Abstands ihrer Objekte voreinander. Größere Objekten führen zu größeren Clusterabständen, kleinere Abstände zu kleineren Clusterabständen. Beispielsweise werden die gleichnamigen agglomerativen Clusteranalyseverfahren mit den drei Clusterdistanzen (*Single-Linkage*, *Complete-Linkage*, *Group-Average*) [2].S.82. [4].S.47. erfasst.

- Die ***Single-Linkage-Distanz*** zweier Cluster $X = (x_1, ...x_n)$ und $X = (y_1, ...y_m)$ misst den Clusterabstand seitens des minimalen Abstands zwischen den Objekten der Cluster X und Y und wird wie folgt definiert:

$$dist(X, Y) : \quad \inf_{i \leq n, j \leq m} \|x_i - y_j\|$$

- Das andere Extrem stellt die ***Complete-Linkage-Distanz*** dar. Diese repräsentiert die Definition des Abstands nebst X und Y als Distanz zu den Objekten, die am weitesten voneinander entfernt sind.

$$dist(X, Y) : \quad \sup_{i \leq n, j \leq m} \|x_i - y_j\|$$

  Beide oben genannten Linkage-Distanzen stehen nicht im Einklang mit der *geometrischen Anschauung*. Während die Single-Linkage-Distanz oft als zu klein wahrgenommen wird, wird die Complete-Linkage-Distanz andererseits als zu groß empfunden.

- Als Ausweg aus diesem Dilemma fungiert die ***Group-Average-Distanz***, welche die Distanzen der Objekte aus X und Y mittelt.

$$dist(X, Y) : \quad \frac{1}{n \cdot m} \sum_{i \leq n, j \leq m} \|x_i - y_j\|$$

Zwischen Complete-Linkage-Distanz und Single-Linkage-Distanz befindet sich die Group-Average-Distanz von zwei Clustern. Sie ist kleiner als die Complete-Linkage-Distanz und gleichzeitig größer als Single-Linkage-Distanz.

Ähnlichkeitsdefinition

Daten diverser Cluster sind nicht ähnlich zueinander. Das Abstrahieren des Begriffs bewirkt eine Sinngebung in der Fragestellung der Ähnlichkeit. In der Umgebung des Clusterings von hochdimensionalen Datensätzen ist Ähnlichkeit in verschiedenartigen Verfahren und Anwendungsbereichen anders definiert. So z.B. wird Ähnlichkeit als Nähe bzgl. der Euklidischen Distanz definiert. Für die folgende Gleichung der *Minkowski Distanz*

$$dist_{ij} = (\sum_{k=1}^{d} |x_{ik} - x_{jk}|^q)^{\frac{1}{q}}$$

$dist_{ij}$ repräsentiert die Distanz zwischen zwei Dateninstanzen x_i und x_j. Die Distanz ermittelt sich aus der oben definierten Formel. Die getrennten Clustering-Kategorien verfügen über eigene Notationen bzw. Definitionen von Ähnlichkeit. Aufgrund des Zusammenhangs inmitten Algorithmus und Definition können jene durchaus nicht nebst den getrennten Kategorien von Verfahren gewechselt werden [18] .
Bspw. gibt es den Bereich der auf Distanz basierten Algorithmen, die mit Konzepten wie Nähe [19] oder Nachbarschaften [6] wirken. Mit großer Anzahl an Dimensionen können ebendiese Algorithmen weniger gewissenhafte Lösungen liefern. Die Vertrauenswürdigkeit der Ergebnisse wird mit folgender Gleichung dargestellt.

$$d \to \infty, f = \frac{dist_{max} - dist_{min}}{dist_{min}} \to 0$$

Diese Gleichung vermittelt den Effekt innerhalb wachsender Dimensionalität d des Datensatzes innerhalb distanzbasierter Algorithmen den höchstmögliche Abstand *distmax* wie auch das minimale Gegenstück auf den gleichen Wert, die in einem Punkt zusammenlaufen, was die Lösungen qualitativ schmälert.

2.3.2 Ergebnisvalidierung

Die Validierung der Lösungen einer Clustering-Analyse (*Cluster Validity*) ist mit der Definition der Ähnlichkeitsmetrik verbunden. Weil es in dieser Arbeit um die Erstellung von Clustern mit nicht-überwachten Algorithmen geht, verfügen sie über eine Ergebnisqualität, die von ihrer impliziten oder expliziten Ähnlichkeitsmetrik abhängt.Für die Validierung von Clusteringergebnissen in erster Regel hinsichtlich der Anzahl der erhaltenen Cluster existieren viele diverse Methoden. Im Folgenden werden einige der häufigsten wie in [24] erläutert.

Gap Statistic

Die Gap-Statistik hält gegeneinander die Summe innerhalb der Intra-Cluster-Variation für andersartige Werte. Die Vorhersage der perfekten Anzahl an Clustern ist ein Wert, der die größte Lückenstatistik ergibt. Dies bedeutet, dass die Clusterstruktur weit von der zufälligen gleichmäßigen Verteilung der Punkte weggerückt ist. Der Algorithmus läuft wie folgt ab:

- Führen Sie die Clusteranalyse aus und variieren Sie die Zahl der Cluster von $k = 1, ..., k_{max}$, und berechnen Sie die korrespondierende Summe im Zuge der clusterinternen variation W_k.
- Mit zufälligen gleichmäßigen Verteilung generieren Sie Datensätze von B-Referenz. Clustern Sie jeden jener Referenzdatensätze mit diverser Anzahl von Clustern $k = 1, ..., k_{max}$ und berechnen Sie die korrespondierende Summe binnen der clusterinternen Variation W_{kb}.
- Die geschätzte Lückenstatistik wird jetzt als Abweichung des beobachteten W_k Wertes aus seinem erwarteten Wert W_{kb} unter der Nullhypothese berechnet: $Gap(K) \quad \frac{1}{B} \cdot \sum_{b=1}^{B} \log(W_{kb}^*) - \log(W_{kb})$. Berechnen Sie gleichfalls die Standardabweichung der Punkt einfügen
- Am Ende wird die Menge der Cluster als kleinster Wert von k gewählt, in dieser Art, dass die Lückenstatistik während einer Standardabweichung der Lücke bei *k+1* liegt: $Gap(k) \geq Gap(k+1) - S_{k+1}$.

Calinski-Harabasz-Index

Das Varianzverhältniskriterium (Calinski-Harabasz-Index) ist vergleichsweise zügig zu berechnen. Es wird in die Relation der Varianz der Quadratsummen der Abstände einzelner Objekte zu ihrem Clusterzentrum durch die Summe der Distanz nebst den Clusterzentren und dem Mittelpunkt der Daten eines Clusters unterteilt. Ein gutartiges Clusterergebnis weist währenddessen einen hohen *Calinski-Harabasz-Wert* auf. Die Punktzahl ist allgemein höher, wenn die Cluster dicht und in Ordnung alleinig sind, wie es dem Standardkonzept eines Clusters entspricht.

Silhouette Coefficient

Der durchschnittliche Silhouettenansatz ermittelt, wie in Ordnung jedes Objekt in seinem Cluster befindet. Eine hohe durchschnittliche Silhouettenbreite weist auf eine gute Clusterbildung hin. Die Silhouettenmethode ermittelt die durchschnittliche Silhouette von Beobachtungen für divergente Werte von k. Die bestmögliche Anzahl von Clustern k ist diejenige, die die durchschnittliche Silhouette über einen Bereich machbarer Werte für k maximiert.

- Clustering-Algorithmus für verschiedenartige Werte von k ausführen lassen. Wie z.B. indem k von *1* bis *10* Clustern schwankt wird.
- Kalkulieren Sie für jedes k die durchschnittliche Silhouette der Beobachtungen.
- Zeichnen Sie die Kurve der Berechnung gemäß der Anzahl der Cluster k.
- Die angemessene Anzahl von Clustern betrachtet die Position des Maximums.

Davies-Bouldin Index

Der Davies-Bouldin-Index bildet den Vergleich von intra- und inter-Cluster-Varianzen. Dieser Index wird wie [3], in verschiedenen Schritten definiert.

- Erstmals wird die intra-Cluster-Dispersion mit folgender Gleichung, berechnet:

$$S_i = \left\{ \frac{1}{T_i} \sum_{j=1}^{T_i} |X_j - A_j|^q \right\}^{\frac{1}{q}}$$

wobei:

i: identifizierter Cluster
T_i: Anzahl von Beobachtungen im Cluster i
X_j: j-ter Beobachtung im Cluster i
A_i: Schwerpunkt des Clusters i

Die intra-Cluster Dispersion wird durch grundsätzliche Berechnung des durchschnittliche Abstands zwischen jeder Beobachtung des Clusters und seinem Schwerpunkt ermittelt und dazu q auf 2 (q 2) gesetzt.

- Dann wird das Trennmaß nach dem Autor von [3] mit der Gleichung berechnet:

$$M_{ij} = \left\{ \sum_{k=1}^{N} |a_{ki} - a_{kj}|^p \right\}^{\frac{1}{p}} = \|A_i - A_j\|_p$$

mit:

a_{ki}: k-te Komponente von n-dimensionaler Schwerpunkt
a_{kj}: k-te Komponente von n-dimensionaler Schwerpunkt i
N: Gesamtzahl an Clustern

- Danach wird die Ähnlichkeit zwischen Clustern berechnet. Diese wird als Summe von zwei intra-Cluster-Dispersionen berechnet, die durch das Trennmaß dividiert werden.

$$R_{ij} = \frac{S_i + S_j}{M_{ij}}$$

S_i: Intra-Cluster-Dispersion des Clusters i
S_j: Intra-Cluster-Dispersion des Clusters j
M_{ij}: Abstand zwischen Punkten von Cluster i und j

- Anschließend wird der ähnlichste Cluster für jeden Cluster i gefunden, der nach [3] mit

$$R_i \equiv max(R_{ij}) \quad \text{mit} \quad i \neq j$$

definiert ist.

- Ausschließlich wird der Davies-Bouldin-Index berechnet. Er erfasst den Durchschnitt der Ähnlichkeitsmaße jedes Clusters mit einem Cluster, das ihm am besten ähnelt.

$$\bar{R} = \frac{i}{N} \sum_{i=1}^{N} R_i$$

Dunn Index

Auch Dunn's Index verfolgt die Idee, den Zusammenhang zwischen Trennung und Nähe einzelner Cluster zu vergleichen

$$DI(C_k) = min_{i=1,\ldots,k} \left\{ min_{j=i+1,\ldots,k} \left(\frac{d_C(C_i,C_j)}{max_{k=1,\ldots,k}(\Delta(C_k))} \right) \right\}$$

mit $d_C(C_i, Cj) = min_{x \in C_i, y \in C_j} d(x,y)$ und $\Delta(C) = max_{x,y \in C} d(x,y)$
d(x,y) ist ein Maß für den Abstand zwischen zwei Clustern, und $\Delta(C)$ bestimmt den Durchmesser jedes Clusters, mit anderen Worten die *Ausbreitung* jedes Clusters. Das Ziel ist es, den Dunn-Exponenten zu maximieren, da größere Werte eine gute Trennung und Homogenität bedeutet.

Elbow Criterion

Das Elbow-Kriterium dient auf besondere Weise dazu, um während einem K-means-Clustering die perfekte Anzahl von Clustern zu messen. In diesem Zusammenhang werden in einem Diagramm auf der x-Achse die Nummer der Cluster, auf der y-Achse die Summe der quadrierten Abweichungen der geteilten Punkte zum passenden Clusterzentrum aufgetragen. Bei diesem Punkt handelt es sich um die tadellose Anzahl an Clustern, wenn in dem in dieser Art entstandenen Diagramm ein Knick (*Ellbogen*) wahrnehmbar sein sollte. Ab diesem sinkt bekanntermaßen die Aussagekraft der geteilten Cluster, weil sich die Summe der quadrierten Abweichungen allein noch mühelos ändert. Die Nutzung jener Methode *Knies einer Kurve* als Grenzpunkt ist eine gängige Vorgehensweise in der mathematischen Verbesserung zur Wahl eines Punktes, an dem abnehmende Erträge die zusätzlichen Aufwendungen nicht länger wert sind. Das bedeutet in der Clusteranalyse die Auswahl einer Anzahl von Clustern, im Zuge derer das Hinzufügen von sonstigen Clustern keine wesentlich bessere Modellierung der Daten ergibt. Die bestmögliche Anzahl von Clustern kann wie folgt definiert werden:

- Clustering-Algorithmus (z. B. k-Means-Clustering) für unterschiedliche Werte von k. z. B., indem k von 1 bis 10 Clustern schwankt.

- Es wird dann für jedes k die Gesamtsumme des Quadrats im Zuge des Clusters berechnet.
- Danach zeichnen Sie die Kurve von dieser Gesamtsumme gleichartig der Anzahl der Cluster k.
- Ausschließlich die Lage einer Biegung (Knie) wird im Allgemeinen als Indikator für die vergleichbare Anzahl von Clustern betrachtet.

2.4 Dimensionsreduktion

Das Endziel der Dimensionsreduktion besteht darin, einen hochdimensionalen Datensatz in einen geringerwertig dimensionaleren Raum zu überführen ohne währenddessen wichtige Informationen für das übrige Data-Mining zu verlieren. In diesem Abschnitt werden andersartige Ansätze zur Reduzierung von Dimensionen betrachtet. Ihre Leistung wird auf der Basis der auf die Daten angewendeten Ergebnisse von durchschnittlichen Clustering-Algorithmen bewertet. All jene Algorithmus sind unüberwacht, d. h., sie brauchen keine Markierungen. Die verbesserte Leistung der Clustering-Algorithmen wird genauso ermöglichen, das Clustering zu visualisieren. Im Bereich *Unüberwachtes Lernen* existieren andersartige Algorithmen, die eine Dimensionsreduktion realisieren. In diesem Rahmen soll indes der Fokus auf drei verschiedenartigen Verfahren gelegt werden (***PCA***,***T-SNE***, ***UMAP***), die in dieser Arbeit genutzt werden.

2.4.1 T-Distributed Stochastic Neighbor Embedding (T-SNE)

T-distributed Stochastic Neighbor Embedding (T-SNE) ist eine nichtlineare Dimensionsreduktionstechnik zur Visualisierung von mehrdimensionalen Clustern. Für hochdimensionalen Datensätze gibt dieses Verfahren eine visuelle Ausgabe je nach Präferenz in zwei oder drei Dimensionen aus. Im Gegensatz zu SNE ist t-SNE bequemer zu optimieren und liefert bessere optische Ausgaben [15]. Kritisch zu betrachten ist die Ausgabe von t-SNE. Sie liefert visuell einen Zusammenhang von Daten. Dennoch ist es schwierig, genauere Informationen herauszulesen. Zusätzlich bildet die Laufzeit einen weiteren Nachteil von t-SNE. Nach [15] wird der Standard-Algorithmus verwendet. In dieser Art hat jener eine Laufzeit in $\Theta(p^2)$ mit p als die Nummer an Datenpunkten. Außerdem kann dargestellt

werden, dass T-SNE nicht für jede Eingabe gegen ein globales Optimum konvergiert. Somit kann es vorkommen, dass für allein geringfügig weitere Eingabedaten stark divergente Ausgaben erzeugt werden.

2.4.2 Principal Component Analysis (PCA)

Die Autoren [20] verstehen die Principal Component Analysis (PCA) als eine technologische lineare Transformation, die eine orthogonale Transformation verwendet, innerhalb derer das symmetrische innere Produkt der Informationen erhalten bleibt. Ein Satz korrelierter Variablen wird in kleinere Gruppe von unkorrelierten Variablen umgewandelt. Der Prozess der Umwandlung von Wissen in weniger Dimensionen enthält die Normalisierung der Informationen, die Generierung einer Kovarianzmatrix sowie die Extraktion von Top-k-Eigenvektoren, die die Dimensionen der Neuschöpfung repräsentieren. Darüber hinaus sind die Vereinfachung, Reduzierung, Modellierung und Klassifizierung der Daten Ziele von PCA. In der Praxis wird eine Dimensionsreduktion eines mehrdimensionalen Raumes auf einen zwei- oder dreidimensionalen gewählt. Mit Hilfe des Ansatzes der Dimensionsreduktion besteht die Möglichkeit große Datensätze per PCA zu visualisieren. Diese Visualisierung beinhaltet überwiegend Gruppierungen der Datenpunkte aus dem Datensatz. Mit Hilfe jene Gruppierung ist es glaubwürdig Ausreißer zu identifizieren, weil jene keiner der Gruppen zugeordnet werden und somit einen visuellen Abstand zu allen Gruppen aufweisen.
Ein Nachteil von PCA besteht in der Laufzeitkomplexität der Berechnung der Dimensionsreduktion. Diese wird durch das Lösen eines Gleichungssystems kalkuliert, mit f als der Anzahl an Dimensionen und p als der Zahl an Datenpunkten, in $\Theta(f^2 \cdot p)$, und die Berechnung der Zerlegung der Eigenwerte in $\Theta(f^3)$. Dadurch liegt die Gesamtlaufzeit von PCA in $\Theta(f^2 \cdot p + f^3)$ [21] .
Bei Visualisierung von PCA ist die Dimensionsreduktion ein weiterer Nachteil. Vor allem für mehrdimensionale Daten kann eine Reduktion auf zwei oder drei Dimensionen eine zu starke Restriktion darstellen. Daher können aufgrund der starken Beschränkung der Dimensionen womöglich Abhängigkeiten der Features zueinander nicht visualisiert werden. Dies kann darin enden, dass Visualisierungen entstehen, innerhalb derer keine Eingruppierungen der Daten zu identifizieren sind [16].

2.4.3 Uniform Manifold Approximation and Projection (UMAP)

Uniform Manifold Approximation and Projection (UMAP), wie in [26] beschrieben, ist wahrhaftig eine alternative Technik zur Dimensionsreduktion und Visualisierung, vergleichbar wie T-SNE, jedoch im Vergleich zu T-SNE ist UMAP ein beliebteres Verfahren zur Dimensionsreduktion. Es erleichtert genauso eine nichtlineare Reduktion. Hinter diesem Algorithmus steht ein komplex mathematischer Kontext. Um es dennoch einfach zu formulieren, konstruiert UMAP eine Darstellung der hochdimensionalen Graphen und testet eine optimierung von niedrigdimensionalen Graphen, dass sie eine ähnliche Struktur wie hochdimensionale aufweisen. UMAP kann in zwei Hauptschritte aufgeteilt und wie in Abbildung 2.7 in noch kleinere Komponenten zerlegen werden. Die Abbildung zeigt, wie bei der Analyse jeder Abschnitt durchgelaufen wird.

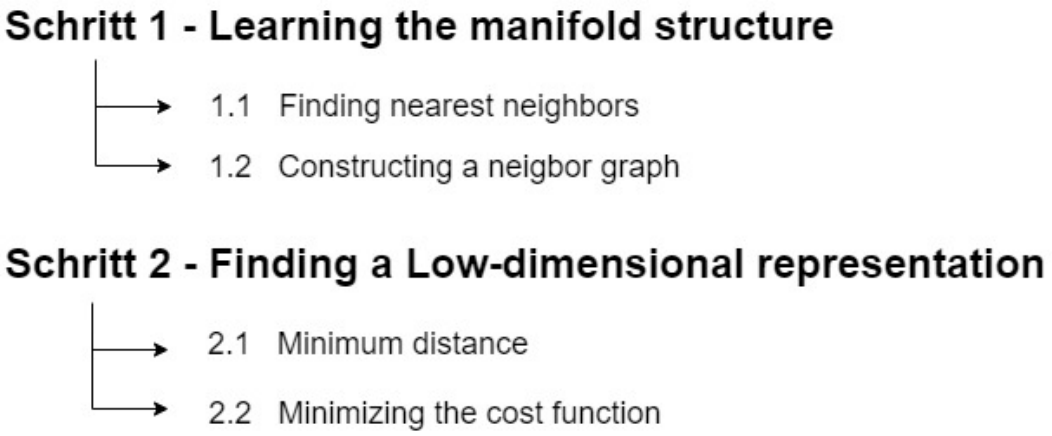


Abbildung 2.7: UMAP-Schritte und -Komponenten

2.5 Zusammenfassung

Für ein besseres Verständnis dieser Arbeit wurden in diesem Kapitel zunächst die Lernstile des maschinellen Lernens verdeutlicht. Zweitens wurden die Herausforderungen beim hochdimensionalen Clustering und beträchtliche Kategorien von Clustering-Verfahren präsentiert. Letztlich wurden die drei Algorithmen von Dimensionsreduktion, die in dieser Arbeit eingesetzt wurden, erläutert. Im nächsten Kapitel werden der gesamte Prozessablauf sowie die Realisierung der Arbeit erklärt.

3 Prozessablauf / Realisierung

Das folgende Kapitel befasst sich mit dem Prozessablauf (szs. Realisierung) dieser Arbeit. Weiterhin werden die verschiedenartigen Datensätze, die für die Experimente in dieser Arbeit genutzt werden, geschildert. Im Anschluss wird die Analyse durch die Implementierung von in dieser Arbeit verwendeten Clusteralgorithmen erklärt.

3.1 Ablauf

Der gesamte Prozess der Realisierung der Analyse, die ebendiese Arbeit behandelt, ist wie in Abbildung 3.1 in verschieder Analyseschritte Punkt einfügen. Diese gelten für jedes in diese Arbeit analysierte Clustering-Verfahren.

Start des Algorithmus: Zunächst wird der Algorithmus mit dem zu analysierenden Datensatz gestartet. Entsprechend des Datensatzes wird entschieden, ob eine Merkmalsextraktion bzw. Standardisierung[1] der Daten erforderlich ist.

Merkmalsextraktion: Bei der Clusteranalyse ist stets, essentiell mit guten und für den Bereich der Klassifizierung vorbereiteten Datensätzen zu arbeiten, um die Performance des Algorithmus zu steigern. Aufgrunddessen werden im zweiten Schritt die Merkmale des Datensatzes extrahiert, wenn dies nötig ist.

[1] besteht in der Umwandlung von der Normalverteilung in der Standardnormalverteilung, in dem einen *Z-Wert* aus den Daten berechnet wird.

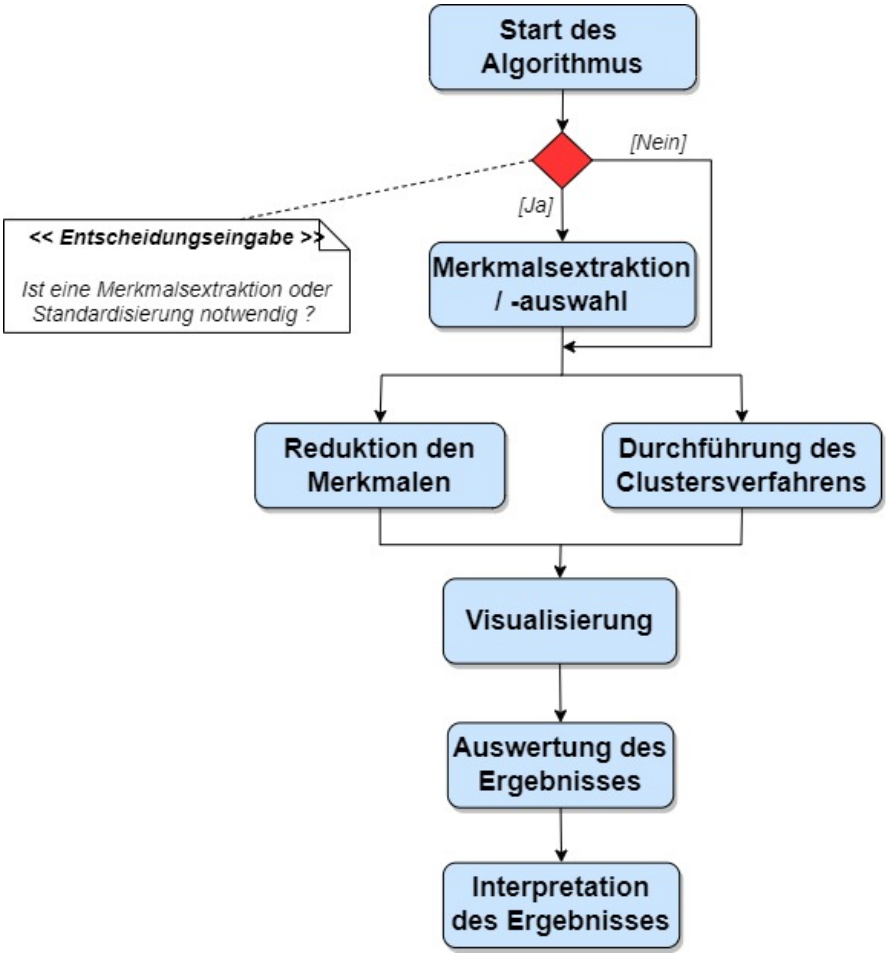


Abbildung 3.1: Ablauf der Realisierung der Analyse

Reduktion der Merkmalen und Entwurf des Clusterverfahrens: Nach dem Start des Algorithmus mit geeignetem Datensatz werden einerseits die Merkmale dieses Datensatzes mit den Algorithmen der Dimensionsreduktion (*PCA*, *T-SNE* und *UMAP*) auf zwei Merkmale reduziert. Der Datensatz wird andererseits mit dem Clusterverfahren analysiert. Das Resultat der Dimensionsreduktion wird für die Visualisierung des Clusterergebnisses verwendet.

Visualisierung: Daraufhin wird das Ergebnis des Clusterverfahrens im Zweier-Dimension Diagramm dargestellt. Die Visualisierung der Clusteranalyse zeigt die Darstellung des Datensatzes in Gruppen (auch *Clustern* genannt). Auf dem Graph sind drei andersartige

Darstellungen zu identifizieren, die der Visualisierung mit den drei Dimensionsreduktionen entsprechen.

Auswertung des Ergebnisses: Im Folgenden wird das Ergebnis ausgewertet. Hierbei werden unterschiedliche Index-Werte wie z. B. (*Calinski–Harabasz-index*[2], *Silhouette Score*[3]) entsprechend des Clusterverfahrens ermittelt.

Interpretation des Ergebnisses: Schließlich wird das Ergebnis der Analyse interpretiert. Dazu werden die Werte von Indexen für diverse analysierte Datensätze begutachtet, um das Verhältnis des Clusterverfahrens zu erklären.

3.2 Datensätze

Für ebendiese Bachelorarbeit wurden für die Analyse der Clustering-Verfahren drei unterschiedliche mehrdimensionale Datensätze mit verschiedenen Dimensionen benutzt. Diese Datensätze beinhalten unterschiedliche Mengen an Datenpunkte und wurden bereits für den Bereich Clusteranalyse vorbereitet.

BMI-Datensatz (Body Mass Index)

Dieser Datensatz[4] wurde entwickelt, um den Body Mass Index (BMI) von Personen basierend auf Geschlecht, Größe und Gewicht zu schätzen. Die Herausforderung entsteht, weil der Datensatz wenige Datenpunkte enthält und stark unausgeglichen ist. Die Daten enthalten 500 Proben sowie die folgenden Spalten:

Geschlecht : Männlich / Weiblich
Höhe : Anzahl (cm)
Gewicht : Anzahl (kg)

[2] Scikit Learn Developers. *Ccalinski Harabasz Score*. 2007. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.calinski_harabasz_score.html.
[3] Scikit Learn Developers. *Silhouette Score*. 2007. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score.
[4] https://www.kaggle.com/datasets/yasserh/bmidataset

Index : (0-Extrem schwach / 1-Schwach / 2-Normal / 3-Übergewicht / 4-Fettleibigkeit / 5-Extreme Fettleibigkeit)

Die Daten sind als *csv-Datei* verfügbar und sehen wie folgt (siehe Tabelle [3.1]) aus.

Geschlecht	Höhe in cm	Gewicht in kg	BMI-Index
Männlich	174	96	4
Männlich	189	87	2
...	...	...	...

Tabelle 3.1: Überblick über Beispieldatensatz von BMI

BlogFeedback Data Set

Diese Daten stammen aus Blogbeiträgen[5] und wurde in [22] veröffentlicht, die gecrawlt und verarbeitet wurden. Daten bezogen besteht die Vorhersageaufgabe darin, die Anzahl der Kommentare in den nächsten 24 Stunden vorherzusagen. Um diese Situation zu simulieren, werden eine Basiszeit (in der Vergangenheit) und die Blogbeiträge mit den meisten Veröffentlichungen 72 Stunden vor dem ausgewählten Basisdatum bzw. Basiszeit ausgewählt. Das Ziel ist die Anzahl der Kommentare, die ein Blogbeitrag in den nächsten 24 Stunden erhalten wird (relativ zur Basiszeit). Die Datenpunkte in diesem Datensatz enthalten Features, die aus Blogbeiträgen (*zip-Datei*) extrahiert wurden. (siehe Tabelle [3.2])

Anzahl von Datenpunkten : 60021
Anzahl von Merkmalen : 281
Type von Werten : Integer, Real

[5] https://archive.ics.uci.edu/ml/datasets/BlogFeedback

Durchschnitt	Standardabweichung	...	Anzahl der Kommentare in 24 Std.
40,30467	53,845657	...	1
40,30467	53,845657	...	0
...	...	...	...

Tabelle 3.2: Überblick über Beispieldatensatz von BlogFeedback Data Set

MicroMass Data Set

Dieser Datensatz[6] zur Erforschung von Methoden des maschinellen Lernens zur Identifizierung von Mikroorganismen aus Massenspektrometriedaten besteht aus zwei Teilen: Eine Referenzgruppe von 20g positiver und negativer Bakterienarten, die 9 Gattungen abdecken, von denen bekannt ist, dass sie schwer zwischen mehreren Arten durch Massenspektrometrie (MALDI-TOF) zu unterscheiden sind. Jede Spezies wird durch 11 bis 60 Massenspektren von 7 bis 20 Bakterienstämmen repräsentiert, für insgesamt 571 Spektren von 213 Stämmen. Die Spektren wurden nach einem in der klinischen Routine verwendeten kulturbasierten Standardarbeitsablauf erhalten, bei dem Mikroorganismen zunächst 24 bis 48 Stunden lang auf Agarplatten gezüchtet, dann ein Teil der Kolonien gepickt, auf MALDI-Objektträger getüpfelt und Massenspektren erhalten wurden.
Basierend auf diesem Referenzpanel wurde ein dedizierter In-vitro-Modellmischungsdatensatz erstellt. Dazu wurden 10 Artenpaare mit unterschiedlicher taxonomischer Nähe betrachtet.

Anzahl von Datenpunkten : 931
Anzahl von Merkmalen : 1300
Type von Werten : Real

3.3 Beschreibung der Verfahren

Im Rahmen dieser Arbeit wird die Clusteranalyse auf vier andersartigen Clustering-Verfahren (*Agglomeratives Clustering*, *Expectation Maximization Clustering*, *Spektrales*

[6] https://archive.ics.uci.edu/ml/datasets/MicroMass

Clustering, *CURE Clustering*) basiert. Für alle diese Verfahren wird die schon implementierte Version verwendet und entsprechend dem Datensatz hierzu einige Programmbibliotheken für die Auswertung des Ergebnisses gebraucht. Die Visualisierung der Clusterergebnisse erfolgt durch drei Verfahren zu Dimensionsreduktion.

- Die Implementierung des agglomerativen Clustering wird vom Artikel auf stackabuse.com[7] bereitgestellt. Die Methode *clusteranalyse(nClusters, dataset)* aus Listing [3.1] stellt den Hauptteil der Implementierung vor.

Listing 3.1: Agglomerative Clustering

```
1  from time import time
2  import matplotlib.pyplot as plt
3  import pandas as pd
4  import numpy as np
5  import scipy.cluster.hierarchy as shc
6  from sklearn.cluster import AgglomerativeClustering
7  from sklearn.metrics import silhouette_score
8  from sklearn.metrics import davies_bouldin_score
9  from dunnSklearn import dunn
10 from sklearn.metrics.pairwise import euclidean_distances
11 from sklearn import metrics
12
13 file_1 = 'dataset/BMI-Datensatz.csv'  # Dataset von BMI (Body Mass Index)
14 file_2 = 'dataset/blogData_train.csv'   # BlogFeedback Data Set
15 file_3 = 'dataset/pure_spectra_matrix_test.csv'
16
17 def clusteranalyse(n_clusters, dataset):
18     # Erstellung Vorhersagemodell
19     t0 = time()
20     model = AgglomerativeClustering(n_clusters=n_clusters, linkage='ward')
           .fit(dataset)
21     fit_time = time() - t0
22
23     label = model.labels_
24
25     return dataset, label, fit_time
26
27 # Darstellung des Dendrogramms
28 def dendrogramDraw(dataset):
29     plt.figure(figsize=(10, 7))
```

[7] https://stackabuse.com/hierarchical-clustering-with-python-and-scikit-learn/

```
30      plt.title("Drogen Dendograms")
31      plt.xlabel("Beobachtungen")
32      plt.ylabel("Distanz")
33      dend = shc.dendrogram(shc.linkage(dataset, method='ward'))
34      plt.savefig("dendogramm.svg", dpi=320, format="svg", bbox_inches='
            tight')
```

Der Algorithmus verwendet die Methode *dendrogramDraw(dataset)*, um ein Dendrogramm zu zeichnen und zu visualisieren, wie der Datensatz strukturiert ist. Es werden verschiedene Arten von agglomerativem Clustering (*ward*, *average*, *complete*, *single*) durchgeführt, indem der Wert von *„Linkage“*[8] geändert wird. Außerdem werden die Bibliotheken *Scikit-Learn* und *dunnSklearn* verwendet, um die verschiedenen Metriken (*Silhouette Index*, *Davies Bouldin Index*, *Calinski Harabasz Index* und *Dunn Index*) zu berechnen.
Nachdem die Anzahl der Cluster gemäß dem Datensatz bestimmt wurden, wird die Klasse *AgglomerativeClustering()* verwendet, um den Clustering-Algorithmus auszuführen. Dazu wird der Standardwert des *affinity*-Parameters benutzt (*affinity 'euclidean'*: Abstand zwischen den Datenpunkten). Als nächstes wird das Modell mit der *fit()*-Methode erstellt und schließlich das Label (Variable: *„label“*) bestimmt, das das Clustering-Ergebnis repräsentiert.
Die Ergebnisse der Clusteranalyse werden in einem 2D-Plot dargestellt, wobei die Achsen die Ergebnisse des Algorithmus zur Dimensionsreduktion sind.

- EM ist ein iterativer Algorithmus, der auf dem Gaußschen Mischungsmodell (*GMM*) basiert. Das nachstehende Listing [3.2] illustriert einen Teil der für die Arbeit benutzten Implementierung, die von einer Veröffentlichung in *Towards Data Science*[9] bereitgestellt wird.

Listing 3.2: Expectation Maximization Clustering (EM)

```
1  import numpy as np
2  import pandas as pd
3  from time import time
4  from sklearn.mixture import GaussianMixture
5  from matplotlib import pyplot as plt
6  from sklearn.metrics import silhouette_score
```

[8] Das Linkage kriterium bestimmt den zwischen Beobachtungssätzen verwendeten Abstand.
[9] https://towardsdatascience.com/implement-expectation-maximization-em-algorithm-in-python-from-scratch-f1278d1b9137

```
7  from sklearn import metrics
8
9  file_1 = 'dataset/BMI-Datensatz.csv'  # Dataset von BMI (Body Mass Index)
10 file_2 = 'dataset/blogData_train.csv'   # BlogFeedback Data Set
11 file_3 = 'dataset/pure_spectra_matrix_test.csv'
12
13 def GMM_sklearn(x, n_components=3):
14   # Erstellung Modell und Messung von Ablaufzeit der Analyse
15     t0 = time()
16     model = GaussianMixture(n_components=n_components,
17                             covariance_type='full',
18                             tol=0.001, reg_covar=1e-1,
19                             max_iter=1000,
20                             ).fit(x)
21     fit_time = time() - t0
22     print("\nscikit learn:\n\tphi: %s\n\tmu_0: %s\n\tmu_1: %s\n\tsigma_0:
           %s\n\tsigma_1: %s"
23           % (model.weights_[1], model.means_[0, :], model.means_[1, :],
                 model.covariances_[0, :], model.covariances_[1, :]))
24
25     return x, model.predict(x), fit_time, model.predict_proba(x)[:, 1]
```

Die Metriken *Silhouette Index* und *Calinski Harabasz Index* werden mit Hilfe der Bibliothek *Scikit-Learn* für Auswertung des Ergebnisses berechnet. Bei der Erstellung von Labels (Clusterergebnisse) wird die Kovarianz auf *„full“* gesetzt, sodass jede Komponente ihre eigene allgemeine Kovarianz-Matrix besitzt. Überdies wird entsprechend des Datensatzes der Wert von Regular Kovarianz (*reg_ covar*) geändert, um ein besseres Ergebnis zu erzielen. *reg_ covar* kann als nicht negative Regularisierung definiert werden, die der Kovarianzdiagonale hinzugefügt wird. Dies ermöglich, sicherzustellen, dass die Kovarianzmatrix vollständig positiv bleibt.
Zunächst wird ein Modell mit der Methode *fit()* der Klasse *GaussianMixture* entworfen. *fit()* iteriert *max_ iter* Mal zwischen dem E-Schritt und dem M-Schritt in jedem Versuch, bis die Änderung kleiner als der Parameter *tol* ist. Danach wird die *predict()*-Methode verwendet, um das Model zu erstellen.

- Die folgende Methode *spectralClustering()* von Listing [3.3] führt den Kernteil der Implementierung[10] von Spectral-Clustering auf und mit den vier Indexen (*Silhouette*

[10] https://de.acervolima.com/ml-spektrale-clusterbildung/

Index, *Davies Bouldin Index*, *Dunn Index* und *Calinski Harabasz Index*) wird dieser Algorithmus ausgewertet.

Listing 3.3: Spectral Clustering

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3  import numpy as np
4  from time import time
5  from sklearn.cluster import SpectralClustering
6  from sklearn.metrics import silhouette_score
7  from sklearn.metrics import davies_bouldin_score
8  from dunnSklearn import dunn
9  from sklearn.metrics.pairwise import euclidean_distances
10 from sklearn import metrics
11
12 file_1 = 'dataset/BMI-Datensatz.csv'  # Dataset von BMI (Body Mass Index)
13 file_2 = 'dataset/blogData_train.csv'   # BlogFeedback Data Set
14 file_3 = 'dataset/pure_spectra_matrix_test.csv'
15
16 def spectralClustering(n_clusters, dataset):
17   # Erstellung Modell und Messung von Ablaufzeit der Analyse
18     t0 = time()
19     labels = SpectralClustering(n_clusters, affinity='rbf').fit_predict(
           dataset)
20     fit_time = time() - t0
21
22     return dataset, labels, fit_time
```

Dieses Clustering-Verfahren wird mit zwei Werten des Parameters *Affinity* (*rbf* / *nearest_ neighbors*) durchgeführt. Dieser Parameter definiert, wie die Affinitätsmatrix aufgebaut ist. Bei *Affinity* „*rbf*“ wird die Affinitätsmatrix unter Verwendung des RBF-Kernels (Radial Basis Function) konstruiert, wenn *Affinity* „*nearest_ neighbors*“, dann wird die Affinitätsmatrix durch Berechnung des nächsten Nachbargraphen konstruiert.
Die Variable „*labels*“ (*ndarray von Cluster Labebs*) repräsentiert das Ergebnis dieses Clustering-Verfahrens, das mit der Methode *fit_ predict()* der Klasse *SpectralClustering* erstellt wird.

- Die Implementierung von CURE Clustering, das für die Analyse benutzt wird, wird in GitLab[11] bereitgestellt. Listing [3.4] legt den Hauptteil der Implementierung dar und be-

[11] https://github.com/annoviko/pyclustering/blob/master/pyclustering/cluster/examples/

leuchtet wie die Datenpunkte mittels der Programmbibliothek *pyclustering.cluster* klassifiziert werden.

Listing 3.4: CURE Clustering

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from time import time
from pyclustering.cluster.cure import cure
from pyclustering.cluster import cluster_visualizer
from pyclustering.cluster.silhouette import silhouette
from pyclustering.utils import timedcall
from pyclustering import cluster

file_1 = 'dataset/BMI-Datensatz.csv'  # Dataset von BMI (Body Mass Index)
file_2 = 'dataset/blogData_train.csv'   # BlogFeedback Data Set
file_3 = 'dataset/pure_spectra_matrix_test.csv'

def template_clustering(number_clusters, file, number_represent_points=1,
    compression=0.5, draw=True, ccore_flag=True):
    sample = file
  # Erstellung Modell und Messung von Ablaufzeit der Analyse
    t0 = time()
    cure_instance = cure(sample, number_clusters, number_represent_points,
         compression, ccore_flag)
    fit_time = time() - t0

    clusters = cure_instance.get_clusters()
    representors = cure_instance.get_representors()
    means = cure_instance.get_means()

    return sample, clusters, means, representors, fit_time,
        number_clusters
```

Der Algorithmus wird unter Verwendung der spezifischen Werte der Parameter (*number_represent_point*, *compression*, *ccore_flag*) für die Hauptanalysemethode durchgeführt. *number_represent_point* definiert die Anzahl der repräsentativen Punkte für jedes Cluster, *compression* repräsentiert den Grad der Schrumpfung der Repräsentationspunkte in Richtung des Mittelwerts des neu erstellten Clusters nach der Zusammenführung in jedem Schritt. und *ccore_flag* definiert, ob CCORE (C++ Lösung) für die Lösung verwendet wird.
Zur Ausführung des Algorithmus wird die oben genannte Methode mit dem Datensatz

aufgerufen, der vorab möglicherweise vorbereitet wird. Zunächst wird eine Instanz der Klasse *CURE*: *„cure_ instance“* (Instanz des CURE Clustering-Algorithmus) erstellt. Die Clustering-Ergebnisse und Repräsentanten werden dann von der Instanz unter Verwendung der Methoden *„get_ clusters()“* und *„get_ representors()“* bestimmt.

- Listing [3.5] charakterisiert die Implementierung von Algorithmen zur Dimensionsreduktion, die mit Hilfe der Biblitheken (*sklearn.manifold*, *sklearn.preprocessing*, *sklearn.decomposition*) realisiert werden.

Listing 3.5: Algorithmen für Reduktierung den Dimensionen

```
1  from sklearn.manifold import TSNE
2  from sklearn.preprocessing import StandardScaler
3  from sklearn.decomposition import PCA
4  from umap import UMAP
5
6  # Dimension Reduktion mit T-SNE Algorithmus. n_component : Anzahl
7  # von Komponente nach Ablauf des Algorithmus
8  def tsne_algorithm(file, n_component=2):
9      tsne_em = TSNE(n_components=n_component, perplexity=30.0, n_iter=1000,
           verbose=1).fit_transform(file)
10     return tsne_em
11
12 # Dimension Reduktion mit PCA Algorithmus. n_component : Anzahl
13 # von Komponente nach Ablauf des Algorithmus
14 def pca_algorithm(file, n_component=2):
15     # Standardisierung
16     data = StandardScaler().fit_transform(file)
17     # PCA Algorithmus
18     pca_data = PCA(n_components=n_component).fit_transform(data)
19     return pca_data
20
21 # Dimension Reduktion mit UMAP Algorithm. n_component : Anzahl
22 # von Komponente nach Ablauf des Algorithmus
23 def umap_algorithm(file, n_component=2):
24     umap_data = UMAP(n_components=n_component).fit_transform(file)
25     return umap_data
```

Zunächst beschreibt die Methode *tsne_ algorithm()*, wie tsne-Verfahren verwendet werden, um eine Dimensionsreduktion zu erreichen. Diese Methode nimmt den Datensatz als Parameter und die endgültige Anzahl an Dimensionen wird mit dem Parameter *n_ component* fixiert. Die Klasse *TSNE()* wandelt Ähnlichkeiten zwischen Datenpunkten in

gemeinsame Wahrscheinlichkeiten um und versucht, den Unterschied zwischen den Wahrscheinlichkeiten von niedrigdimensionalen Einbettungen und mehrdimensionalen Daten zu minimieren. Dann gibt die Methode *fit_ transform()* das transformierte Ergebnis zurück.
Anschließend wird das PCA-Verfahren mit der Methode *pca_ algorithm()* ausgeführt. Standardisieren Sie zunächst den Datasatz, der anschließend mit der Klasse *PCA()* aus der Bibliothek *sklearn.preprocessing* reduziert wird. Abhängig von der Form der Eingabedaten und der Anzahl der zu extrahierenden Komponenten wird eine LAPACK-Implementierung der vollständigen SVD[12] verwendet.
Drittens wird *umap_ algorithm()* verwendet, um die Dimensionalität unter Verwendung der *UMAP()*-Klasse aus der *umap* Bibliothek zu reduzieren. Diese Klasse findet niedrigdimensionale Einbettungen von Daten, die sich der zugrunde liegenden Mannigfaltigkeit approximiert.
Nach Durchführung jedes einzelnen diesen Algorithmen werden zweidimensionale Daten zurückgegeben.

3.4 Zusammenfassung

In diesem Kapitel wird erstens über den vollständigen Prozessablauf des praktischen Teils dieser Arbeit gesprochen. Zweitens wurden die für die Analyse der Algorithmen verwendeten Datensätze definiert. Drittens wurden die Implementierungen der differenzierten Clustering-Verfahren vorgestellt. In nachfolgendem Kapitel geht es darum, die Lösungen der Untersuchung auszuwerten und zu interpretieren.

[12] SVD_LAPACK berechnet die singuläre Wertzerlegung einer Matrix durch Aufrufen der Unterroutine LAPACK

4 Analyse / Auswertung der Ergebnissen

Im Fokus des vierten Kapitels stehen die Analysen des oben genannten Clustering-Verfahrens und ihre Auswertung. Dies gliedert sich in zwei Teile für jeden Algorithmus. Der erste Teil widmet sich der Bewertung und der zweite beschäftigt sich mit der Visualisierung der Analyse.

4.1 Bewertung und Visualisierung

In dieser Arbeit konzentriert sich die Analyse für die Validierung der Cluster-Verfahren nur auf interne Cluster-Validierungsindexe und wird mit drei mehrdimensionalen Datensätze durchgeführt. Um die Darstellung von Ergebnissen der Metriken in der Tabelle zu erleichtern, werden diese umbenannt: *Daten-01* (BMI Data Set), *Daten-02* (BlogFeedback Data set) und *Daten-03* (MicroMass Data set).

4.1.1 Agglomerative Clustering

Resultate

Tabelle [4.1] liefert einen Überblick über verschiedene agglomerative Clusterings und diverse Metriken, die zur Validierung des Algorithmus verwendet werden. Die vier Arten von Verfahren wurden mit unterschiedlicher Anzahl von Clustern (*1, ..., 10*) durchgeführt und gemäß dem Datensatz sind die besten Lösungen in der Tabelle zusammengefasst. Eine Gesamtübersicht von Resutaten (siehe Tabelle [4.1] und Abbildung [4.1][1]) lässt erkennen, dass „agglomerative.ward“ (*linkage ward*) für die zur Analyse ausgewählten Datensätze besser geeignet ist. Im Vergleich zu den anderen Clustern ist die Ähnlichkeit

[1] Boxplot der Metriken für jede agglomerative Clustering auf die drei Datensätze

innerhalb der Cluster (CHS und SS) für das agglomerative Clustering mit dem Ward-Parameter am besten und die Datenpunkte sind auch gut verteilt. Es ist anzugeben, je niedriger der DB-Indexwert ist, desto besser ist das Clustering, und je höher der Dunn-Indexwert ist, desto besser ist die Clusteranalyse.

Clustering-Verfahren	Datensätze	DBS	Dunn	CHS	SS	Clusteranzahl	Clusterverteilung
Agglomerative.ward	Daten 01	0.848	0.027	592.653	0.430	2	0.534 0.466
	Daten 02	0.847	0.016	736.487	0.510	5	0.124 0.043 0.042 0.684 0.107
	Daten 03	1.888	0.073	57.291	0.290	2	0.894 0.106
Agglomerative.average	Daten 01	1.012	0.043	528.803	0.350	3	0.256 0.350 0.394
	Daten 02	0.108	0.482	51.229	0.840	2	0.999 0.001
	Daten 03	0.404	0.180	13.515	0.540	6	0.985 0.002 0.007 0.002 0.002 0.002
Agglomerative.complete	Daten 01	0.785	0.027	697.152	0.470	2	0.534 0.466
	Daten 02	0.108	0.485	51,510	0.840	2	0.999 0.001
	Daten 03	1.124	0.065	39.161	0.220	7	0.861 0.011 0.011 0.009 0.047 0.016 0.045
Agglomerative.single	Daten 01	0.537	0.068	2,909	0.220	2	0.998 0.002
	Daten 02	0.108	0.485	51.510	0.840	2	0.999 0.001
	Daten 03	0.279	0.490	8.648	0.600	2	0.998 0.002

Tabelle 4.1: Metriks Resultat: Agglomevatives Clustering

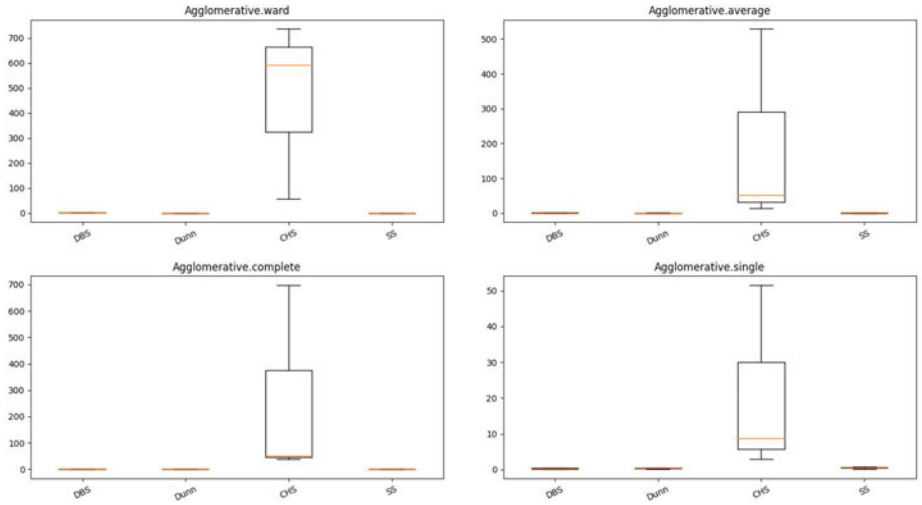


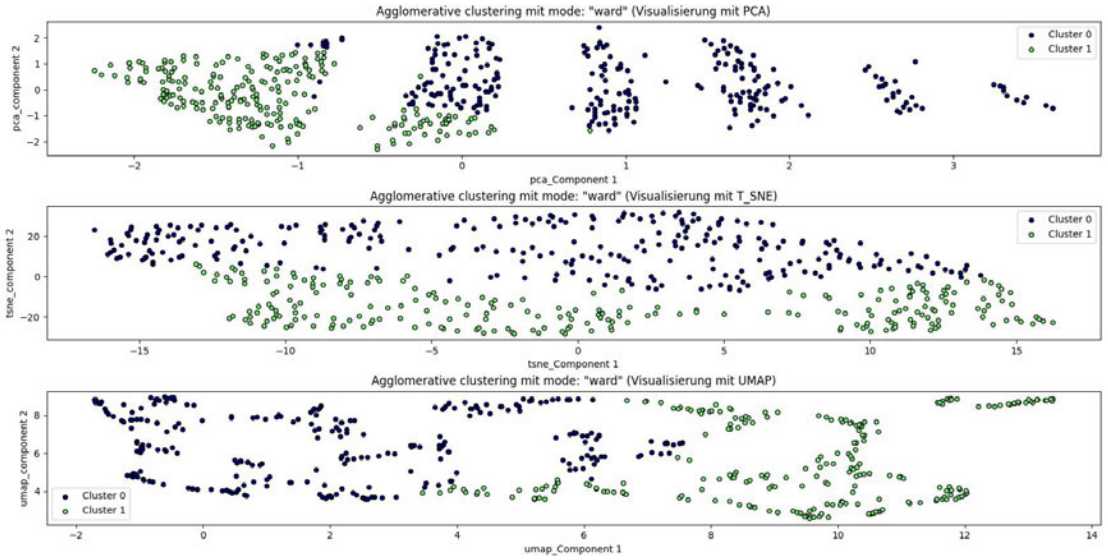
Abbildung 4.1: Darstellung verschiedener Indexe auf die drei Datensätze für jedes agglomerative Verfahren

Die Ergebnisse zeigen einerseits, dass je höher die Dimensionalität der Daten ausfällt, desto schlechter der Algorithmus wird. Am Beispiel des agglomerativen Clusterings mit ward Linkage beträgt die Ähnlichkeit von *Data-01* 0,43 und die Ähnlichkeit von *Data-03* 0,29. Andererseits hat *Data-02* laut Calinski-Harabasz-Indexwerten (736,487) und

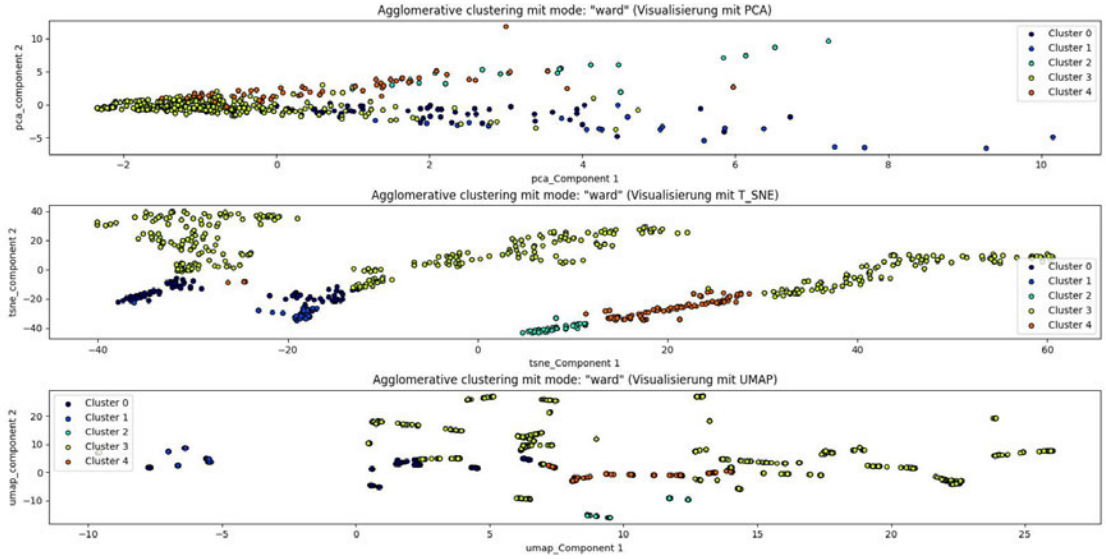
Silhouette-Indexwerten (0,510) die besten Clustering-Ergebnisse bei „agglomerative.ward“, obwohl es mehr Dimensionen als *Data-01* besitzt. Der Grund dafür ist, dass die Datenpunkte von *Data-02* strukturierter und zweckvoller als die Datenpunkte von *Data-01* und *Data-03* für diese Art von Analyse sind.

Visualisierung

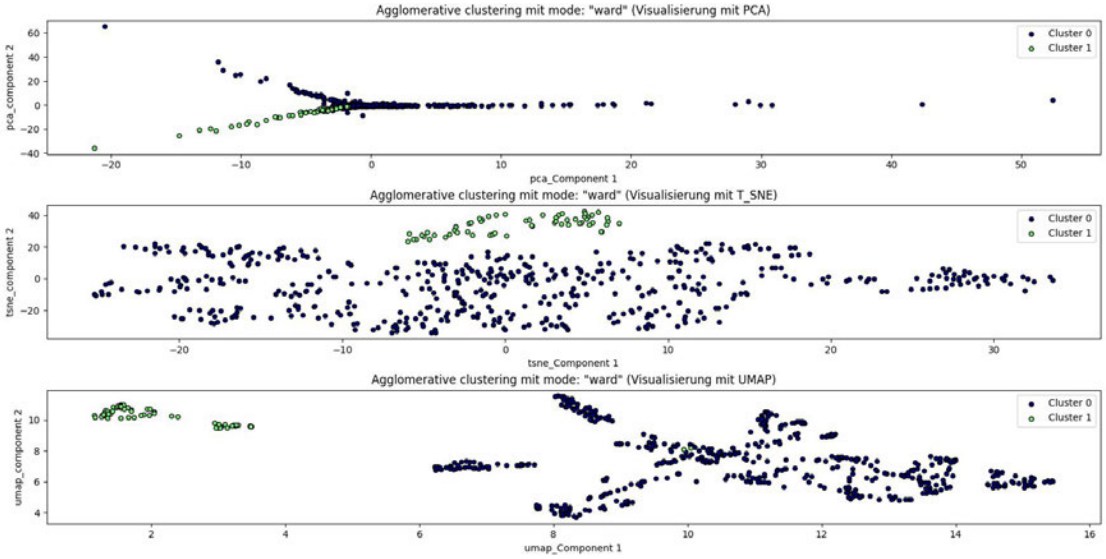
Die Abbildung [4.2] veranschaulicht die Clustering-Ergebnisse des agglomerativen Clusterings unter Verwendung von *PCA*, *T-SNE* und *UMAP*. In Abbildung [4.2] ist deutlich zu erkennen, dass das *PCA-Verfahren* nicht optimal ist, wenn Clusterergebnisse aus hochdimensionalen Datensätzen angezeigt werden.



(a) *Agglomeratives Clustering (linkage „ward“) für Daten-01*



(b) *Agglomeratives Clustering (linkage „ward“) für Daten-02*



(c) *Agglomeratives Clustering (linkage „ward“) für Daten-03*

Abbildung 4.2: Visualisierung von agglomerativen Clustering mit „ward Linkage“: Darstellung mit den Algorithmen der Dimensionsreduktion (PCA,TSNE,UMAP)

4.1.2 Expectation Maximization Clustering

Resultate

Für das Expectation-Maximization-Clustering sind die resultierenden Werte der Matrizen, die zur Validierung des Verfahrens verwendet wurden, in Tabelle [4.2] zusammengefasst. Wie bei anderen in dieser Arbeit analysierten Clustering-Methoden wurde diese Analyse mit mehreren Clusteranzahlen im Bereich von 1 bis 10 durchgeführt. Nach Durchführung einer Wertanalyse der Metriken (*CHS*, *SS*) für zehn Läufe ist ersichtlich, dass die Clusteranalyse mit zwei Clustern für die drei Datensätzen die besten Ergebnisse liefert. Das Ergebnis aus nachfolgender Tabelle [4.2] zeigt, dass die Performanzen des EM-Clustering stark abfallen, wenn die Datensätze zunehmend mehr Dimensionen enthalten.

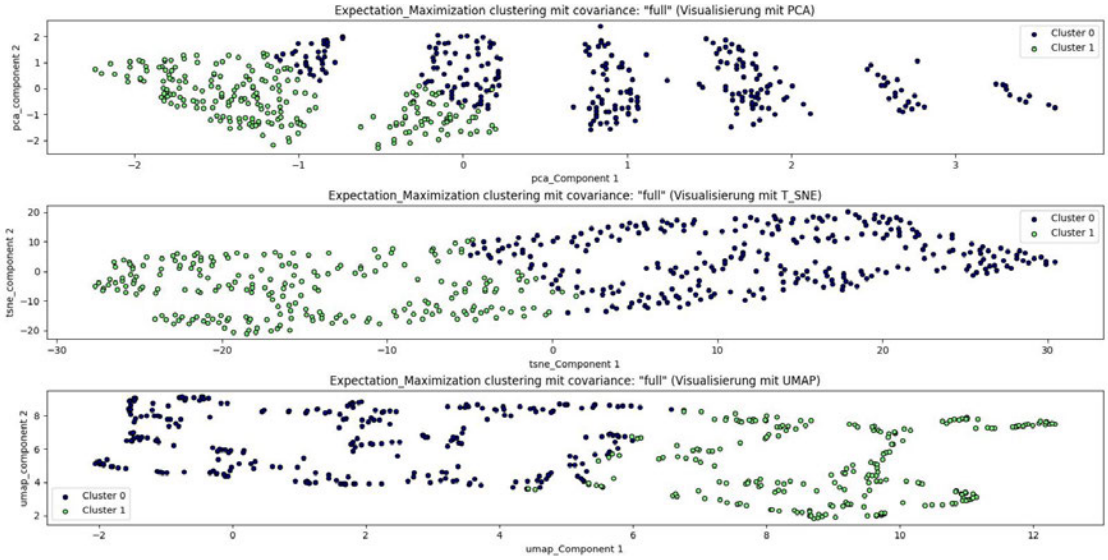
Clustering-Verfahren	Datensätze	CHS	SS	Clusteranzahl	Clusterverteilung
Expectation.Maximization	Daten 01	705.102	0.474	2	0.546 0.454
	Daten 02	163.328	0.189	2	0.530 0.470
	Daten 03	45.449	0.294	2	0.112 0.888

Tabelle 4.2: Metriks Resultat: Expectation Maximization Clustering

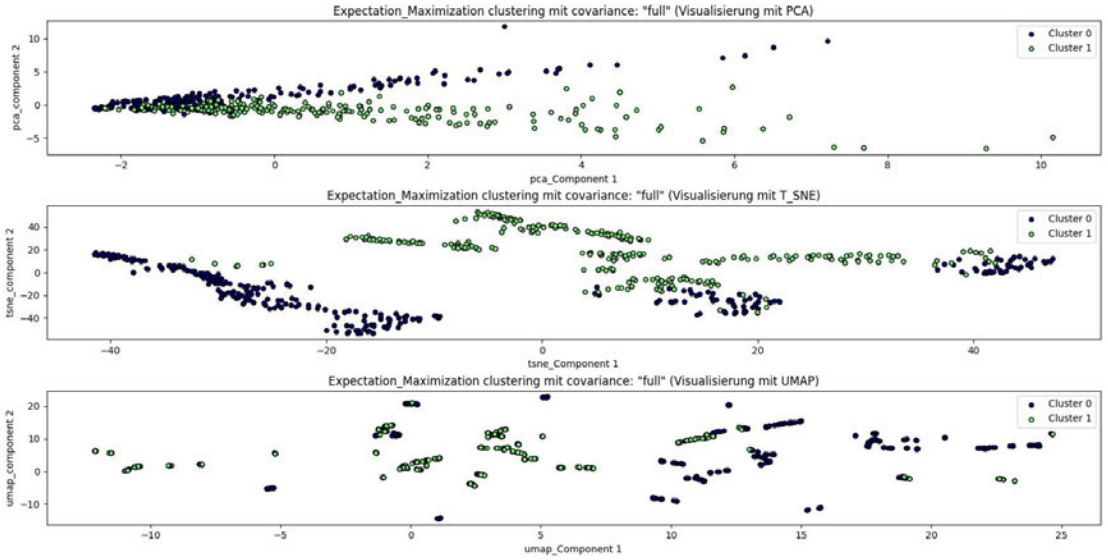
Die Clusteranalyse von *Data-01* (3-Dimensionen) ergibt einen Calinski-Harabasz-Indexwert von 705,102 und einen Silhouette-indexwert von 0,474. Außerdem sind die Datenpunkte gut verteilt. Bei der Analyse der Datensätze *Data-02* (281 Dimensionen) und *Data-03* (1300 Dimensionen) sind die Werte beider Metriken schlechter geworden.

Visualisierung

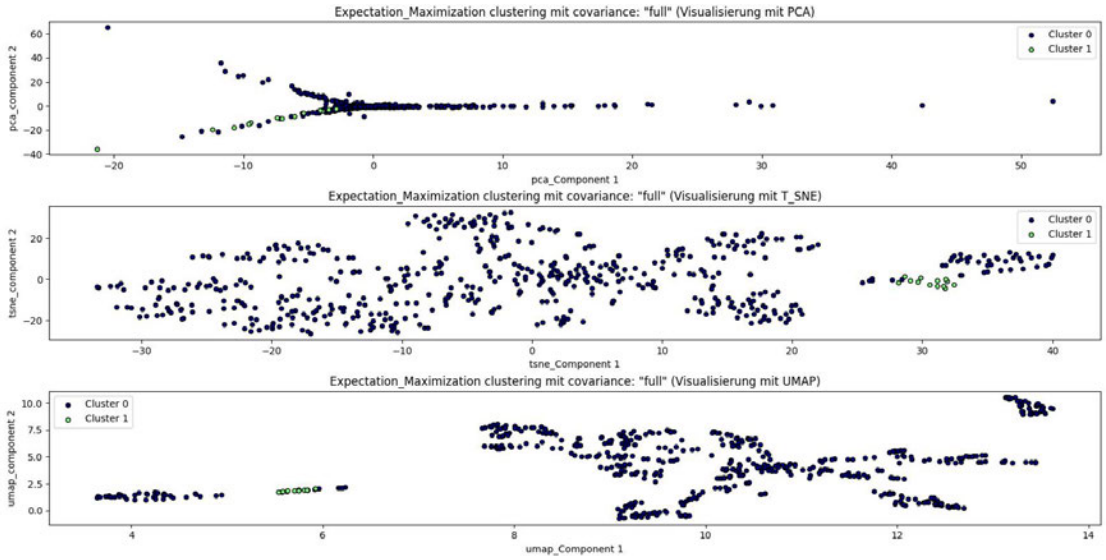
In der Grafik [4.3] geht es um die Darstellung von Expectation Maximization Clustering mit Hilfe von PCA, T-SNE und UMAP. Es illustriert auch, dass T-SNE der beste Algorithmus zur Darstellung von Clusterergebnissen ist.



(a) *Expectation Maximization Clustering: Daten-01*



(b) *Expectation Maximization Clustering: Daten-02*



(c) *Expectation Maximization Clustering: Daten-03*

Abbildung 4.3: Visualisierung von Expectation Maximization Clustering: Darstellung mit den Algorithmen von Dimensionsreduktion (PCA,TSNE,UMAP)

4.1.3 Spectral Clustering

Resultate

Tabelle [4.3] zeigt die Ergebnisse von Metriken für Spectral-Clustering. Um den Algorithmus besser analysieren zu können, wurde die Analyse mit zwei verschiedenen Werten des Parameters *„affinity“* (*rbf* und *nearest-neighbors*) durchgeführt. Die Ähnlichkeitsmaße innerhalb des Clusters (CHS und SS) führen vor Augen, dass Spectral.nn(*affinity nearest-neighbors*) im Vergleich zum Spectral.rbf(*affinity rbf*) besser geeignet ist. Beispielsweise hat Spectral.nn im selben Datensatz (Data-01) einen Calinski-Harabasz-Indexwerten von 711.530 und Spectral.rbf einen Calinski-Harabasz-Indexwerten von 2.909. Übrigens hat die Verwendung von Spectral.nn zur Analyse des Datensatzes (data-03) einen Silhouette-Indexwert von -0,101: Dieser negativen Wert von Silhouette-Score zeigt an, dass ein oder mehrere Datenpunkte dem falschen Cluster zugewiesen wurden, da ein anderes Cluster ähnlicher ist.

Clustering-Verfahren	Datensätze	DBS	Dunn	CHS	SS	Clusteranzahl	Clusterverteilung
Spectral.rbf	Daten 01	0.537	0.068	2.909	0.217	2	0.998 0.002
	Daten 02	1.152	0.027	0.436	0.053	2	0.999 0.001
	Daten 03	9.632	0.019	0.888	0.101	2	0.070 0.930
Spectral.nn	Daten 01	0,782	0.019	711.530	0.476	2	0.458 0.542
	Daten 02	0.859	0.016	500.403	0.591	2	0.110 0.890
	Daten 03	1.820	0.070	58.804	0.311	2	0.904 0.096

Tabelle 4.3: Metriks Resultat: Spectral Clustering

Abbildung [4.4][2] weißt darauf hin, dass „Spectral.nn“ (*affinity nearest-neighbors*) für die Analyse zweckvoller ist.

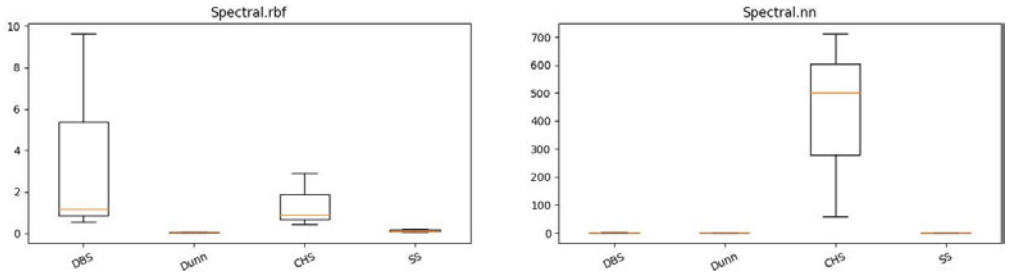
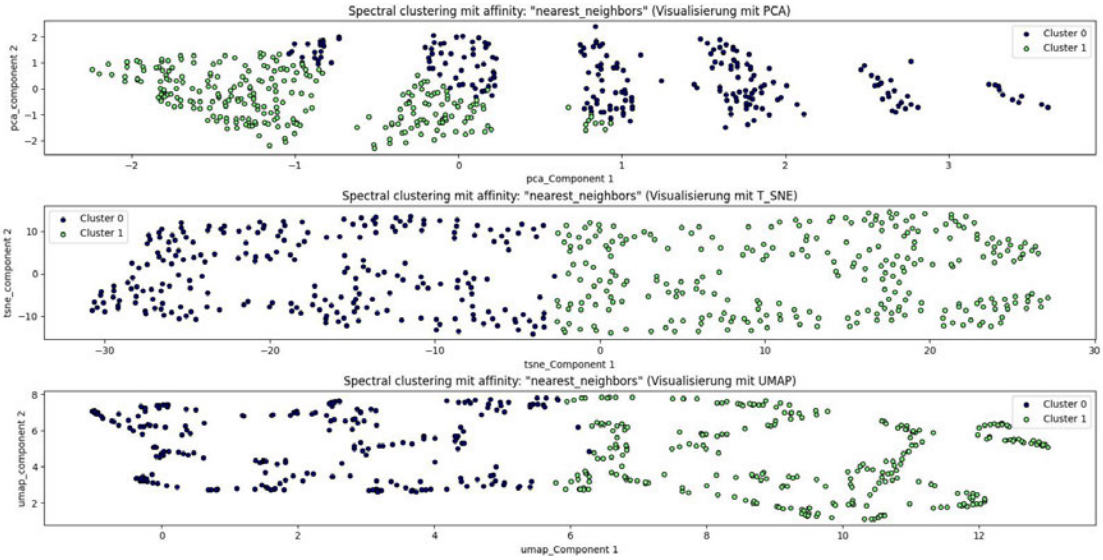


Abbildung 4.4: Darstellung verschiedener Indexe auf die drei Datensätze für jedes spektrale Verfahren

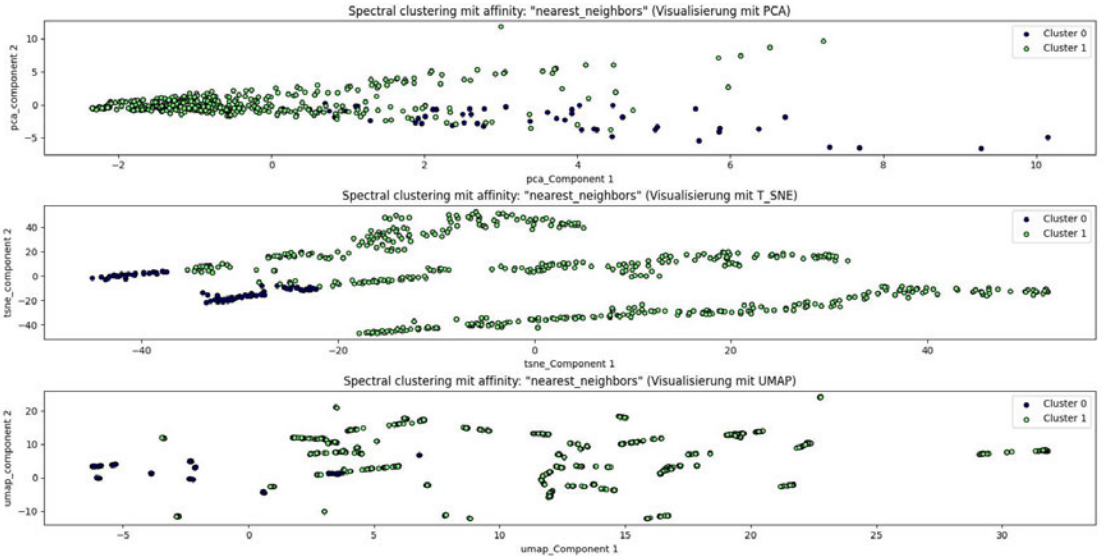
Visualisierung

Abbildung [4.5] zeigt die Visualisierung von spektralem Clustering unter Verwendung von nearest-neighbors affinity. Bei jedem Diagramm handelt es sich um eine Darstellung der analytischen Clustering-Ergebnisse für einen bestimmten Datensatz.

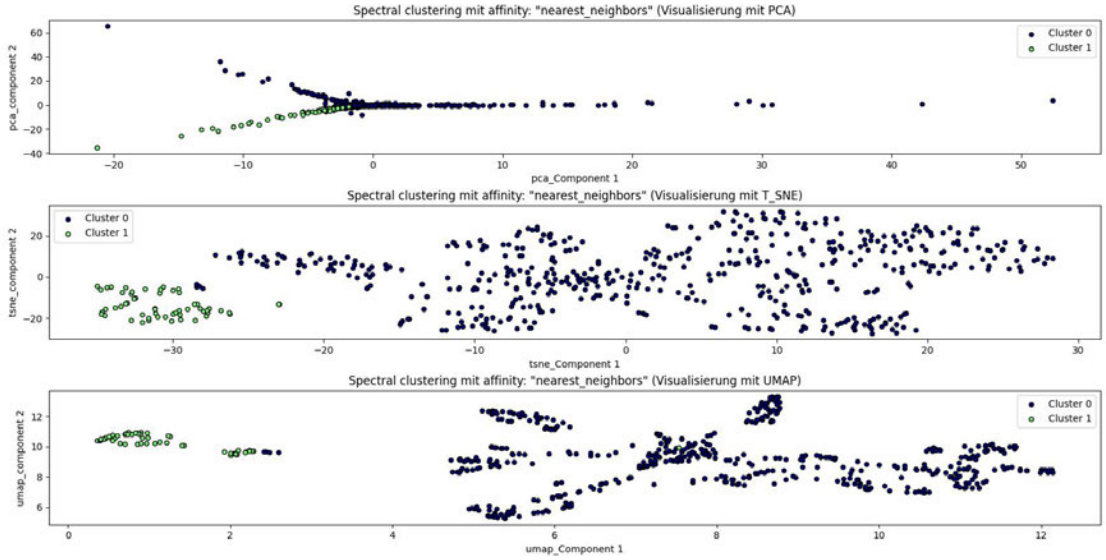
[2] Boxplot der Metriken für Spectral-Clustering bei *affinity nearest-neighbors* und *affinity rbf* auf die drei Datensätze



(a) *Spectral-Clustering (affinity nearest-neighbors): Daten-01*



(b) *Spectral-Clustering (affinity nearest-neighbors): Daten-02*



(c) *Spectral-Clustering (affinity nearest-neighbors): Daten-03*

Abbildung 4.5: Spectral-Clustering mit „affinity nearest-neighbors“: Darstellung mit den Algorithmen zur Dimensionsreduktion (PCA,TSNE,UMAP)

4.1.4 CURE Clustering

Für die zuvor in dieser Arbeit analysierten Clustering-Methoden wird gezeigt, dass einige Metriken wie (DBS, CHS, SS, ...) für die Validierung der Clusterergebnisse wichtig sind. Bei Verwendung von CURE-Clustering sind einige dieser Metriken nicht geeignet, um den Algorithmus zu validieren.

Resultate

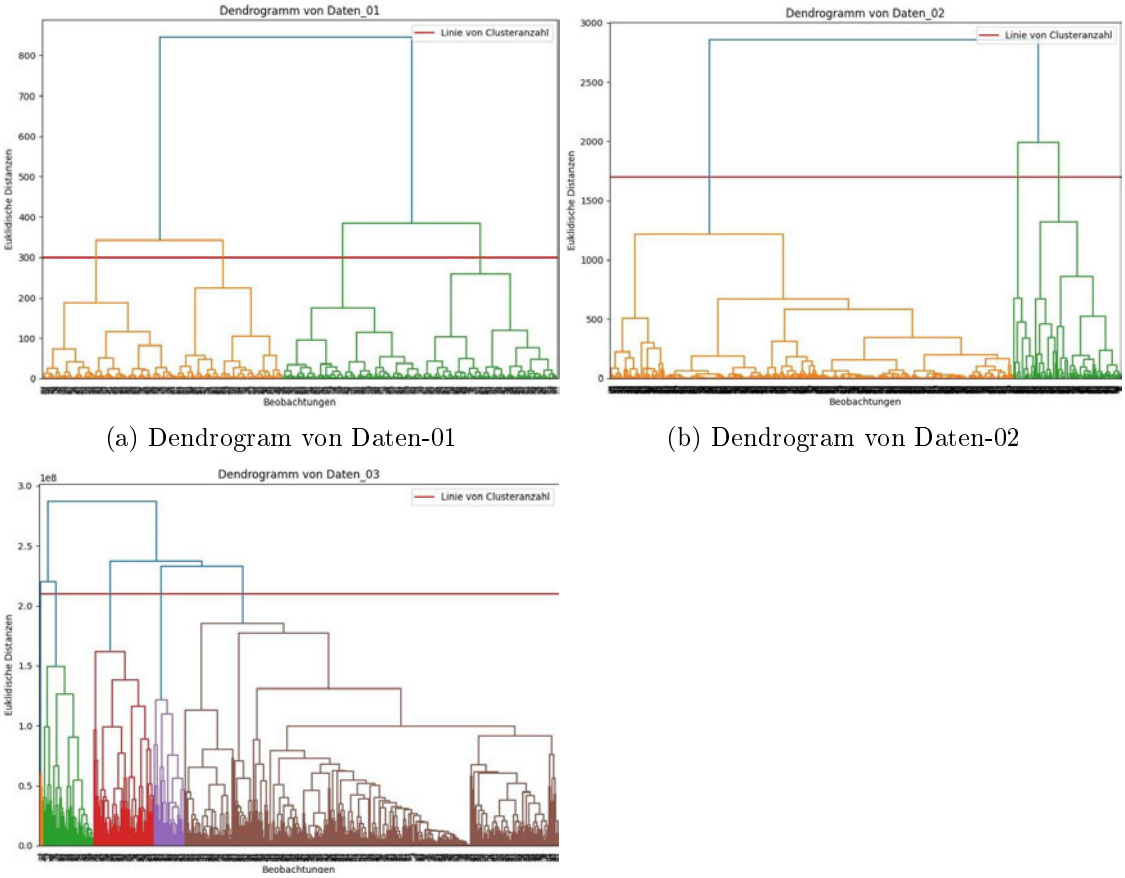
Clustering-Verfahren	Datensätze	SS	Clusteranzahl	Clusterverteilung
CURE	Daten 01	nan	4	0.242 0.232 0.210 0.316
	Daten 02	nan	3	0.998 0.001 0.001
	Daten 03	nan	5	0.979 0.014 0.003 0.002 0.002

Tabelle 4.4: Metriks Resultat: CURE Clustering

Die Tabelle [4.4] enthält Informationen zu den Ergebnissen der CURE-Clusteranalyse auf die drei Datensätze. Die Werte von Silhouette-Score(SS) werden in der Tabelle als

„nan“ bezeichnet; dies weist darauf hin, dass diese Metrik im Vergleich zu anderen Clustering-Verfahren nicht anwendbar ist. Einerseits kann der optimale Wert der Clusteranzahl durch die Berechnung einer bestimmten Validierungsmetrik (*RMSSTD* oder *RSS*) ermittelt werden. Andererseits, da CURE Clustering ein Typischer Algorithmus von hierarchischem Clustering ist (Siehe Abschnitt [2.2]), dient die Darstellung des Datensatzes im Dendrogramm auch dazu, die entsprechende Anzahl an Clustern zu bestimmen.

Die untenstehende Abbildung [4.6] veranschaulicht das Dendrogramm der Datensätze *Data-01* (siehe [4.6a]), *Data-02* (siehe [4.6b]) und *Data-03* (siehe [4.6c]). Die Dendrogramme wurden unter Verwendung von *„ward“*-Methode dargestellt.



(a) Dendrogram von Daten-01

(b) Dendrogram von Daten-02

(c) Dendrogram von Daten-03

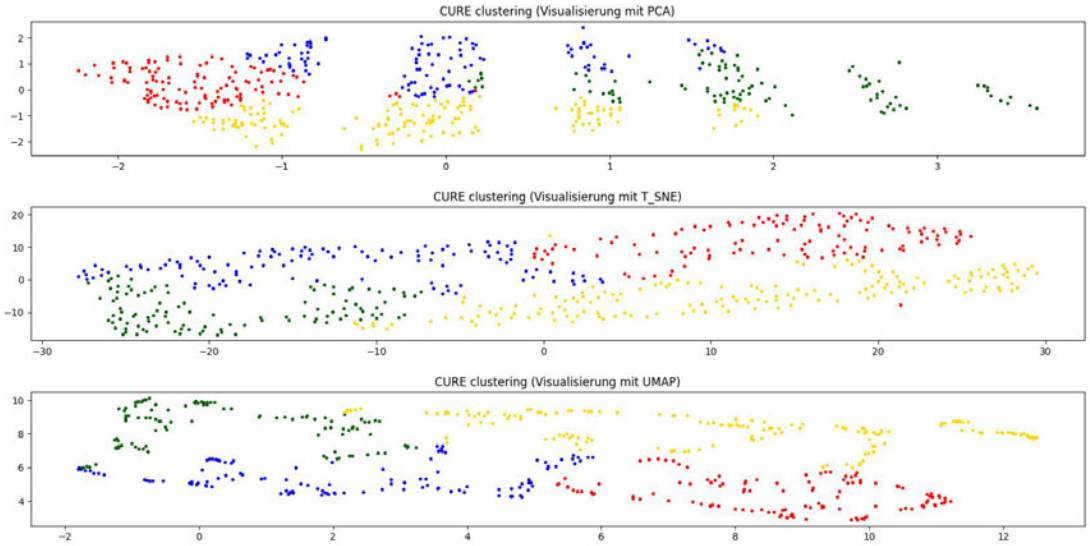
Abbildung 4.6: Dendrogramme von den drei Datensätzen mit *„ward“* Methode

Die ***„ward“*-Methode** ist eigentlich eine Methode, die versucht, die Varianz innerhalb jedes Clusters zu minimieren. Die X-Achse besteht aus Beobachtungen (sogenannte Da-

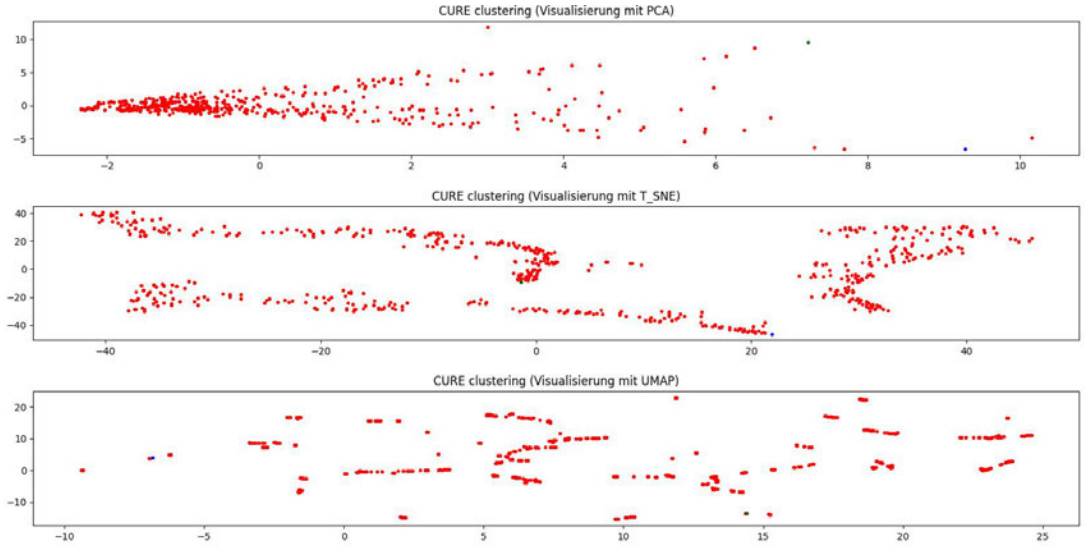
tenpunkte) und die Y-Achse aus euklidischen Abständen zwischen Clustern. Die optimale Anzahl von Clustern wird aus dem Diagramm bestimmt. Dabei wird nach der maximalen Entfernung, die vertikal erreicht werden kann, ohne die horizontale Linie zu überschreiten, gesucht. Dann wird an diesem Punkt eine horizontale Linie gezogen und die Anzahl der im Diagramm durchgestrichenen vertikalen Linien gezählt, um die optimale Anzahl von Clustern zu bestimmen. Daher gibt es vier Cluster, die auf die Analyse von *Daten-01* anwendbar sind, und drei Cluster, die auf *Daten-02* und fünf auf *Daten-03* anwendbar sind (siehe Tabelle [4.4]). Die Informationen zur Verteilung der Cluster in der Tabelle zeigen deutlich, dass je höher die Dimensionalität des Datensatzes ist, desto schlechter das CURE-Verfahren ist.

Visualisierung

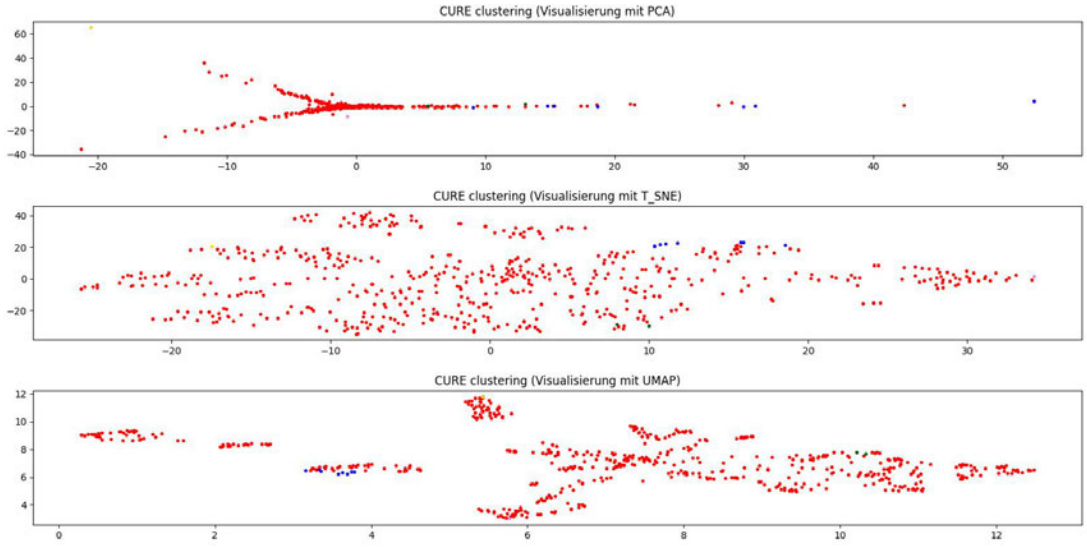
Das Image [4.7] beschreibt die Visualisierung von Clustering-Ergebnissen für CURE-Clustering unter Verwendung verschiedener Methoden. Der Abbildung ist zu entnehmen, dass die Dimensionsreduktionsalgorithmen (*T-SNE* und *UMAP*) repräsentativer für die Clustering-Ergebnisse sind als *PCA*.



(a) *CURE-Clustering: Daten-01*



(b) *CURE-Clustering: Daten-02*



(c) *CURE-Clustering: Daten-03*

Abbildung 4.7: Visualisierung von CURE-Clustering: Darstellung mit den Algorithmen von Dimensionsreduktion (PCA,TSNE,UMAP)

4.2 Zusammenfassung

Im Rahmen dieses Kapitels werden die Clusterergebnisse für jedes in dieser Arbeit analysierte Clustering-Verfahren diskutiert. Darüber hinaus wird auch die Visualisierung dieser Ergebnisse vorgestellt und diskutiert. Das nächste Kapitel fasst die gesamte Arbeit zusammen. Dabei werden nicht nur die wichtigsten bereits im Hauptteil identifizierten Ergebnisse präsentiert, sondern auch offen gebliebene Fragen und Perspektiven für die zukünftige Forschung aufgezeigt.

5 Zusammenfassung und Ausblick

Die Zielsetzung der vorliegenden Arbeit bestand darin ein lernförderndes System zu realisieren, das nach der Analyse verschiedener Clustering-Verfahren auf unterschiedlichen Datensätzen genau modellieren kann, wie gut bzw. schlecht diese Algorithmen sind, wenn die Datensätze zunehmend mehr Dimensionen enthalten. Außerdem ging es auch darum, die Visualisierung mit Hilfe von Algorithmen zur Dimensionsreduktion darszutellen.
Dieses fünfte Kapitel fasst die Ergebnisse dieser Arbeit zusammen. Außerdem wird ein Ausblick gegeben, wie das Projekt verbessert oder erweitert werden kann.

5.1 Zusammenfassung

Datensätze besitzen komplexe Eigenschaften, die durch mehrere Aspekte dargestellt werden. Einer besteht darin, eine große Datenmenge in Form einer großen Anzahl von Dateninstanzen zu erstellen, die sich aus verschiedenen Dimensionen zusammensetzen. Neben der rechnerischen Komplexität der Anzahl von Datenpunkten und Dimensionen gibt es einen weiteren Aspekt bei der hochdimensionalen Analyse.
Kapitel [2] stellt die Herausforderungen vor, die Clustering-Verfahren bei der Analyse komplexer Datensätze berücksichtigen sollten. Darüber hinaus stellt dieses Kapitel einige Metriken zur Validierung von Clustering-Ergebnissen vor. Die Anwendungsfall der in dieser Arbeit vorgeschlagenen Clustering-Methode konzentriert sich auf die Analyse von drei hochdimensionalen Datensätzen. Kapitel [3] stellt die Eigenschaften der Datensätze *BMI-Datensatz*, *BlogFeedback-Datensatz* und *Mikro-Mass datensatz* vor und beschreibt die vier Clustering-Verfahren, die für die Analyse verwendet wurden.
Eine eingehende Analyse hat gezeigt, dass einige Clustering-Methoden nicht nur für zweidimensionale Datensätze, sondern auch für hochdimensionale Datensätze geeignet sind. Kapitel [4] zeigt das Ergebnis nach der Ausführung von *agglomerativen Clustering*,

EM-Clustering, *Spectral-Clustering*, *CURE-Clustering*. Die präsentierten Clusterergebnisse rechtfertigen die Behauptung, dass die Dimensionalität die Leistung des Clustering-Verfahrens beeinflusst. Bei der Durchführung der Analyse müssen einige Parameter des Algorithmus entsprechend dem Datensatz angepasst werden, um optimale Lösungen zu erzielen.
Der Prozess der Vorverarbeitung der bereitgestellten Daten zeigt, dass die Auswahl geeigneter Datensätze für die Clusteranalyse umfangreich ist. Die vorgestellten Experimente zeigen, dass Clustering-Verfahren, die für hochdimensionale Datensätze geeignet sind, schlechter werden, wenn die Daten mehr und mehr Dimensionen aufweisen.
Ein Blick auf die Visualisierung der Studien im 2D-Raum mit *PCA*, *T-SNE*, *UMAP* zeigt deutlich, dass *T-SNE* und *UMAP* die beiden besten Methoden zur Dimensionsreduktion für diesen Anwendungsfall sind.

5.2 Ausblick

Der in diesem Beitrag vorgestellte Ansatz zur Analyse von Clustering-Methoden für unterschiedliche Datensätze stellt eine Grundlagenarbeit dar, die als Basis für die Weiterentwicklung dienen kann. Ein weiterer möglicher Forschungsschwerpunkt ist die Implementierung eines Systems oder eines Clustering-Verfahrens, das es ermöglicht, sehr hochdimensionale Datensätze (Sowie *Big Data*) schnell und mit guten Leistungsfähigkeiten zu analysieren und zu gruppieren. Literaturrecherchen in einigen Bibliotheken wie *Pyclustering*[1] und *Scikit-Learn* zeigen, dass diese Umsetzung noch offen und möglich ist.
Neben der Ermöglichung robusterer Clustering-Verfahren stellt die Optimierung von Metriken zur Validierung von Analysen eine Entwicklungsmöglichkeit dar. Im Gegensatz zu einigen Verfahren fordern die in dieser Arbeit analysierten Clustering-Verfahren die Anzahl an Clustern, um den Datensatz zu analysieren. Dies ist ein wichtiger Punkt der Clusteranalyse, da es hochwirksame Metriken erfordert. Beim CURE-Clustering in der Bibliothek *Pyclustering* beispielsweise gibt es kaum einen Validierungsindex, der die Anzahl der Cluster präzise bestimmen kann.

[1] https://pyclustering.github.io/docs/0.8.2/html/de/da1/namespacepyclustering.html

Literatur

[1] J. MacQUEEN. “Some Methods for classification and analysis of multivariate observations”. In: *University of California and Los Angeles* (1967), S. 281–297.

[2] Bock und H.H. “linkage13”. In: *Automatische Klassificakation*. Vandenhoeck und Ruprecht, Göttingen, 1974.

[3] David L. Davies und Donald W. Bouldin. *DaviesBoulin*. IEEE Transactions on Pattern Analysis und Machine Intelligence, 1979. ISBN: 10.1109/TPAMI.1979.4766909. URL: https://ieeexplore.ieee.org/document/4766909.

[4] Everitt und B.S. “linkage14”. In: *Cluster Analysis*. 3. Aufl. Arnold und London, 1993.

[5] Martin Esther u. a. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: *Fraunhofer-Gesellschaft* (1996), S. 226–231. URL: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf?.

[6] Martin Esther u. a. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: *Fraunhofer-Gesellschaft* (1996), S. 226–231. URL: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf?.

[7] H. Kaufmann und H. Pape. “agglomerative”. In: *Clusteranalyse*. Manning Publications Co, 1996, S. 457–458. ISBN: 9781617291807.

[8] Tood K Moon. “em”. In: *The expectation-maximization algorithm*. IEEE Signal Processing Magazine, 1996, S. 47–60. ISBN: 10535888.

[9] Trevor Hastie, Robert Tibshirani und Jerome Friedman. *Hochdimension*. 2. Aufl. Springer, 2001.

[10] Dr. Richard Hoberg. “Clusteranalyse”. In: *Clusteranalyse, Klassifikation und Datentiefe*. 1. Aufl. Josef Eul Verlag GmbH, 2002. ISBN: 3899360583.

[11] Christopher M. Bishop. “Pattern”. In: *Pattern Recognition and Machine Learning*. Springer New York und NY, 2006, S. 7.

[12] Scikit Learn Developers. *Ccalinski Harabasz Score*. 2007. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.calinski_harabasz_score.html.

[13] Scikit Learn Developers. *Silhouette Score*. 2007. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score.

[14] Laurens Van der Maaten und Georey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* (2008).

[15] Laurens Van der Maaten und Georey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* (2008), S. 2579–2605.

[16] A. L. Mayer. "Multidimension". In: *Strengths and weaknesses of common sustainability indices for multidimensional systems*. Environment international, 2008, S. 277–291.

[17] Don Wunsch und Rui Xu. "inbook10". In: *Clustering*. John Wiley und Sons, 2008, S. 2–24. ISBN: 9780470382783.

[18] Hans-Peter Kriegel, Peer Kröger und Arthur Zimek. "Ähnlichkeit15". In: *Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering*. Authors Info und Claims, 2009, S. 1–58. URL: https://doi.org/10.1145/1497577.1497578.

[19] Harvey J. Miller und Jiawei Han. "Ähnlichkeit16". In: *Reactive Design Patterns*. 2. Aufl. Manning Publications Co, 2009, S. 9–16. ISBN: 9780429189296. URL: https://doi.org/10.1201/9781420073980.

[20] Hervé Abdi, Williams und Lynne J. "Principal component analysis". In: *Wiley interdisciplinary* (2010), S. 37–52.

[21] R. Arora u. a. "Optimization". In: *Stochastic optimization for PCA and PLS*. IEEE, 2012, S. 861–868.

[22] Krisztian Buza. *Feedback-Vorhersage für Blogs*. 2014. URL: http://www.cs.bme.hu/~buza.

[23] Charrad u. a. "NbClust: Ein R-Paket zur Bestimmung der relevanten Anzahl von Clustern in einem Datensatz". In: *Zeitschrift für Statistische Software* (2014), S. 1–36.

[24] Alboukadel Kassambara. "Validierung". In: *Practical Guide to Cluster Analysis in R*. 1. Aufl. Data Novia, 2017.

[25] Inga Dö. u. a. "Reactive Design Patterns". In: *Fraunhofer-Gesellschaft* (2018), S. 52. URL: https://www.bigdata-ai.fraunhofer.de/content/dam/%09%09bigdata/de/documents/Publikationen/Fraunhofer_Studie_ML_201809.pdf.

[26] Leland McInnes, John Healy und James Melville. "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction". In: *Cornell University* (2018). URL: https://doi.org/10.48550/arXiv.1802.03426.

[27] Prof. Dr. rer. nat. Jörg Frochte. "MachinellesLernen". In: *Maschinelles Lernen: Grundlagen und Algorithmen in python*. 3. Aufl. Carl Hanser Verlag München, 2021. Kap. 2, S. 14. ISBN: 9783446461444.

[28] Tobias Marzell. *Clustering mit Machine Learning - Ein ausführlicher Leitfaden*. 2021. URL: https://rocketloop.de/de/blog/clustering-machine-learning-ausfuhrlicher-leitfaden/#section-three.

A Appendices

These documents are available in the CD.

- Doxygen Code documentation compressed in a zip file.
- Raw data of all tests performed(.zip)
- The code source of the implemented circuit breaker as an archive file. (.tar.gz)

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort Datum Unterschrift im Original