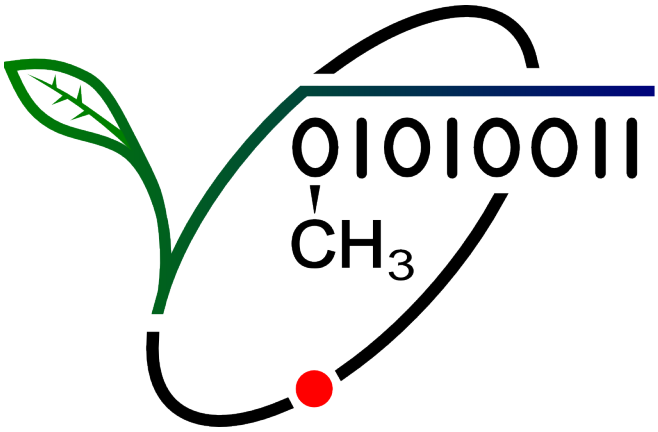
Spezialschulteil des Albert-Schweitzer-Gymnasiums Erfurt



Seminarfacharbeit Klassenstufen 11/12

Schuljahre 2022 bis 2024

Automatisierte Multiobjektclusterung mithilfe von Machine-Learning-Verfahren

Methoden der optimalen Clusterung verschiedener Datensätze mit Hilfe eines neuronalen Netzes

---

Seminarfachbetreuerin:	Frau Dr. Marion Moor
Fachbetreuer:	Herr Johannes Süpke
Außenbetreuer:	Herr Dr. Wolfgang Felber Herr Christopher Sobel
Gruppenmitglieder:	Hagen Jacob, Jan Edgar König, Robert Vetter
Ort, Datum:	Erfurt, 21.12.2023

Inhaltsverzeichnis

1 Einleitung

In einer zunehmend digitalisierten Welt spielt die Automatisierung von Prozessen eine immer wichtigere Rolle. Jeff Bezos, Gründer von Amazon, betonte bereits im Jahr 2022: „Es gibt keine Alternative zur digitalen Transformation.“[1] In diesem Kontext ist die Fähigkeit, Daten effizient zu strukturieren und zu verstehen, unerlässlich für die Entwicklung von Technologien und die Optimierung verschiedener Systeme. Aktuell wird diese Informationsanalyse oft noch händisch von Datenanalysten durchgeführt, die zahlreiche Datenpunkte in Diagrammen korrekt ihren jeweiligen Gruppierungen zuordnen müssen. Diese manuelle Zuweisung kann jedoch sehr zeitintensiv sein. Daher ist es wesentlich effizienter, Verfahren zu entwickeln und einzusetzen, die automatisiert eine sinnvolle Strukturierung und Klassifizierung dieser Informationen ermöglichen. Somit wird der Anschluss an die fortschreitende Digitalisierung gehalten.

Clustering-Algorithmen, insbesondere im Bereich des unüberwachten Lernens, sind solche Verfahren, die es ermöglichen, Datenmengen mit unterschiedlichen Eigenschaften zu analysieren und in Gruppen einzuteilen. Dabei existieren bereits jetzt schon unzählige solcher Methoden, wobei jede ihre spezifischen Vor- und Nachteile besitzt. Die Herausforderung besteht nun in der automatisierten Auswahl des am besten geeigneten Verfahrens für die jeweiligen Messdaten, um den Selektionsprozess erheblich zu beschleunigen. Dadurch könnte nicht nur der Zeitaufwand reduziert, sondern auch die Genauigkeit der Klassifikation in vielen Anwendungsfällen verbessert werden. Das Hauptziel dieser Arbeit spiegelt dieses Bestreben wider: Die Entwicklung eines neuronalen Netzes, welches selbstständig den optimalen Clustering-Algorithmus für einen gegebenen Datensatz bestimmt. Dabei wird sich speziell auf die automatisierte Selektion komplementärer Algorithmen konzentriert und nicht auf die gesamte Bandbreite der Klassifikationsverfahren.

Im Rahmen dieser Arbeit wird sich zunächst den Grundlagen des Clusterings gewidmet, um theoretisches Vorwissen zu sammeln. Dabei sind die Funktionsweise und die einzelnen Vor- und Nachteile ausgewählter Algorithmen relevant, um deren Unterschiede zu verstehen und somit das finale Ziel erreichen zu können. Anschließend wird mit der Extraktion charakteristischer Eigenschaften von synthetisch generierten Datensätzen fortgefahren. Dadurch kann eine gegebene Punktemenge effizient und detailliert beschrieben werden. Zusätzlich wird im Zuge der Entwicklung eines Algorithmus zur Detektion von Anomalien die Beseitigung von Ausreißern vorgenommen. Somit wird es ermöglicht, ein neuronales Netz mithilfe einer Vielzahl von spezifischen Charakteristiken vieler Datensätze zu trainieren.

Um dieses Vorhaben in die Realität umzusetzen, werden mehrere Methoden verwendet. Im Anschluss an ein tiefgehendes Literaturstudium über die theoretischen Konzepte hinter dem Clustering, wird mit der Implementierung relevanter Algorithmen fortgefahren und diese an unterschiedlichen Datensätzen getestet. Die dabei gesammelten praktischen Erkenntnisse werden durch eine digitale Fortbildung beim Fraunhofer-Institut in Nürnberg vertieft. Dabei ist es ebenso das Ziel, auftretende Probleme im Konsens zu besprechen. Bei dem Schaffensprozess der Arbeit wird sich auch mit wissenschaftlichen Publikationen auseinandergesetzt. Es werden Konspekte erstellt, welche beim Verständnisprozess von solchen Ausarbeitungen unterstützen.

Im Anschluss möchten wir uns für die zielführende Kooperation mit unserer Seminarfachbetreuerin Frau Dr. Moor sowie unserem Fachbetreuer Herrn Süpke bedanken, die uns mit konstruktiven formellen und fachlichen Hinweisen unterstützten. Darüber hinaus gebührt auch besonderer Dank unseren Außenbetreuern Herrn Dr. Felber sowie Herrn Sobel vom Fraunhofer-Institut in Nürnberg für ihre tiefgreifende fachliche Assistenz. Abschließend danken wir allen weiteren Personen, die uns bei der Erstellung dieser Arbeit begleiteten.

[1] Stäudtner, Jürgen: *Sprüche und Zitate zur Digitalisierung, der digitalen Transformation und dem digitalen Wandel*, `https://www.cridon.de/sprueche-zitate-digitalisierung/` [Zugriff am 25.11.2023]

2 Relevante Begriffe zu Machine-Learning-Verfahren

2.1 Klassifikation und Regression als Teilbereiche des überwachten Lernens

Neben der Regression gilt die Klassifikation als der zweite grundlegende Modellierungsansatz im überwachten Lernen, bei dem Datenpunkte vorgegebenen Kategorien zugeordnet werden.[2] Für den Menschen ist es dabei intuitiv verständlich, welche Konzepte eine Klasse repräsentieren. So wird beispielsweise eindeutig erkannt, wie ein Knochenbruch aussieht oder was ein Fahrzeug von einem Tier unterscheidet. Der Computer muss hierfür jedoch zunächst Trainingsdaten durchlaufen, damit ein Klassifikator durch die Beispiele sozusagen eine direkte Definition für jede Kategorie generieren kann und damit Muster problemlos wiedererkennt. Dieses Paradigma wird als „überwachtes Lernen“ bezeichnet, da die Trainingsdatensätze bereits die Eingabedaten und die dazugehörigen korrekten Lösungen beinhalten.[3] Eine gründliche Erfassung und Vorbereitung der Daten ist dabei essentiell, um die Klassifikation durchzuführen, was jedoch auch eine hohe Rechenleistung erfordert.
Die Regression stellt hingegen eine spezifische Kategorie des überwachten Lernens dar, in welcher das Hauptziel die Vorhersage einer abhängigen Variable auf Grundlage von einem oder mehreren Eingabeattributen ist.[4] Somit soll ein Modell erstellt werden, welches in der Lage ist, die Ausgabe in Abhängigkeit von den Eingabefunktionen vorauszusagen. Der Unterschied im Vergleich zur Klassifikation besteht darin, dass ein Regressionsalgorithmus keinen diskreten Wert als Klassenbezeichnung ausgeben, sondern nur eine kontinuierliche Menge vorhersagen kann.[5] Als Beispiel eignet sich hier die Wettervorhersage. Während bei der Klassifikation unterschieden werden kann, ob es am Folgetag kalt oder warm wird, kann die Regression im Gegenzug einen exakten Temperaturwert prognostizieren.

2.2 Clustering und Dimensionsreduktion als Teilbereiche des unüberwachten Lernens

Der entscheidende Unterschied zwischen dem überwachtem Lernen und dem unüberwachtem Lernen liegt in der Aufbereitung der Trainingsdaten. Während die Datensätze bei der Klassifikation und der Regression eine Bezeichnung haben und zu den passenden Ausgabedaten gehören, erhält der Computer beim Clustering und der Dimensionsreduktion nur Eingabedaten ohne jegliche Merkmale.[6] Der Algorithmus hat nun die Aufgabe, die Datenpunkte in eine Anzahl von Gruppierungen aufzuteilen, sodass sich Datenpunkte derselben Gruppe ähnlich sind.
Das Ziel der Dimensionsreduktion ist, einen multidimensionalen Datensatz in einen niedrig dimensionalen Raum zu überführen und dabei keine wertvollen Informationen zu verlieren.[7] Es wird versucht, die Gesamtvarianz mithilfe einer möglichst geringen Anzahl an Dimensionen zu erhalten.

---

[2] vgl. Aunkofer, Benjamin: *Maschinelles Lernen: Klassifikation vs Regression*, `https://data-science-blog.com/blog/2017/12/20/maschinelles-lernen-klassifikation-vs-regression/` [Zugriff am 11.07.2023]
[3] vgl. IBM (Hrsg.): *Was ist überwachtes Lernen?* `https://www.ibm.com/de-de/topics/supervised-learning` [Zugriff am 11.07.2023]
[4] vgl. Novustat (Hrsg.) *Regression Statistik*, `https://novustat.com/statistik-glossar/regression-statistik.html` [Zugriff am 11.07.2023]
[5] vgl. Gupta, Sakshi: *Regression vs. Classification in Machine Learning: What's the Difference?*, `https://www.springboard.com/blog/data-science/regression-vs-classification` [Zugriff am 29.12.2022]
[6] vgl. Delua, Julianna: *Supervised vs. Unsupervised Learning: What's the Difference?*, `https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning` [Zugriff am 11.07.2023]
[7] vgl. Zheng, Alice; Casari, Amanda: *Merkmalskonstruktion für Machine Learning - Prinzipien und Techniken der Datenaufbereitung*, 1. Aufl., Springfield (Vereinigte Staaten von Amerika), O'Reilly, 2019

3 Analyse essentieller Clustering-Algorithmen

3.1 K-Means als einfache und effiziente Clustering-Methode

Funktionsweise des K-Means Algorithmus

Der K-Means Algorithmus ist aufgrund seiner einfachen Implementierung und Effizienz einer der beliebtesten Clustering-Algorithmen. Zur Zuordnung der Punkte zu ihren jeweiligen Gruppierungen verwendet er Prototypen. Dies bedeutet, dass jedes Cluster durch einen Schwerpunkt (Prototyp) definiert wird. Der Algorithmus verlangt demzufolge auch einen Parameter: die Anzahl der Cluster beziehungsweise Schwerpunkte k.[8] Nachfolgend wird die Funktionsweise des K-Means Algorithmus exemplarisch in einem Flussdiagramm abgebildet:

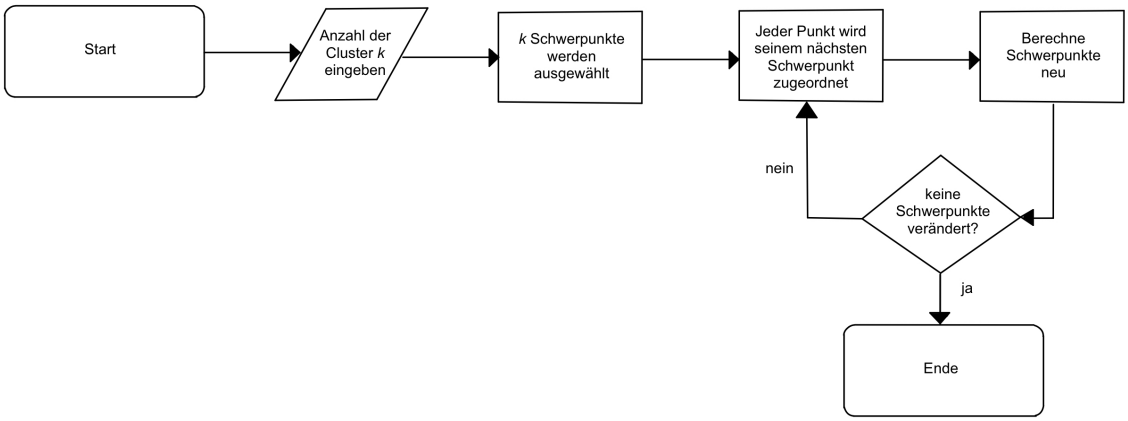


Abbildung 1: Flussdiagramm zur Veranschaulichung der Funktionsweise des K-Means Algorithmus

Vor- und Nachteile des K-Means Algorithmus

Als ein Vorteil von K-Means ist zu betonen, dass es sich um ein relativ einfaches und intuitives Verfahren handelt. Es erfordert nur eine geringe Rechenkapazität und kann daher auch bei großen Datensätzen effizient eingesetzt werden. Entsprechend lässt es sich auch sehr einfach implementieren. [9]
Trotz dieser Vorzüge besitzt K-Means einige Schwachstellen. Ein Hauptproblem besteht darin, dass die Anzahl der Cluster im Voraus festgelegt werden muss, was in der Praxis oft nicht einfach ist. Darüber hinaus sind Ausreißerpunkte eine Herausforderung für K-Means, da diese aufgrund der euklidischen Distanzberechnung eine hohe Gewichtung erhalten können. Außerdem ist K-Means nicht gut geeignet für Datensätze, deren Cluster nicht sphärisch sind oder unterschiedliche Größen und Streuungen aufweisen.[10]
Letztlich ermöglicht K-Means nur eine harte Clusterzuordnung. Dies bedeutet, dass jeder Datenpunkt genau einem Cluster zugewiesen wird. Im Gegensatz dazu stehen Verfahren wie beispielsweise Gaussian-Mixture-Models, bei denen eine gewisse Unschärfe im Clustering quantifiziert werden kann (siehe Kap. 3.3).

[8] vgl. Raschka, Sebastian; Mirjalili, Vahid: *Python Machine Learning - Machine Learning and Deep Learning with Python, scikit-learn, and Tensorflow 2*, 3. Aufl., Birmingham (Vereinigtes Königreich), Packt, 2019
[9] vgl. Google (Hrsg.): *k-Means Vor- und Nachteile*, `https://developers.google.com/machine-learning/clustering/algorithm/advantages-disadvantages?hl=de` [Zugriff am 30.10.2023]
[10] vgl. Technische Universität Dortmund (Hrsg.): *Limitations of k-Means Clustering*, `https://dm.cs.tu-dortmund.de/en/mlbits/cluster-kmeans-limitations/` [Zugriff am 30.10.2023]

3.2 DBSCAN zur dichtebasierten Segmentierung

Funktionsweise von DBSCAN

Ein weiterer Clustering-Algorithmus trägt den Namen „DBSCAN“. Dabei stellt diese Bezeichnung eine Abkürzung für „Density-Based Spatial Clustering of Applications with Noise“ dar, was sinngemäß mit „dichtebasierte räumliche Clusteranalyse mit Rauschen“ übersetzt werden kann.[11] Die Grundidee dieses Algorithmus basiert auf dem Prinzip der „Dichteverbundenheit“, welches aussagt, dass zwei Punkte genau dann als *dichtverbunden* gelten, wenn eine Kette von *dichten* Punkten (sogenannten Kernpunkte) existiert. Diese Kernpunkte müssen im Abstand ϵ mindestens $minPts$ viele Punkte besitzen. Dabei symbolisiert ϵ den maximalen Abstand zwischen zwei Punkten, damit diese noch als benachbart gelten, wohingegen $minPts$ die Mindestanzahl an Punkten vorgibt, mit denen ein Punkt benachbart sein muss, damit dieser als Kernpunkt gilt.[12] Die Gesamtheit der Punkte, die durch dieselben Kernpunkte miteinander verbunden sind, bezeichnet man als Cluster. Punkte, welche keinem Cluster zugeordnet werden können, nennt man Rauschen. Der Algorithmus unterscheidet also zusammenfassend zwischen drei Arten von Punkten:

- *Kernpunkte* sind Punkte, welche *dicht* sind und damit mindestens $minPts$ viele Punkte als Nachbarn haben, [13]
- *dichte-erreichbare Punkte* sind Punkte, die selbst nicht *dicht* sind, aber von einem Kernpunkt aus erreichbar und somit Teil des Clusters sind und [14]
- *Rauschpunkte*, die weder *dicht* noch *dichte-erreichbar* sind (siehe Anhang S. 17 Abb. 7). [15]

Nachfolgend ist die Funktionsweise des DBSCAN-Algorithmus als Pseudocode abgebildet: [16]

Algorithm 1: Funktionsweise des DBSCAN-Algorithmus

```
Funktion dbscan(Datensatz D, Epsilon eps, MinPunkte minPts):
    ClusterID = 0
    for jeden Punkt p in D do
        if p ist nicht besucht then
            markiere p als besucht
            Nachbarn = findeNachbarn(p, eps)
            if Anzahl von Nachbarn ≥ minPts then
                erhöhe ClusterID
                füge p und alle Nachbarn dem Cluster mit ClusterID hinzu
```

[11] vgl. Chauhan, Nagesh Singh: *DBSCAN Clustering Algorithm in Machine Learning*, https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html [Zugriff am 30.10.2023]

[12] vgl. Busch, Michael: *Analyse dichtebasierter Clusteralgorithmen am Beispiel von DBSCAN und MajorClust*, https://downloads.webis.de/theses/papers/busch_2005.pdf [Zugriff am 26.11.2023]

[13] vgl. Seidl, Prof. Dr. Thomas: *Knowledge Discovery in Databases I (WS 2020/21) -Dichtebasiertes Clustering*, https://www.dbs.ifi.lmu.de/Lehre/KDD/SS14/skript/KDD-3-Clustering-2.pdf [Zugriff am 27.05.2023]

[14] vgl. Grellmann, Martin: *DBSCAN (Density-Based Spatial Clustering of Applications with Noise)*, https://martin-grellmann.de/dbscan-density-based-spatial-clustering-of-applications-with-noise [Zugriff am 27.05.2023]

[15] vgl. Lippke, Steffen: *DBSCAN und BIRCH einfach erklärt und angewandt*, https://lippke.li/dbscan-und-birch/ [Zugriff am 27.05.2023]

[16] vgl. Sharma, Abhishek: *How to Master the Popular DBSCAN Clustering Algorithm for Machine Learning*, https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/ [Zugriff am 30.10.2023]

Der DBSCAN-Algorithmus beginnt mit einem zufällig gewählten, noch nicht besuchten Punkt im Datensatz. Wenn dieser Punkt eine ausreichende Anzahl an eng beieinander liegenden Nachbarn besitzt, wird ein neues Cluster gebildet. Ansonsten wird der Punkt als Rauschen markiert. Der Algorithmus wiederholt den Prozess, indem er alle benachbarten Punkte überprüft, um das Cluster zu erweitern. Dies geschieht solange, bis keine weiteren Punkte gefunden werden, die nahe genug beieinander liegen. Sobald kein weiterer Punkt dem aktuellen Cluster zugeordnet werden kann, beginnt der Algorithmus erneut mit einem neuen, noch nicht besuchten Punkt aus dem Datensatz. Dabei gibt die Funktion $findeNachbarn(p, eps)$ alle Datenpunkte aus dem Datensatz zurück, die innerhalb eines definierten Abstands (ϵ) vom gegebenen Punkt p liegen.

Vor- und Nachteile von DBSCAN

DBSCAN weist damit einige sehr vorteilhafte Eigenschaften auf. So muss die Anzahl der zu findenden Cluster nicht im Voraus bekannt sein, wie es bei K-Means der Fall ist. Des Weiteren muss die Form eines Clusters nicht sphärisch sein, sondern kann jeder beliebigen Form entsprechen und trotzdem noch als ein Cluster erkannt werden. Außerdem ist es dem Algorithmus möglich, innere Punkte (die in Clustern liegen) und äußere Punkte, also Rauschen, zu erkennen.[17] Trotz der vielen Vorteile im Vergleich zu K-Means besitzt der Algorithmus zwei essentielle Nachteile. Dabei ist zuerst anzuführen, dass das Ergebnis sehr stark von den beiden Parametern ϵ und $minPts$ abhängig ist, weshalb es einer zeitaufwendigen Optimierung eben jener bedarf, um ein bestmögliches Ergebnis zu erzielen. Das Ergebnis des Clustering-Prozesses mit DBSCAN bei verschiedenen Parameterkombinationen ist in der folgenden Abbildung dargestellt:

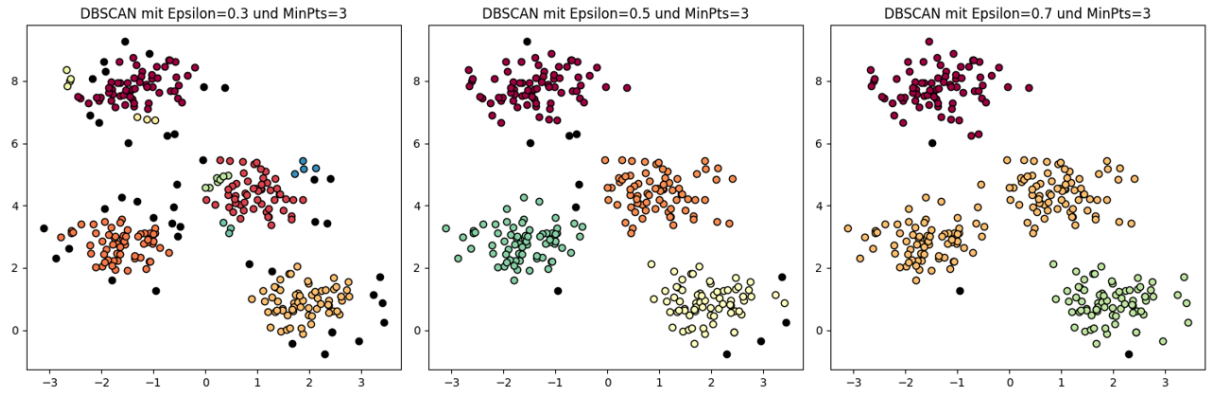


Abbildung 2: Vergleich der Leistung des DBSCAN-Algorithmus bei verschiedenen Parameterkombinationen

Darüber hinaus werden bei kleinen Datensätzen, welche nur wenige Punkte umfassen, sehr viele Punkte als Rauschen klassifiziert, womit keine aussagekräftigen Ergebnisse entstehen.

[17] vgl. Ester, Martin; Kriegel, Hans-Peter; Sander, Jörg; Xu, Xiaowei: *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*, `https://www.dbs.ifi.lmu.de/Publikationen/Papers/KDD-96.final.frame.pdf` [Zugriff am 30.10.2023]

3.3 Gaussian-Mixture-Models zur wahrscheinlichkeitsbasierten Sortierung

Funktionsweise des Gaussian-Mixture-Models

Gaussian-Mixture-Models (GMMs) sind stochastische Modelle, die eine Kombination von normalverteilten Komponenten verwenden, um Datenpunkte zu modellieren.[18] Im Gegensatz zu K-Means und DBSCAN, die „harte“ Zuweisungen von Datenpunkten zu Clustern machen, weist ein GMM jedem Datenpunkt eine gewisse Wahrscheinlichkeit zu, in jedem der vorhandenen Cluster zu liegen. Das bedeutet, dass Datenpunkte mit niedriger Wahrscheinlichkeit zwar in der Nähe von Clusterzentren liegen können, aber nicht eindeutig einem spezifischen Cluster zugeordnet werden können. Mathematisch gesehen kann ein GMM als Linearkombination von normalverteilten Dichtefunktionen dargestellt werden:[19]

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \tag{1}$$

Hierbei stellt K die Anzahl der Cluster dar, π_k symbolisiert das Gewicht des k-ten Clusters, $\boldsymbol{\mu_k}$ und $\boldsymbol{\Sigma_k}$ sind der Mittelwert und die Kovarianzmatrix des k-ten Clusters und $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})$ ist die Dichtefunktion einer multivariaten Normalverteilung mit Mittelwert $\boldsymbol{\mu_k}$ und Kovarianzmatrix $\boldsymbol{\Sigma_k}$. Das Ziel von GMMs besteht darin, die Parameter $\pi_k, \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}$ so zu schätzen, dass sie am besten zu den Daten passen.
Zur Vereinfachung wird oft angenommen, dass alle Cluster dieselbe Kovarianzmatrix $\boldsymbol{\Sigma}$ teilen. In diesem Fall reduziert sich das GMM auf einen Satz von Normalverteilungen, die jeweils durch ihren Mittelwert $\boldsymbol{\mu_k}$ und ihr Gewicht π_k charakterisiert sind.
GMMs arbeiten mithilfe des $Expectation - Maximization$ (EM) Algorithmus. Der EM-Algorithmus ist ein iteratives Verfahren, das aus zwei Schritten besteht: dem E-Schritt und dem M-Schritt. [20] Im E-Schritt wird, basierend auf den aktuellen Schätzungen für die Parameter, für jeden Datenpunkt die Wahrscheinlichkeit berechnet, dass dieser zu einem bestimmten Cluster gehört. Im M-Schritt werden die Parameter des GMMs aufbauend auf den Ergebnissen des E-Schritts aktualisiert.[21] Diese Schritte werden abwechselnd wiederholt, bis die Parameter konvergieren. Dies ist im folgenden Pseudocode vereinfacht dargestellt:

Algorithm 2: EM-Algorithmus von Gaussian-Mixture-Models

```
Funktion EM(Datensatz D):
    Initialisiere Parameter für alle Cluster
    while ! HatKonvergiert() do
        // E-Schritt
        for jeden Datenpunkt d_i in D do
            BerechneWahrscheinlichkeiten(d_i)
        // M-Schritt
        AktualisiereParameter()
```

[18] vgl. Dadi, Harihara; Venkatesh, P.; Poornesh, P.; L., Narayana Rao; Kumar, N. : *Tracking Multiple Moving Objects Using Gaussian Mixture Model*, https://www.researchgate.net/publication/305709395_Tracking_Multiple_Moving_Objects_Using_Gaussian_Mixture_Model[Zugriff am 30.10.2023]
[19] vgl. Grosse, Roger; Srivastava, Nitish: https://www.cs.toronto.edu/rgrosse/csc321/mixture_models.pdf [letzter Zugriff 30.10.2023]
[20] vgl. Foley, Daniel: *Gaussian Mixture Modelling (GMM) - Making Sense of Text Data using Unsupervised Learning*, https://towardsdatascience.com/gaussian-mixture-modelling-gmm-833c88587c7f [Zugriff am 30.10.2023]
[21] vgl. Lazyprogrammer (Hrsg.): *Unsupervised Learning I - Gaussian Mixture Model (GMM)*, https://lazyprogrammer.me/mlcompendium/clustering/gmm.html [Zugriff am 30.10.2023]

Vor- und Nachteile des Gaussian-Mixture-Models

Zusammenfassend können GMMs Cluster in Datensätzen identifizieren, die eine sehr komplexe Struktur aufweisen (zum Beispiel wenn die Cluster überlappen oder nicht völlig separierbar sind). In solchen Fällen treffen K-Means und DBSCAN auf ihre Grenzen (siehe Anhang S. 17 Abb. 8). Auch können die Unsicherheiten der Zuordnung eines Datenpunktes zu einem Cluster quantifiziert werden, indem die Wahrscheinlichkeiten der Zugehörigkeiten berechnet werden.[22] Ähnlich wie DBSCAN sind GMMs auch nicht auf kugelförmige Cluster beschränkt und sind damit deutlich flexibler bei der Modellierung von Daten. Diese Angepasstheit wird mittels der Kovarianzmatrix erreicht, welche jeder normalverteilten Komponente angepasst werden kann. Doch haben GMMs auch Nachteile. So ist ihr Hauptnachteil die höhere Komplexität im Vergleich zu anderen Clustering-Verfahren, was zu einer längeren Berechnungslaufzeit führen kann. Des Weiteren benötigen GMMs eine größere Anzahl an Datenpunkten, um sinnvolle Cluster identifizieren zu können. Insbesondere dann, wenn viele einzelne Cluster vorliegen, besteht ein erhöhter Bedarf an Datenpunkten, damit die Parameter ermittelt werden können.[23] Diese müssen zudem auch sorgfältig gewählt werden, da geringfügige Abweichungen vom Optimalwert schon stark abweichende Clustering-Ergebnisse hervorrufen können.

3.4 Hierarchisches Clustering für distanzbasierte Strukturbildungen

Funktionsweise des hierarchischen Clusterings

Der letzte Clustering-Algorithmus, der in dieser Arbeit betrachtet werden soll, ist die hierarchische Zuordnung von Punkten zu entsprechenden Gruppierungen. Es existieren zwei Haupttypen von hierarchischen Clustering-Methoden: agglomeratives Clustering und divisives Clustering.[24] Gemäß der Methode des agglomerativen Clusterings werden einzelne Datenpunkte anfangs als separate Cluster behandelt, die anschließend schrittweise zu umfangreicheren Gruppierungen zusammengeführt werden. Dies geschieht solange, bis schließlich ein einziges, gemeinsames Cluster alle Punkte umfasst. Beim divisiven Clustering wird der Prozess umgekehrt ausgeführt, indem zunächst alle Datenpunkte als ein Cluster betrachtet werden und dann schrittweise in immer kleinere Cluster unterteilt werden. Die Zuordnung der Punkte zu den jeweiligen Gruppierungen ist dabei abhängig von ihrer Distanz zueinander.
Das Prinzip des hierarchischen Clusterings findet sich auch in der Biologie wieder. Dort können verschiedene Tierarten anhand ihrer genetischen Merkmale in Gattungen oder Familien zusammengefasst werden. Dieses lässt sich auch auf abstrakte Daten übertragen.
Die durch das hierarchische Clustern entstehende Struktur kann zudem durch ein Dendrogramm abgebildet werden. Ein Dendrogramm ist dabei ein Baumdiagramm zur hierarchischen Darstellung von Clustern.[25] Auf der x-Achse im Dendrogramm sind die Indizes der Punkte abgebildet, wobei man auf der y-Achse die Entfernung dieser Punkte (oder Cluster) zueinander ablesen kann. Horizontale Linien symbolisieren die Zusammenführung von zwei Clustern, wobei vertikale Linien Auskunft darüber geben, welche Cluster beziehungsweise Punkte zusammengeführt wurden. Nachfolgend ist ein solches Dendrogramm exemplarisch abgebildet.

[22] vgl. Singh, Aarti: *Gaussian Mixture Models: Lecture from the ML-Department of Carnegie Mellon University*, `https://www.cs.cmu.edu/~aarti/Class/10315_Fall19/lecs/Lecture20.pdf` [Zugriff am 30.10.2023]
[23] vgl. Ellis, Christina: *When to use gaussian mixture models*, `https://crunchingthedata.com/when-to-use-gaussian-mixture-models/` [Zugriff am 30.10.2023]
[24] vgl. Roux, Maurice: *A Comparative Study of Divisive and Agglomerative Hierarchical Clustering Algorithms*, `https://hal.science/hal-02085844/document` [Zugriff am 30.10.2023]
[25] vgl. Max, Edraw: *Dendogramm*, `https://www.edrawsoft.com/de/article/what-is-dendrogram.html` [Zugriff am 29.12.2022]

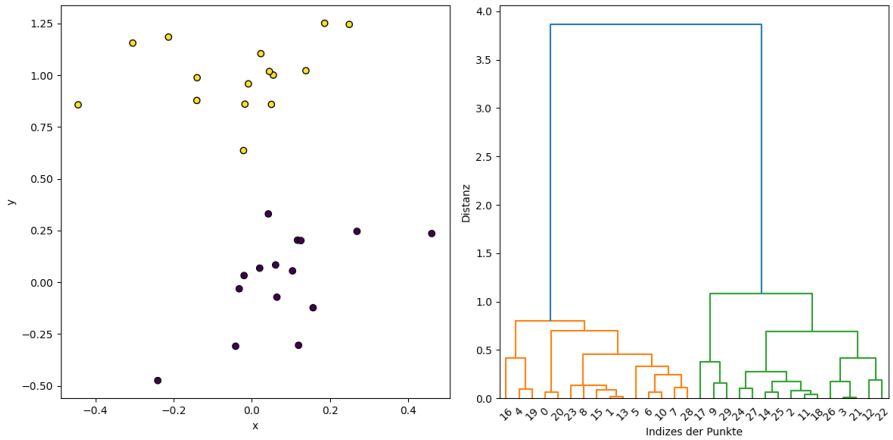


Abbildung 3: Punktegruppierungen nach dem Prinzip des hierarchischen Clusterings

Vor- und Nachteile des hierarchischen Clusterings

Als Vorteil ist zuerst anzumerken, dass hierarchisches Clustering sehr leicht zu verstehen ist und der Algorithmus nicht auf komplexen mathematischen Sachverhalten basiert. Das Dendrogramm ermöglicht dem Benutzer, die gebildeten Cluster klar und leicht verständlich zu visualisieren. Ein weiterer Vorteil ist, dass hierarchisches Clustering ausgezeichnet für Datenpunkte funktioniert, die eine natürliche hierarchische Beziehung haben (beispielsweise hierarchische Organisationsstrukturen in Unternehmen). Es kann auch verschiedene Formen von Clustern behandeln, da es nicht auf einer vordefinierten Annahme über die Gestalt der Cluster basiert.[26]

Dennoch weist diese Methode auch einige Nachteile auf. Einer davon ist ihre Rechenintensität. Hierarchisches Clustering erfordert eine Distanzberechnung zwischen jedem Paar von Datenpunkten, was zu einer potenziellen Rechenkomplexität führt. Somit ist dieser Algorithmus ausschließlich für kleine Punktemengen geeignet. Zudem ist das hierarchische Clustering anfällig für Störungen. Einmal zusammengeführte oder aufgeteilte Datenpunkte können in späteren Stufen nicht mehr geändert werden, was in einer unbeständigen Clusterstruktur resultieren kann, wenn es starke Ausreißer oder Rauschen in den Daten gibt.[27]

3.5 Begründete Auswahl komplementärer Clustering-Algorithmen

Bei der Auswahl verschiedener Clustering-Algorithmen ist eine wesentliche Herausforderung zu bewältigen: Es gilt, eine umfassende und vielseitige Anwendbarkeit für diverse Szenarien zu gewährleisten, ohne dabei die Anzahl der Algorithmen übermäßig zu erhöhen, was wiederum die Effizienz und die Durchsichtigkeit des Gesamtverfahrens beeinträchtigen könnte. Um einen Kompromiss zu schaffen, wurde bei der Auswahl entsprechender Methoden besonders auf komplementäre Eigenschaften zwischen diversen Algorithmen geachtet, sodass mit vergleichsweise wenigen Algorithmen ein sehr breites Spektrum an Datenerscheinungen abgedeckt werden kann (siehe Anhang S. 18 Abb. 9). Durch dieses Auswahlverfahren gelang es, eine Zusammenstellung von Algorithmen zu treffen, welche Anwendung bei geometrischen, dichtebasierten, statistischen und hierarchischen Verteilungen findet.

[26] vgl. Sultana, Shaik Irfana: *How the Hierarchical Clustering Algorithm Works*, `https://dataaspirant.com/hierarchical-clustering-algorithm` [Zugriff am 30.10.2023]

[27] vgl. Block, Tim: *What are the Strengths and Weaknesses of Hierarchical Clustering?*, `https://www.displayr.com/strengths-weaknesses-hierarchical-clustering/` [Zugriff am 30.10.2023]

4 Evaluierungsmethoden des Clusterings

4.1 Bedeutung von Bewertungsmetriken im Clustering

Evaluierungsmethoden ermöglichen es, die Qualität und Genauigkeit des Clusterings mithilfe eines der im vorherigen Kapitel vorgestellten Algorithmen zu bestimmen. Somit können mithilfe dieser Metriken Aussagen über die Genauigkeit der Zuordnung der Punkte durch die jeweilige Clustering-Methode getroffen werden. Zur verbesserten Anschaulichkeit der aktuellen Position im Prozessablauf der Clusteranalyse ist im Anhang ein Flussdiagramm abgebildet (siehe Anhang S. 19 Abb. 10).

4.2 Bewertung mit dem Inertia-Wert

Der Inertia-Wert gibt die Entfernung der Punkte innerhalb eines Clusters an und liefert Auskunft über die Qualität der einzelnen Clusterzusammenstellungen. Dazu wird die Distanz aller Datenpunkte zu ihrem entsprechenden Clusterzentrum berechnet. Mathematisch lässt sich dies als Summe ausdrücken, wobei für jeden der N Datenpunkte der Abstand zwischen dem Datenpunkt x_i und dem dazugehörigen Zentrum des Clusters C_k ausgerechnet wird. Anschließend wird dieser quadriert:[28]

$$I = \sum_{i=1}^{N} (x_i - C_k)^2 \tag{2}$$

Dieser Inertia-Wert kann von null bis unendlich reichen. Dabei sind Werte sehr nah an null präferiert und stellen damit dar, dass (fast) alle Datenpunkte eng um ein Clusterzentrum liegen.
Der Inertia-Wert dient nicht nur zur Evaluierung des Lernprozesses, sondern kann zudem auch zur Bestimmung der Anzahl der Cluster verwendet werden. Dabei ist zu beachten, dass mit zunehmender Anzahl an Clustern auch der Inertia-Wert weiter abnimmt, da irgendwann jeder Datenpunkt einem Cluster entspricht und damit der Inertia-Wert gleich null ist. Um also nun die Anzahl der Cluster k zu finden, muss die Anzahl der Cluster ermittelt werden, ab welchen der Inertia-Wert asymptotisch gegen null konvergiert.

4.3 Bewertung mittels des Silhouette-Scores

Eine weitere Möglichkeit, um den Lernprozess zu evaluieren, ist der Silhouette-Score S. Dieser Wert vergleicht nicht nur die Abstände innerhalb eines Clusters, sondern zudem auch die Abstände zu den anderen Clustern:[29]

$$S_i = \frac{b_i - a_i}{max(a_i, b_i)} \tag{3}$$

Dabei ist a_i der mittlere Abstand aller Punkte innerhalb des Clusters i und b_i der mittlere Abstand zu allen anderen Gruppierungen. Der Wertebereich geht dabei von minus eins bis eins, wobei bei Werten nahe der eins die Punkte nicht nur sehr gut innerhalb des eigenen Clusters liegen, sondern die Cluster auch noch sehr weit voneinander entfernt sind. Diese Bewertungsmetrik funktioniert jedoch nicht für alle Punkteanordnungen, sondern gibt teilweise auch falsche Ergebnisse aus (siehe Anhang S. 18 Abb. 9).

[28] vgl. Brus, Patrick: *Clustering: How to Find Hyperparameters using Inertia*, `https://towardsdatascience.com/clustering-how-to-find-hyperparameters-using-inertia-b0343c6fe819` [Zugriff am 30.12.2022]
[29] vgl. Gaido, Marco: *Distributed Silhouette Algorithm: Evaluating Clustering on Big Data*, `https://arxiv.org/pdf/2303.14102.pdf` [Zugriff am 30.12.2022]

5 Detektion und Analyse von Anomalien in den Datensätzen

Anomalien oder Ausreißer sind Datenpunkte, die sich deutlich von der Mehrheit der anderen Datenpunkte in einem Datensatz unterscheiden. Ihre Detektion ist ein entscheidender Zwischenschritt zur Minimierung der Ungenauigkeit bei der Extraktion charakteristischer Merkmale (gleichbedeutend mit *Features*) der Daten. Anomalien passen häufig nicht zu den Clustern und liegen daher weit entfernt von allen Clusterzentren. Sie werden in der Regel durch Messfehler herbeigeführt, weswegen Ausreißer in den meisten Fällen aus vorhandenen Datensätzen aussortiert werden müssen, um die Clustering-Ergebnisse nicht zu verfälschen. Die Detektion von Ausreißern ist zurzeit Gegenstand aktueller Forschung. Daher existiert noch kein System, welches Anomalien in einem Datensatz schnell und zuverlässig erkennt.
Dennoch wurde eine Lösung entwickelt, welche den Anforderungen genügt und in der Lage ist, einen Großteil der Ausreißer korrekt zu identifizieren. Hierbei konnte auf das Vorwissen von den verschiedenen Clustering-Algorithmen zurückgegriffen werden: Der DBSCAN-Algorithmus ist mit den richtigen Eingabeparametern in der Lage, solche Ausreißer zu identifizieren. Ihm werden als Hyperparameter ϵ und $minPts$ zugewiesen. Die Methode zur Bestimmung von ϵ im DBSCAN-Algorithmus basiert auf der Berechnung der Distanzen zu den k nächsten Nachbarn für jeden Datenpunkt in einem Datensatz. Nach dem Sortieren dieser Distanzen wird der ϵ-Wert so gewählt, dass er dem Abstand zum weitesten der k nächsten Nachbarn für einen bestimmten Prozentsatz der Datenpunkte entspricht.[30] Somit ließ sich ein Parameter schon sehr genau bestimmen. Nun musste noch der korrekte Wert für $minPts$ gefunden werden. Zur Bewertung der verschiedenen Cluster, die durch das Variieren von $minPts$ entstanden, diente der Silhouette-Score als Evaluierungsmetrik. Zuletzt wurde sich für den Wert von $minPts$ entschieden, der den höchsten Silhouette-Score erzielt hat.
Jedoch ist DBSCAN alleine noch nicht in der Lage, alle Ausreißer zu identifizieren. In der folgenden Abbildung ist auf der linken Seite das Ergebnis des Clustering-Prozesses mit DBSCAN repräsentiert, wobei auf der rechten Seite alle eigentlich korrekten Anomalien markiert wurden.

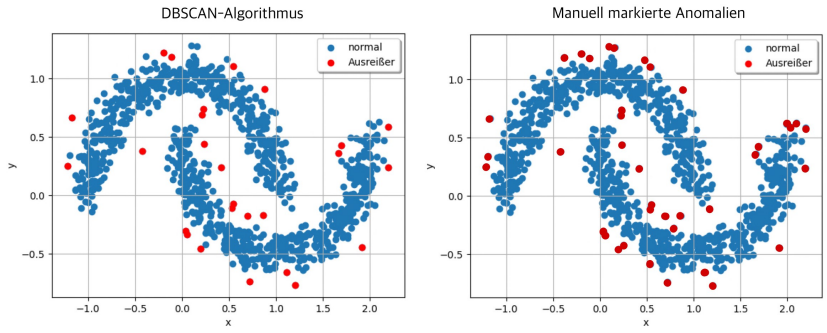


Abbildung 4: Vergleich der Performance des DBSCAN-Algorithmus zu manuell markierten Anomalien

Aufgrund dieser Ungenauigkeit wurde noch eine weitere Ausreißererkennungsmethode inkludiert, nämlich die „Local Outlier Factor“ (kurz LOF). Die LOF identifiziert Anomalien durch den Vergleich der lokalen Dichte eines Datenpunktes mit den Dichten seiner Nachbarn, wobei ein hoher LOF-Wert auf einen Ausreißer hinweist.[31] Abschließend wurden alle gefundenen Anomalien zusammengefügt und das Ergebnis entsprechend ausgegeben (siehe Anhang S. 19 Abb. 11).

[30] vgl. Sefidian, Amir Masoud: *How to determine epsilon and MinPts parameters of DBSCAN clustering*, `https://www.sefidian.com/2022/12/18/how-to-determine-epsilon-and-minpts-parameters-of-dbscan-clustering/` [Zugriff am 30.10.2023]
[31] vgl. Matzer, Michael: *Grundlagen Statistik & Algorithmen, Teil 7 - So deckt der Local Outlier Factor Anomalien auf*, `https://www.bigdata-insider.de/so-deckt-der-local-outlier-factor-anomalien-auf-a-803652/` [Zugriff am 30.10.2023]

6 Auswahl von Features für das neuronale Netz

6.1 Bestimmung der Form des Datensatzes

Die Bestimmung der Form des Datensatzes ist ein entscheidender Schritt bei der Selektion des am besten geeigneten Clustering-Algorithmus. Während beispielsweise K-Means sich für sphärische Clusterformen eignet, erzielt Gaussian-Mixture-Models ein besseres Ergebnis bei normalverteilten (also häufig elliptischen) Clusteranordnungen. Für die Erkennung der Form eines Datensatzes gibt es noch keinen fertigen Algorithmus, da dieses Problem aufgrund seiner Komplexität weiterhin Gegenstand aktueller Forschung ist. Dennoch wurde ein System geschaffen, welches basierend auf dem jeweiligen Eingabedatensatz approximierte Vorhersagen über die Form treffen kann. Um eine Aussage über diese tätigen zu können, mussten zunächst einmal Cluster angenähert werden. Hierbei wurde auf einen herkömmlichen Clustering-Prozess verzichtet, denn die Auswahl eines geeigneten Algorithmus wäre bereits das Ergebnis des neuronalen Netzes und damit nicht für die Feature-Bestimmung geeignet. Deshalb wurde entschieden, die Dichte als ausschlaggebendes Kriterium zu verwenden, wobei ein Gauß'scher Kerndichteschätzer verwendet wurde, um eine solche Dichtefunktion zu ermitteln.[32] Nach der Bestimmung dieser wurde für eine Höhe h, welche das Produkt der Durchschnittsdichte und einem Vorfaktor x ist, die Kontur bestimmt, die sich aus allen Punkte der Höhe h zusammensetzte. Um das beste x zu wählen, wurden die Konturen erneut mittels des bereits vorgestellten Silhouette-Scores bewertet (siehe Anhang S. 20 Abb. 12). Dabei wurden sehr rasch zufriedenstellende Resultate erzielt. Es fiel jedoch auf, dass in einigen Fällen die Konturen den Datensatz fehlerhaft widerspiegelten, weshalb ein zweites Verfahren implementiert wurde, um auch diese Fälle richtig abbilden zu können (siehe Anhang S. 20 Abb. 13).

Für dieses zweite Verfahren wurde zunächst ein zweidimensionales Histogramm erstellt[33], welches die Verteilung der gegebenen Datenpunkte verbildlicht. Um nun Konturen zu detektieren, wurde der sogenannte Sobel-Operator verwendet. Dieser dient zur Konturerkennung, indem er hochfrequente Bereiche eines Gradienten als Kante erkennt.[34] Mittels eines Gauß-Filters wurden die entstandenen Umrisse vereinfacht, welche dann der Ausgangspunkt für die Konturen waren. Auch hier wurde sich erneut der Hyperparameteroptimierung bedient, indem der σ-Parameter für die Unschärfe durch den Silhouette-Score optimiert wurde. Es wurden zunächst die besten Konturen evaluiert, die aus beiden Verfahren hervorgingen, und anschließend mithilfe des Silhouette-Scores verglichen. Anhand dieser Ergebnisse wurde die passendere beider Konturen identifiziert, die dann verwendet wurde, um detaillierte Aussagen über die Form der vermeintlichen Cluster sowie die allgemeine Struktur des Datensatzes zu treffen.

[32] vgl. Spodarev, Prof. Dr. Evgeny: *Dichteschätzer*, `https://www.uni-ulm.de/fileadmin/website_uni_ulm/mawi.inst.110/lehre/ss13/Stochastik_I/Skript_4.pdf` [Zugriff am 04.10.2023]

[33] vgl. Saddique, Asif: *Introduction to ROOT*, `https://indico.cern.ch/event/555909/contributions/2265949/attachments/1325123/1988911/Root_Lecture2.pdf` [Zugriff am 04.10.2023]

[34] vgl. Fisher, Robert/Simon Perkins/Ashley Walker/Erik Wolfart: *Sobel Edge Detector*, `https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm` [Zugriff am 04.10.2023]

6.2 Selektion weiterer Features zur exakten Charakterisierung des Datensatzes

Exzentrizität der Kontur

Die Form des Datensatzes als alleiniges Feature würde nicht ausreichen, um den Datensatz vollständig beschreiben zu können. Daher ist es notwendig, noch einige weitere Merkmale zu extrahieren, um eine genauere Aussage über die Beschaffenheit des Datensatzes treffen zu können. Hierbei kann die Exzentrizität der Konturpunkte einen ersten Eindruck über die Verteilung der Datenpunkte vermitteln. Ein Exzentrizitätswert nahe null deutet auf eine annähernd kreisförmige Kontur hin, während ein Wert nahe eins auf eine stark elliptische Form hinweist.

Verhältnis der konvexen Hülle zur Konturfläche

Ein weiteres maßgebliches Merkmal zur Beschreibung eines Datensatzes ist das Verhältnis der Fläche einer konvexen Hülle zur tatsächlichen Konturfläche. Ein niedriger Wert würde vermuten lassen, dass die Konturpunkte eng an der konvexen Hülle liegen, während ein hoher Wert auf das Vorhandensein von Einbuchtungen oder anderen Abweichungen von der konvexen Form verweist. Somit ermöglicht dieses Verhältnis einen Einblick in die allgemeine Geometrie des Datensatzes.

Kompaktheit der Datenpunkte

Die Kompaktheit gibt an, wie nahe die Datenpunkte um ihren Schwerpunkt innerhalb des Clusters verteilt sind. Ein höherer Kompaktheitswert würde bedeuten, dass die Datenpunkte weiter vom Schwerpunkt ihrer Gruppierung entfernt sind, während ein niedrigerer Wert impliziert, dass die Datenpunkte näher am an ihm positioniert sind.[35] Dieses Merkmal erweist sich als besonders aussagekräftig bei der Analyse der räumlichen Verteilung eines Datensatzes.

Kurtosis und Schiefe der Datenpunkte

Die Kurtosis und die Schiefe sind statistische Größen, die zur Erfassung der Punkteverteilung des Datensatzes dienen. Nachfolgend wird sich vorerst der Kurtosis gewidmet, wonach anschließend die Schiefe genauer erläutert wird.

1. Das statistische Maß der Kurtosis wird herangezogen, um zu bestimmen, in welchem Ausmaß die Verteilung der Datenpunkte spitzer oder flacher als eine Normalverteilung ausfällt. Sie ermöglicht es, Einblicke in die zentralen Bereiche und die Extrema der Datenverteilung zu erhalten, und erleichtert dadurch Rückschlüsse auf potentielle Ausreißer. Für Datenpunkte $x_1, x_2, \ldots, x_n$ mit dem Mittelwert $\bar{x}$ und der Standardabweichung s wird die Kurtosis K wie folgt berechnet:[36]

$$K = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s} \right)^4 \tag{4}$$

[35] vgl. DataNovia (Hrsg.): *Cluster Validation Statistics: Must Know Methods*, `https://www.datanovia.com/en/lessons/cluster-validation-statistics-must-know-methods` [Zugriff am 30.10.2023]

[36] vgl. National Institute of Standards and Technology (Hrsg.): *Measures of Skewness and Kurtosis*, `https://www.itl.nist.gov/div898/handbook/eda/section3/eda35b.htm` [Zugriff am 06.10.2023]

Hierbei ist $\bar{x}$ der Durchschnitt der Datenpunkte:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{5}$$

Die Variable s stellt die Standardabweichung des gegebenen Datensatzes dar:

$$s = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{6}$$

2. Die Schiefe gibt das Maß der Asymmetrie der Verteilung der Datenpunkte im Vergleich zur Normalverteilung an. Bei einer positiven Schiefe wird eine Verteilung charakterisiert, bei welcher der rechte Teil der Kurve weniger steil verläuft als der linke Graphenabschnitt (rechtsschiefe Verteilung). Eine negative Schiefe hingegen weist auf eine linksschiefe Verteilung hin, bei der der linke Teil der Kurve flacher als der rechte Teil verläuft. [37] Das Fehlen einer Schiefe lässt darauf schließen, dass die Daten symmetrisch um den Mittelwert verteilt sind. Für Datenpunkte $x_1, x_2, \ldots, x_n$ mit Mittelwert $\bar{x}$ und Standardabweichung s wird die Schiefe S wie folgt berechnet:[38]

$$S = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s} \right)^3 \tag{7}$$

Nachfolgend werden die jeweiligen möglichen Verteilungen der Kurtosis und der Schiefe in Referenz zur Normalverteilung abgebildet. Dabei befinden sich an der x-Achse die x-Koordinaten der Datenpunkte, wobei die y-Achse die Dichte der Punkte abbildet.

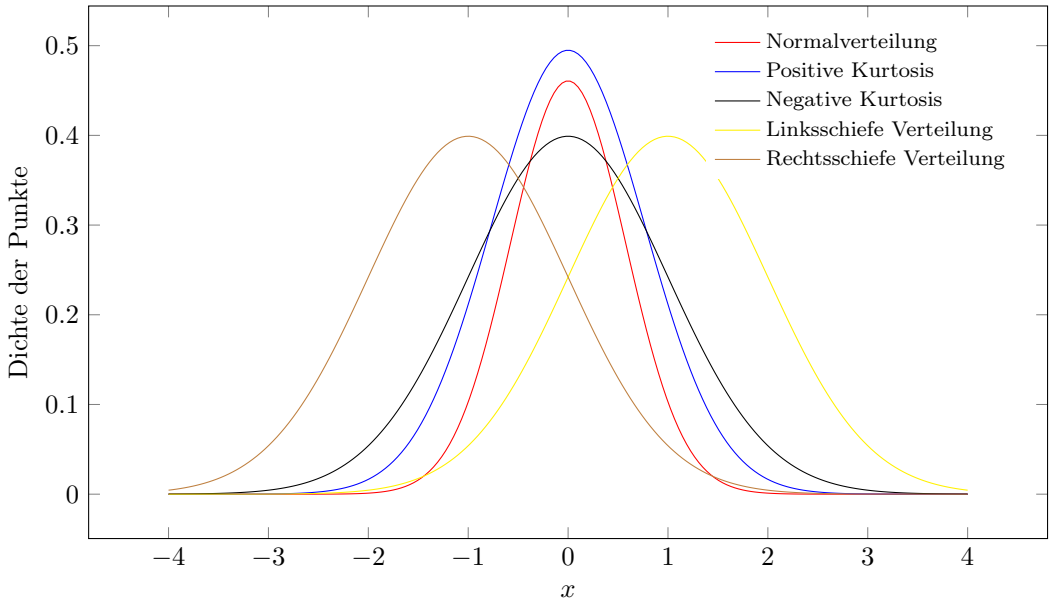


Abbildung 5: Verschiedene Punkteverteilungen im Vergleich zur Normalverteilung

[37] vgl. Gavali, Suvarna: *Skewness and Kurtosis: Quick Guide*, `https://www.analyticsvidhya.com/blog/2021/05/shape-of-data-skewness-and-kurtosis/` [letzer Zugriff: 30.10.2023]
[38] vgl. Weisstein, Eric W.: *Skewness*, `https://mathworld.wolfram.com/Skewness.html` [Zugriff am 06.10.2023]

7 Vorbereitung und Training des neuronalen Netzes

7.1 Generatoren als Basis zur Sammlung von Trainingsdaten

Um das neuronale Netz erfolgreich trainieren zu können, werden Generatoren zur Generierung von Trainingsdaten benötigt. Dabei wurde eine Klasse *DataAugmentor* erstellt, welche verwendet wird, um einzelne Datensätze leicht abzuändern. Dies ist besonders hilfreich, um die Vielfalt und Robustheit des neuronalen Netzes zu erhöhen. In dieser Klasse können zufällige Punkte entfernt und hinzugefügt, Datenpunkte skaliert und verschoben sowie die Daten um ihren Schwerpunkt gedreht werden. Nachfolgend ist die Klasse *DataAugmentor* inklusive ihrer Methoden in einem Klassendiagramm abgebildet.

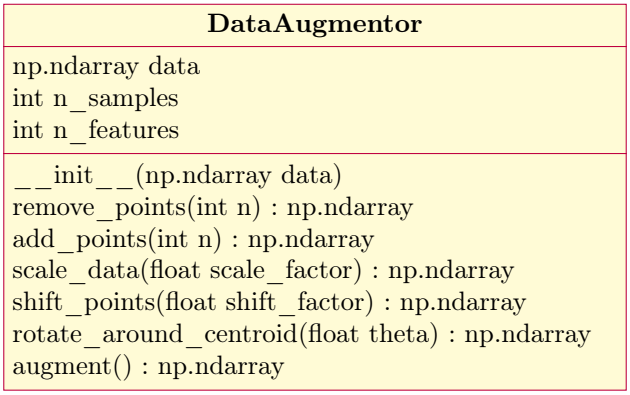


Abbildung 6: Klassendiagramm der Klasse *DataAugmentor*

Nachdem alle generierten Datensätzen in einer Liste abgespeichert wurden, werden für jeden Datensatz die Features berechnet und zusammen mit dem am besten geeigneten Clustering-Algorithmus für einen spezifischen Datensatz in einem $Dataframe$ abgespeichert. Schließlich wird dieser $Dataframe$ als Excel-Datei mit der *to_excel*-Methode gesichert, um ihn anschließend zum Training des neuronalen Netzes verwenden zu können. Der gesamte Prozess zur Kollektion von solchen Daten ist in Form eines Flussdiagrammes im Anhang abgebildet (siehe Anhang S. 21 Abb. 14).

7.2 Trainingsprozess des neuronalen Netzes

Das neuronale Netz stellt nun nur noch eine geringe Herausforderung dar. Zu Beginn des Trainingsprozesses des neuronalen Netzes werden die Daten der Excel-Tabelle in einen *pandas* $DataFrame$ eingelesen. Hierfür wurden verschiedene Anzahlen an Datensätzen generiert, von denen dann die berechneten Features in die Tabelle überschrieben werden. Dabei wurde festgestellt, dass eine höhere Anzahl an Datensätzen eine höhere Genauigkeit erzielte. Daher wurden 5184 Datensätze generiert und die berechneten Features in der Tabelle gesichert. Anschließend wird die Spalte, in der sich jeweils der am besten geeignete Algorithmus für den jeweiligen Datensatz befindet, kodiert, sodass die Algorithmen in numerische Werte umgewandelt werden. Zudem werden die skalierten Merkmale in Trainings- und Testdaten unterteilt. Für die Netzarchitektur wird ein sequentielles Modell verwendet, welches aus einer Eingabeschicht aus 48 Neuronen und einer Ausgabeschicht aus 32 Neuronen besteht. Diese optimalen Werte wurden durch das systematische Probieren verschiedener Parameterkombinationen ermittelt. Nach der Definition wird das Netzmodell trainiert und schließlich anhand der synthetischen Testdaten evaluiert, wobei eine Genauigkeit von $99,81\%$ erzielt wurde.

8 Zusammenfassung

Im Rahmen dieser Arbeit erfolgte eine intensive Auseinandersetzung mit der Thematik des Machine-Learnings und insbesondere des Clusterings. Dies umfasste ein umfangreiches Literaturstudium. Ergänzend dazu wurde an einem aufschlussreichen und informativen Praktikum zum Thema unter der Anleitung von Herrn Christopher Sobel vom Fraunhofer-Institut Nürnberg teilgenommen. Dadurch gelang es, einen Überblick über die verschiedenen Algorithmen und Verfahrensweisen zum Clustern von Datensätzen zu gewinnen und sich Wissen über die Funktionsweise der Algorithmen anzueignen. Dabei lag der Fokus auch auf deren Vor- und Nachteilen sowie auf den Situationen, in denen diese am besten geeignet sind.

Das Hauptziel war die Entwicklung eines Programms, das automatisch den passenden Algorithmus für das Clustering eines beliebigen Datensatzes auswählt. Von Beginn an war offensichtlich, dass einer der besten Ansätze dafür die Nutzung eines neuronalen Netzes ist. Ein erster Schritt war die Entwicklung einer effizienten Ausreißererkennung, um möglichst fehlerfrei mit den Datensätzen arbeiten zu können. Die größte Herausforderung lag in der Ermittlung aussagekräftiger Features, insbesondere in der Darstellung der Form der Cluster eines Datensatzes. Durch die passende Anwendung von Dichtegrafiken und Bilderkennung gelang es schließlich, für jeden Datensatz automatisiert eine Aussage zur Form in einem einzelnen Wert zu treffen. Auf der Basis dieses und weiterer Features begann dann das Training des neuronalen Netzes, welches reibungslos verlief. Nach einigen Tests stellte sich heraus, dass das Produkt mit sehr hoher Genauigkeit arbeitet.

Während des Schaffensprozesses fand ein regelmäßiger Austausch mit den Außenbetreuern vom Fraunhofer-Institut in Nürnberg, Herrn Dr. Wolfgang Felber und Herrn Christoph Sobel, statt. In regelmäßigen Abständen von wenigen Wochen wurden ihnen die Ergebnisse präsentiert und sie konnten mit vielen konstruktiven Ratschlägen unterstützen. Diese fachlich sehr fundierte Beratung half sehr, die Ergebnisse zu bewerten sowie die Vorgehensweise zu hinterfragen und zu überprüfen.

Bisher wurde das Verfahren hauptsächlich auf künstliche und nur einige wenige reale Datensätze angewandt. Um die Funktionalität und Brauchbarkeit für die Realität zu zeigen, könnte ein Problem gefunden werden, bei dem das Clustern von Datensätzen notwendig ist, und dieses mit Hilfe des neuronalen Netzes gelöst werden - zum Beispiel die Steuerung von Robotern unter der Vermeidung von Kollisionen. Im Zuge dessen wäre auch die Implementierung einer geeigneten Benutzeroberfläche von Vorteil.

Ein wesentliches Erweiterungsfeld bietet die Integration weiterer Clustering-Algorithmen. Derzeit ist das neuronale Netz in der Lage, aus vier verschiedenen Algorithmen den optimalen auszuwählen, was eine breite Abdeckung vieler Datensätze ermöglicht. Dennoch existieren spezialisierte Verfahren, die für bestimmte Datentypen effizienter sind, weshalb eine Erweiterung die Rechenprozesse erheblich beschleunigen könnte.

Das Gruppenklima war während der gesamten Bearbeitungszeit sehr ausgeglichen. Eine konkrete Arbeitsteilung wurde beim Programmieren nur selten vorgenommen, was durch die gemeinsame Arbeit an einem Google-Colab Dokument ermöglicht wurde. Grundlegende Ideen, wie zum Beispiel die Methodik der Formerkennung, wurden im direkten Austausch zwischen den Gruppenmitgliedern entwickelt. Das Schreiben der Kapitel verlief ebenfalls reibungslos, wobei die Abschnitte der Arbeit untereinander aufgeteilt wurden.

Zum Abschluss lässt sich sagen, dass auf ein sehr erfolgreiches Jahr zurückgeblickt werden kann. Um das Ziel zu erreichen, mussten einige Hürden überwunden werden. Dennoch hat die Herausforderung immer viel Freude bereitet und dabei geholfen, Vieles zu lernen - nicht nur auf fachlicher Seite. Letztlich konnte ein kleiner Beitrag zur von Jeff Bezos erwähnten digitalen Transformation erbracht werden.

9 Anhang

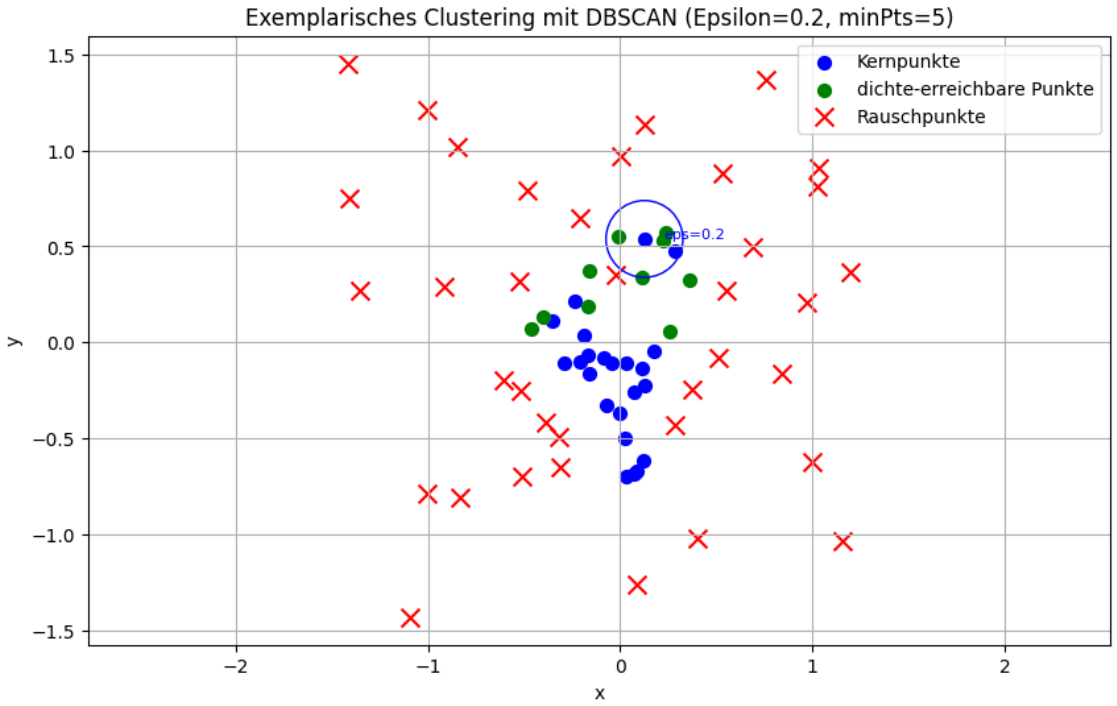


Abbildung 7: Punkteunterteilungen bei DBSCAN

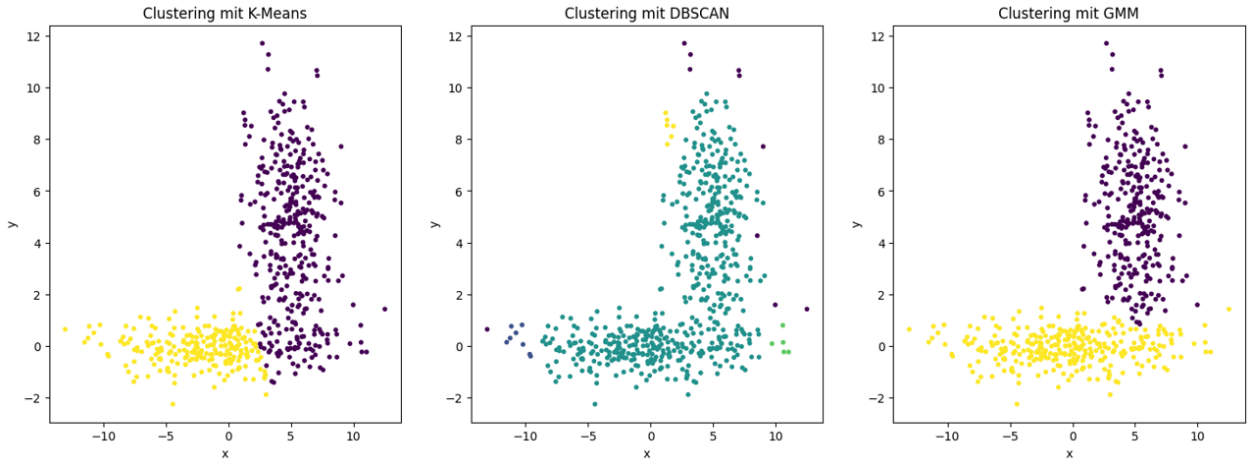


Abbildung 8: Clustering mithilfe von Gaussian-Mixture-Models im Vergleich zu K-Means und DBSCAN

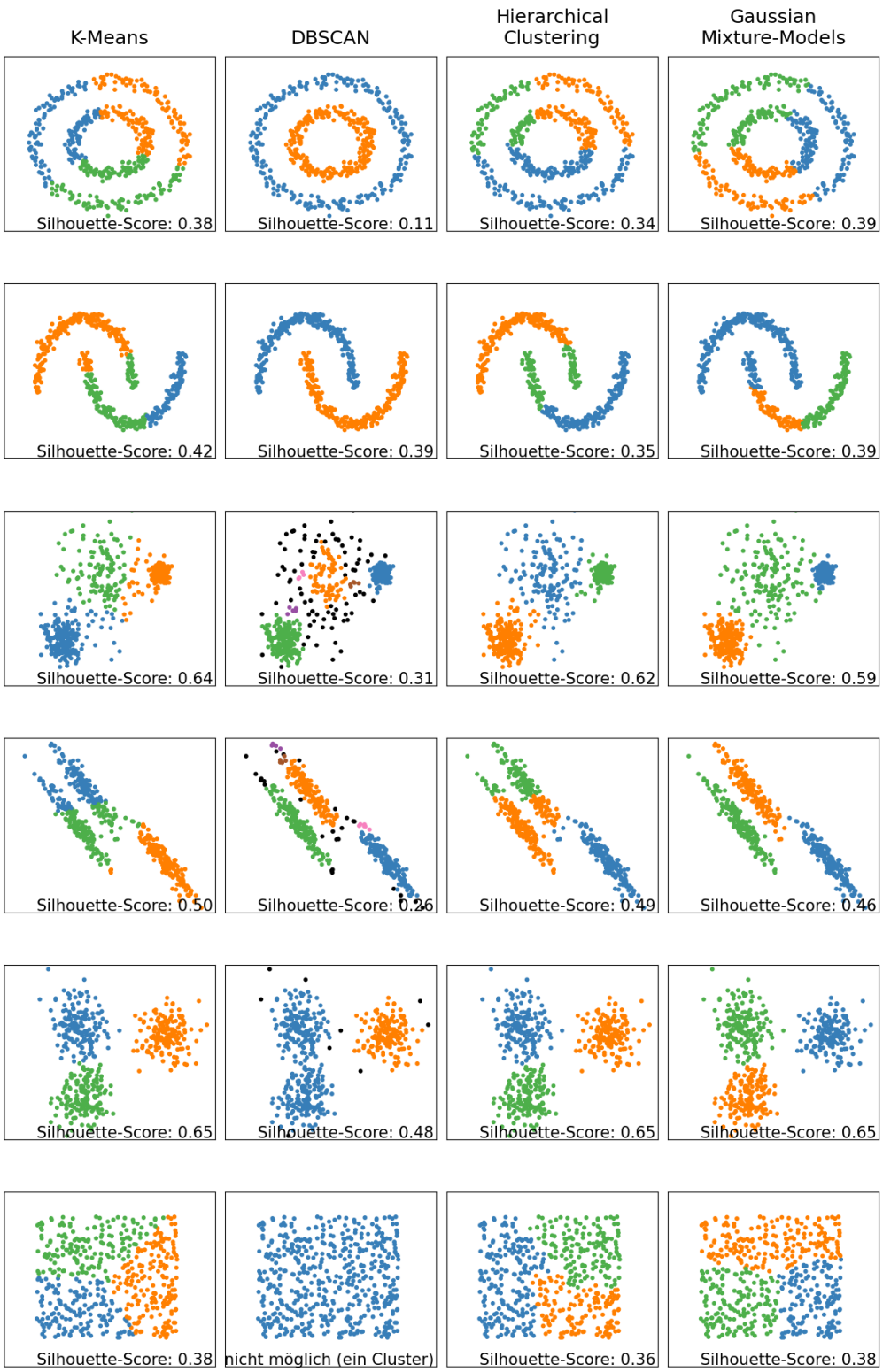


Abbildung 9: Vergleich verschiedener Algorithmen an verschiedenen Punkteverteilungen

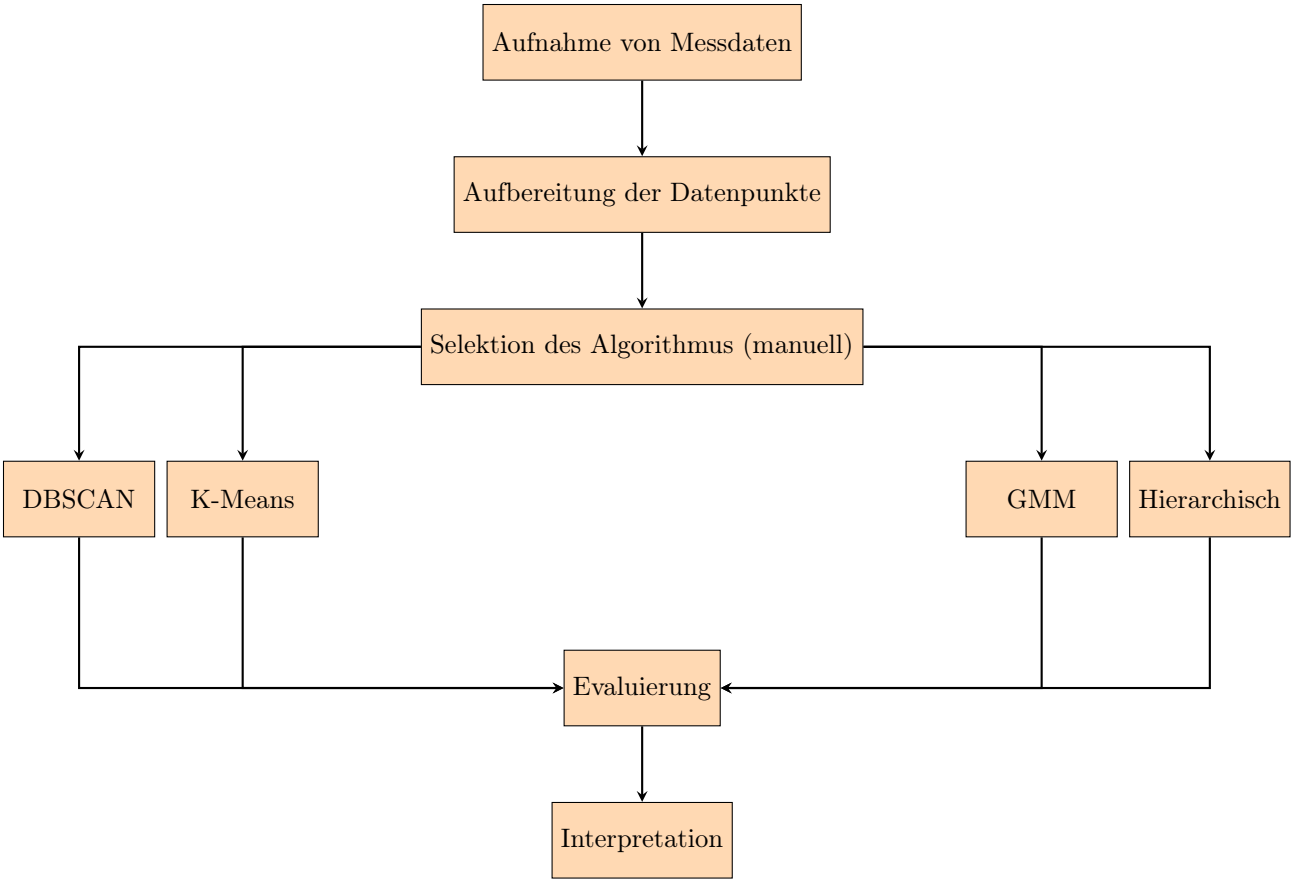


Abbildung 10: Flussdiagramm zur Veranschaulichung des Prozessablaufes bei der Clusteranalyse

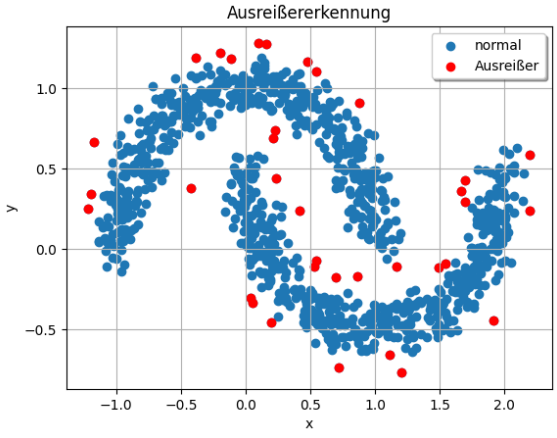
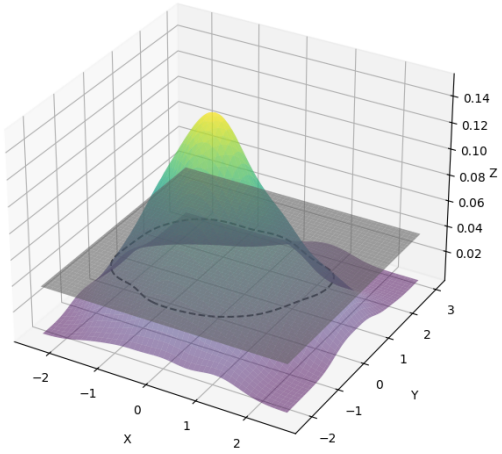
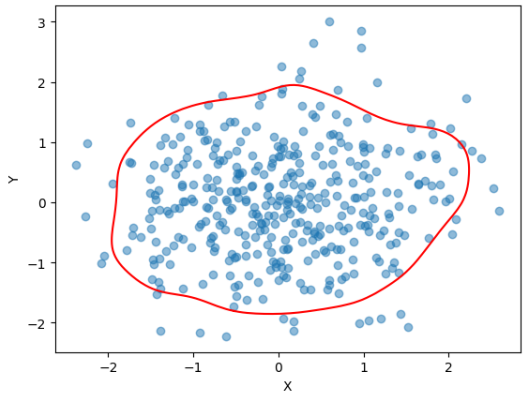


Abbildung 11: Finale Ausreißererkennung durch Kombination der Algorithmen

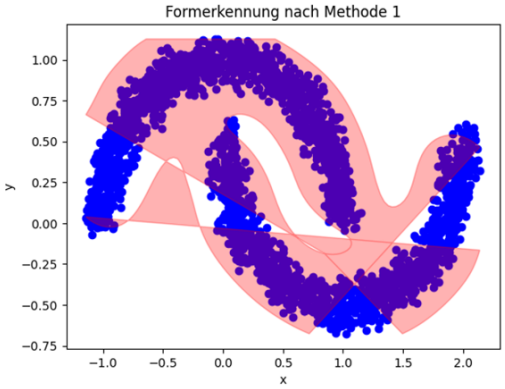


3D-Grafik der Dichteverteilung



2D-Grafik der resultierenden Kontur

Abbildung 12: Erkennung der Kontur nach Methode 1



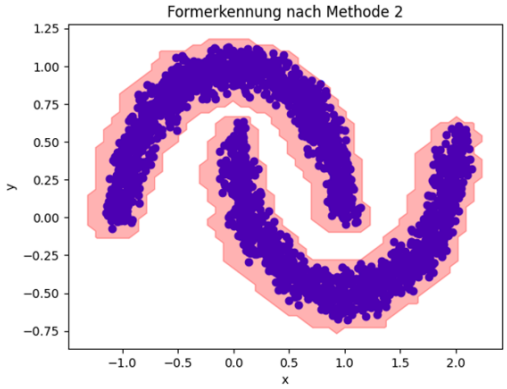


Abbildung 13: Vergleich beider Methoden zur Konturerkennung

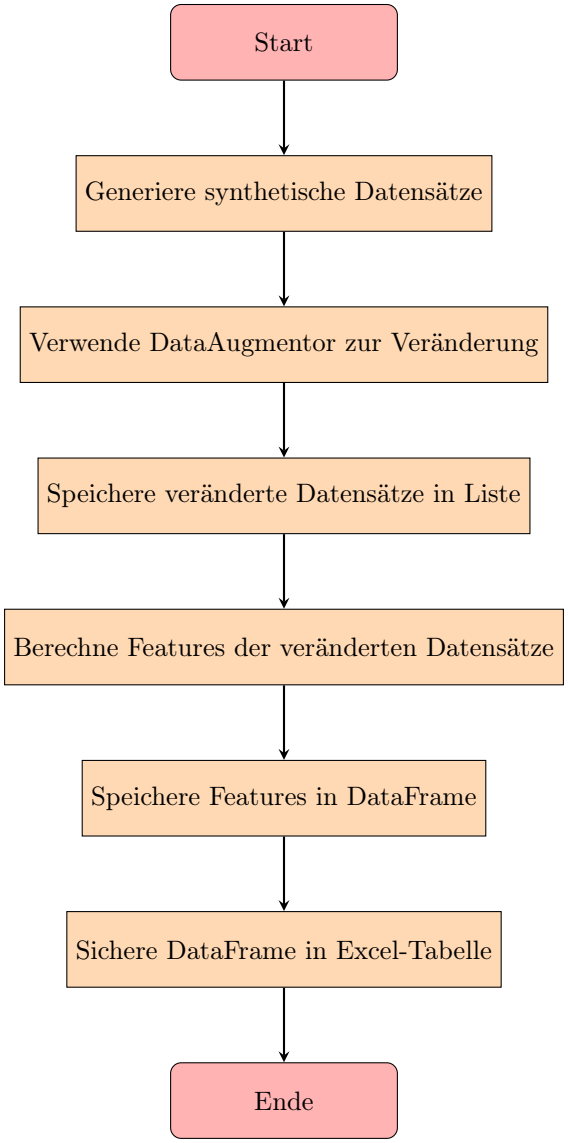


Abbildung 14: Flussdiagramm zur Generierung von Trainingsdaten

10 Literaturverzeichnis

Gedruckte Literatur

Raschka, Sebastian; Mirjalili, Vahid: *Python Machine Learning - Machine Learning and Deep Learning with Python, scikit-learn, and Tensorflow 2*, 3. Aufl., Birmingham (Vereinigtes Königreich), Packt, 2019

Zheng, Alice; Casari, Amanda: *Merkmalskonstruktion für Machine Learning - Prinzipien und Techniken der Datenaufbereitung*, 1. Aufl., Springfield (Vereinigte Staaten von Amerika), O'Reilly, 2019

Internetliteratur

Aunkofer, Benjamin: *Maschinelles Lernen: Klassifikation vs Regression*, https://data-science-blog.com/blog/2017/12/20/maschinelles-lernen-klassifikation-vs-regression/ [Zugriff am 11.07.2023]

Block, Tim: *What are the Strengths and Weaknesses of Hierarchical Clustering?*, https://www.displayr.com/strengths-weaknesses-hierarchical-clustering/ [Zugriff am 30.10.2023]

Brus, Patrick: *Clustering: How to Find Hyperparameters using Inertia*, https://towardsdatascience.com/clustering-how-to-find-hyperparameters-using-inertia-b0343c6fe819 [Zugriff am 30.12.2022]

Busch, Michael: *Analyse dichtebasierter Clusteralgorithmen am Beispiel von DBSCAN und MajorClust*, https://downloads.webis.de/theses/papers/busch_2005.pdf [Zugriff am 26.11.2023]

Chauhan, Nagesh Singh: *DBSCAN Clustering Algorithm in Machine Learning*, https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html [Zugriff am 30.10.2023]

Dadi, Harihara; Venkatesh, P.; Poornesh, P.; L., Narayana Rao; Kumar, N.: *Tracking Multiple Moving Objects Using Gaussian Mixture Model*, https://www.researchgate.net/publication/305709395_Tracking_Multiple_Moving_Objects_Using_Gaussian_Mixture_Model[Zugriff am 30.10.2023]

DataNovia (Hrsg.): *Cluster Validation Statistics: Must Know Methods*, https://www.datanovia.com/en/lessons/cluster-validation-statistics-must-know-methods [Zugriff am 30.10.2023]

Delua, Julianna: *Supervised vs. Unsupervised Learning: What's the Difference?*, https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning [Zugriff am 11.07.2023]

Edraw (Hrsg.): *Dendogramm*, https://www.edrawsoft.com/de/article/what-is-dendrogram.html [Zugriff am 29.12.2022]

Ellis, Christina: *When to use gaussian mixture models*, https://crunchingthedata.com/when-to-use-gaussian-mixture-models/ [Zugriff am 30.10.2023]

Ester, Martin; Kriegel, Hans-Peter; Sander, Jörg; Xu, Xiaowei: *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*, https://www.dbs.ifi.lmu.de/Publikationen/Papers/KDD-96.final.frame.pdf [Zugriff am 30.10.2023]

Fisher, Robert; Simon Perkins; Ashley Walker; Erik Wolfart: *Sobel Edge Detector*, https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm [Zugriff am 04.10.2023]

Foley, Daniel: *Gaussian Mixture Modelling (GMM) - Making Sense of Text Data using Unsupervised Learning*, https://towardsdatascience.com/gaussian-mixture-modelling-gmm-833c88587c7f [Zugriff am 30.10.2023]

Gaido, Marco: *Distributed Silhouette Algorithm: Evaluating Clustering on Big Data*, https://arxiv.org/pdf/2303.14102.pdf [Zugriff am 30.12.2022]

Gavali, Suvarna: *Skewness and Kurtosis: Quick Guide*, https://www.analyticsvidhya.com/blog/2021/05/shape-of-data-skewness-and-kurtosis/ [Zugriff am: 30.10.2023]

Google (Hrsg.): *k-Means Vor- und Nachteile*, https://developers.google.com/machine-learning/clustering/algorithm/advantages-disadvantages?hl=de [Zugriff am 30.10.2023]

Grellmann, Martin: *DBSCAN (Density-Based Spatial Clustering of Applications with Noise)*, https://martin-grellmann.de/dbscan-density-based-spatial-clustering-of-applications-with-noise [Zugriff am 27.05.2023]

Grosse, Roger; Srivastava, Nitish: https://www.cs.toronto.edu/rgrosse/csc321/mixture_models.pdf [Zugriff am 30.10.2023]

Gupta, Sakshi: *Regression vs. Classification in Machine Learning: What's the Difference?*, https://www.springboard.com/blog/data-science/regression-vs-classification [Zugriff am 29.12.2022]

IBM (Hrsg.): *Was ist überwachtes Lernen?* https://www.ibm.com/de-de/topics/supervised-learning [Zugriff am 11.07.2023]

Lazyprogrammer (Hrsg.): *Unsupervised Learning I - Gaussian Mixture Model (GMM)*, https://lazyprogrammer.me/mlcompendium/clustering/gmm.html [Zugriff am 30.10.2023]

Lippke, Steffen: *DBSCAN und BIRCH einfach erklärt und angewandt*, https://lippke.li/dbscan-und-birch/ [Zugriff am 27.05.2023]

Matzer, Michael: *Grundlagen Statistik & Algorithmen, Teil 7 - So deckt der Local Outlier Factor Anomalien auf*, https://www.bigdata-insider.de/so-deckt-der-local-outlier-factor-anomalien-auf-a-803652/ [Zugriff am 30.10.2023]

National Institute of Standards and Technology (Hrsg.): *Measures of Skewness and Kurtosis*, https://www.itl.nist.gov/div898/handbook/eda/section3/eda35b.htm [Zugriff am 06.10.2023]

Novustat (Hrsg.): *Regression Statistik*, https://novustat.com/statistik-glossar/regression-statistik.html [Zugriff am 11.07.2023]

Roux, Maurice: *A Comparative Study of Divisive and Agglomerative Hierarchical Clustering Algorithms*, https://hal.science/hal-02085844/document [Zugriff am 30.10.2023]

Saddique, Asif: *Introduction to ROOT*, https://indico.cern.ch/event/555909/contributions/2265949/attachments/1325123/1988911/Root_Lecture2.pdf [Zugriff am 04.10.2023]

Sefidian, Amir Masoud: *How to determine epsilon and MinPts parameters of DBSCAN clustering*, https://www.sefidian.com/2022/12/18/how-to-determine-epsilon-and-minpts-parameters-of-dbscan-clustering/ [Zugriff am 30.10.2023]

Seidl, Prof. Dr. Thomas: *Knowledge Discovery in Databases I (WS 2020/21) -Dichtebasiertes Clustering*, https://www.dbs.ifi.lmu.de/Lehre/KDD/SS14/skript/KDD-3-Clustering-2.pdf [Zugriff am 27.05.2023]

Sharma, Abhishek: *How to Master the Popular DBSCAN Clustering Algorithm for Machine Learning*, https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/ [Zugriff am 30.10.2023]

Singh, Aarti: *Gaussian Mixture Models: Lecture from the ML-Department of Carnegie Mellon University*, https://www.cs.cmu.edu/~aarti/Class/10315_Fall19/lecs/Lecture20.pdf [Zugriff am 30.10.2023]

Spodarev, Prof. Dr. Evgeny: *Dichteschätzer*, https://www.uni-ulm.de/fileadmin/website_uni_ulm/mawi.inst.110/lehre/ss13/Stochastik_I/Skript_4.pdf [Zugriff am 04.10.2023]

Stäudtner, Jürgen: *Sprüche und Zitate zur Digitalisierung, der digitalen Transformation und dem digitalen Wandel*, https://www.cridon.de/sprueche-zitate-digitalisierung/ [Zugriff am 25.11.2023]

Sultana, Shaik Irfana: *How the Hierarchical Clustering Algorithm Works*, https://dataaspirant.com/hierarchical-clustering-algorithm [Zugriff am 30.10.2023]

Technische Universität Dortmund (Hrsg.): *Limitations of k-Means Clustering*, https://dm.cs.tu-dortmund.de/en/mlbits/cluster-kmeans-limitations/ [Zugriff am 30.10.2023]

Weisstein, Eric W.: *Skewness*, https://mathworld.wolfram.com/Skewness.html [Zugriff am 06.10.2023]

11 Abbildungsverzeichnis

Alle Abbildungen wurden von den Autoren der Arbeit selbst mithilfe eigener Programme erstellt. Keine von den verwendeten Grafiken wurde aus dem Internet oder sonstiger Literatur entnommen.

Eidesstattliche Erklärung

Wir erklären hiermit an Eides statt, dass wir die vorliegende Arbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt haben; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Erfurt, 21.12.2023			
Ort, Datum	Hagen Jacob	Jan Edgar König	Robert Vetter