JOHANNES KEPLER
UNIVERSITÄT LINZ
Netzwerk für Forschung, Lehre und Praxis

EMAD - EM Clustering mit aggregierten Daten

Diplomarbeit zur Erlangung des akademischen Grades
Magister rer. soc. oec.
im Diplomstudium Wirtschaftsinformatik

angefertigt am
Institut für Wirtschaftsinformatik –
Data & Knowledge Engineering

Eingereicht von
Julia Messerklinger

Begutachter
o. Univ.-Prof. Dr. Michael Schrefl
Mitbetreuer
Dipl.-Wirtsch.-Inf. Dr. Mathias Goller

Linz, September 2006

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die Diplomarbeit mit dem Titel *"EMAD - EM Clustering mit aggregierten Daten"* selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und alle den benutzten Quellen wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Linz, den 18.9.2006

(Julia Messerklinger)

Zusammenfassung

Clustering ist eine Data-Mining Aufgabe, die auf Grund der vielen Rechenoperationen bei großen Datenmengen eine lange Laufzeit aufweist, sodass deren Anwendung im Prozess der Knowledge Discovery in Databases (KDD) nur bedingt effizient durchführbar ist. Der in dieser Arbeit angesprochene Ansatz des Vorausschauenden Data-Mining reduziert diese Problematik, indem die gesamten Daten zuerst anwendungsunabhängig mit Hilfe eines Clustering-Verfahrens aufbereitet werden. In einem zweiten Schritt verwendet ein beliebiges Data-Mining Verfahren die aufbereiteten Daten, um die konkrete Analyse durchzuführen. Auf Grund der Aufbereitung können mehrmals Analysen mit veränderten Parametern ausgeführt werden, wobei die Bestimmung der Endergebnisse schneller als mit nicht aggregierten Daten erfolgt.

Diese Arbeit stellt das Clustering-Verfahren EMAD (Erwartungsmaximierung mit aggregierten Daten) vor, dass für den zweiten Schritt des Vorausschauenden Data-Mining entwickelt wurde. Das Clustering-Verfahren Erwartungsmaximierung ist dabei für die Verwendung von aggregierten Daten angepasst worden. Untersuchungen von EMAD zeigten, dass dieses Verfahren bei großen Datenmengen eine gute Skalierbarkeit aufweist.

Abstract

Clustering is a data mining task that is computationally intensive and shows an increasing runtime in large databases, so that its application in the process of Knowledge Discovery in Databases (KDD) can hardly be done efficiently. This work discusses the approach of anticipatory clustering, which reduces this problem by an application-independent preparation of all data via a clustering method. In a second step any data mining method will then use the prepared data for a specific analysis. Because of the generic preparation analyses can be executed repeatedly with modified parameters where the determination of the results is faster than with non aggregated data.

This work introduces the clustering method EMAD (expectation maximization with aggregated data) that is developed for the second step in the anticipatory clustering. For this reason the clustering method expectation maximization has been adjusted to be applicable to aggregated data. Experimental results from EMAD confirm that the algorithm exhibits a good scalability with large databases.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

1 Einleitung

Die Motivation der Knowledge Discovery in Databases (KDD) ist die Extraktion von potentiellem Wissen aus Daten, die durch den Menschen auf Grund der Menge und Komplexität nicht mehr manuell analysierbar sind, um daraus Schlüsse für zukünftiges Handeln zu ziehen. So kann z.B. ein Einzelhändler mit Hilfe der Einkäufe seiner Kunden bestimmen, welcher Typ von Kunde welche Produkte kauft, und auf Grund dieser Informationen sein Sortiment anpassen, um seine Umsätze zu steigern. Eine ähnliche Verbesserung von Absatzmöglichkeiten bzw. eine Optimierung von Marketinginstrumenten ist beispielsweise auch für Unternehmen in der Telekommunikationsbranche denkbar.

Das Data-Mining stellt einen Schritt innerhalb des Prozesses der Knowledge Discovery in Databases, KDD-Prozess genannt, dar, in dem Verfahren zum Finden von Mustern in den Daten ausgeführt werden. Dabei weisen bestehende Verfahren bei großen Datenmengen eine lange Laufzeit auf. Eine Aufgabe des Data-Mining stellt das Clustering dar.

Ein möglicher Anwendungsbereich des Clusterings ist die Optimierung von Angeboten und Marketinginstrumenten in der Telekommunikationsbranche. Die unterschiedlichen bestehenden Angebote der Telekommunikationsunternehmen decken die Bedürfnisse der verschiedenen Zielgruppen, wie z.B. Jugendliche oder Geschäftskunden, ab. Der Wettbewerbsdruck ist bei diesen Unternehmen gestiegen, da die Kunden auf Grund günstigerer Angebote oder aktuellerer Produkte den Anbieter oftmals bei Ablauf des Bindungsvertrages wechseln, weshalb diese Kunden von den Unternehmen gezielt beworben werden.

Damit die Kunden, die vor dem Ablauf des Bindungsvertrages stehen, nicht mit allen Angeboten beworben werden, sondern nur Angebote erhalten, die ihren Bedürfnissen entsprechen, kann Direct Marketing eingesetzt werden. Durch dieses zielgruppenspezifisches Marketing können außerdem die Kosten für das Unternehmen reduziert werden. Auf Grund einer konkreten Analyse des Telefonierverhaltens der Kunden werden diese mit Hilfe des Data-Mining Algorithmus den Zielgruppen zugeordnet. Wegen der großen Menge an Informationen über das Telefonierverhalten, wie Anzahl der Anrufe oder gewählte Nummern, ist die Analyse der gesamten Kunden sehr zeitintensiv. Außerdem sind die Analysen für die Zielgruppen bei

neuen Angeboten des Unternehmens erneut durchzuführen, wie in Abbildung 1 dargestellt.

Die Informationen aus der Analyse des Telefonierverhaltens von bestehenden Kunden bilden außerdem die Grundlage für neue Angebote, um die bestehenden Kunden weiterhin zufrieden zu stellen und neue Kunden zu akquirieren.

Durch eine allgemeine statistische Aufbereitung der Informationen über das Telefonierverhalten der Kunden kann der Zeitaufwand für die Analysen im Vergleich zu nicht aufbereiteten Daten verkürzt werden, da die zu untersuchende Menge an Daten dadurch reduziert wurde. Diese Zusammenfassung ermöglicht nun ein interaktives Arbeiten mit dem in dieser Arbeit vorgestellten Clustering-Verfahren, das mit den aufbereiteten Daten die konkreten Analysen durchführt. Diese allgemeine Aufbereitung muss nicht für jede konkrete Analyse durchgeführt werden, sondern wird für mehrere unterschiedliche Analysen herangezogen, da nur die für die Analyse relevanten Daten untersucht werden. Das Vorgehen wird in Abbildung 2 skizziert.

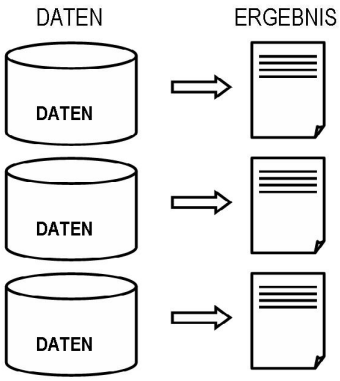

Abbildung 1: Allgemeines Vorgehen bei der Analyse

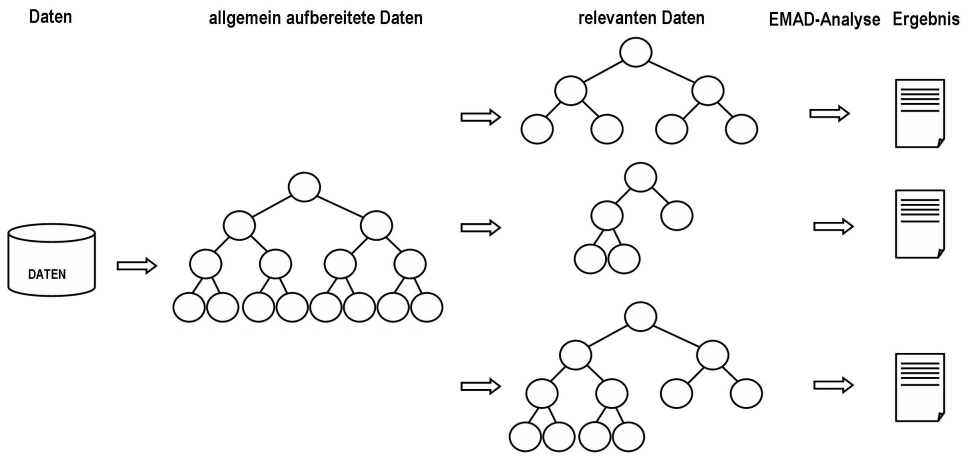

Abbildung 2: Vorgehen der Analyse mit dem in dieser Arbeit vorgestellten Clustering-Verfahren

In der vorliegenden Arbeit wird nun das Clustering-Verfahren Erwartungsmaximierung für die Anwendung von großen Datenmengen in solcher Weise angepasst, dass ein interaktives Arbeiten möglich wird. Durch das angepasste Verfahren EMAD, Erwartungsmaximierung mit aggregierten Daten, kann der Benutzer schneller zu einem Endergebnis kommen, und er hat die Möglichkeit, auf unbrauchbare Ergebnisse zu reagieren, indem er den Algorithmus mit veränderten Parametern erneut ausführt. Die Form der Ergebnisse ändert sich bei diesem Verfahren für den Benutzer nicht.

Diese Benutzerfreundlichkeit des Verfahrens ist auf eine Teilung des KDD-Prozesses zurückzuführen, indem der zeitaufwendige Teil, der die Aufbereitung der Daten umfasst, von jenem, der die Analyse beinhaltet, getrennt wird. Auf Grund der allgemeinen Aufbereitung, die jederzeit durchgeführt werden kann, ist der Benutzer in der Lage, die Analysen mehrmals mit unterschiedlichen Parametern durchzuführen, bis eine verwertbare Lösung gefunden wird. Da die Aufbereitung der Daten einmalig erfolgt, und sie dadurch komprimiert werden, kann die Laufzeit für die nachfolgenden Analysen reduziert werden.

Im folgenden Abschnitt werden der Themenbereich und die Motivation dieser Arbeit dargelegt. Danach geht Abschnitt 1.2 auf die Aufgabenstellung und die Ziele ein und in Abschnitt 1.3 wird der Aufbau dieser Arbeit kurz vorgestellt.

1.1 Gegenstand und Motivation

Das Data-Mining, das im Prozess der KDD ausgeführt wird, versucht mit Hilfe von Algorithmen Muster aus den aufbereiteten relevanten Daten zu extrahieren, um daraus zukünftige Handlungsschritte abzuleiten. Dabei nimmt die Laufzeit auch bei sehr gutem, d.h. linearem, Skalierverhalten mit großen Datenmengen derart zu, sodass die Laufzeit auf mehrere Stunden ansteigen kann, wie Fürst beim Clustern von fünf Millionen Datensätzen mit dem Clustering-Verfahren k-means getestet hat [Für04, Abschnitt 6.2.2].

Neben der Problematik der hohen Laufzeit bei großen Datenmengen kann sich in bestimmten Situationen, wie in Abschnitt 4.1 beschrieben, die Laufzeit noch zusätzlich verlängern. So hat der Benutzer auf Grund seines Hintergrundwissens die Parameter für den Algorithmus zu bestimmen, was die Verwertbarkeit des Ergebnisses entscheidend beeinflussen kann. Außerdem werden ähnliche Anwendungen isoliert ausgeführt, weshalb überlappende Datenmengen mehrfach gelesen und verarbeitet werden, wodurch es zu einer unnötig längeren Gesamtzeit kommt.

Die Idee, auf die diese Arbeit aufbaut, ist die einmalige generische Voranalyse der Daten, sodass die nachfolgenden konkreten Analysen die Zwischenergebnisse der generischen übernehmen, um dadurch schneller zu Endergebnissen zu kommen. Dies ist der Ansatz des Vorausschauenden Data-Mining, vergleiche Abschnitt 4.2, bei dem zuerst die gesamten Daten aufbereitet und Aggregationen von diesen bestimmt werden. Im nächsten Schritt werden dann die aggregierten Daten für die spezifische Anwendung bestimmt und das Data-Mining Verfahren wird darauf angewendet. Dieser zweite Schritt ist unabhängig vom ersten und kann beliebig oft mit unterschiedlichen Anwendungen auf die aggregierten Daten ausgeführt werden.

EMAD, Erwartungsmaximierung mit aggregierten Daten, wurde für den zweiten Schritt im Vorausschauenden Data-Mining entwickelt, indem das Clustering-Verfahren Erwartungsmaximierung (EM), vergleiche Abschnitt 2.2.3.5, für die Verwendung von aggregierten Daten angepasst wurde.

Der Ansatz des Vorausschauenden Data-Mining wird weitgehend durch den Algorithmus CHAD, Clustering Hierarchically Aggregated Data, umgesetzt, der mit Hilfe eines hierarchischen Clustering-Verfahrens die gesamten Daten aufbereitet, und in dem EMAD die anwendungsspezifischen Clustering-Modelle erstellt.

1.2 Aufgabenstellung und Zielsetzung

Aufgabenstellung dieser Diplomarbeit ist die Anpassung der Clustering-Methode EM und deren Umsetzung in einem Versuch, sodass diese in dem bereits bestehenden System CHAD, das auf der Kombination von Data-Mining Verfahren beruht, ausgeführt werden kann.

Gemäß der Aufgabenstellung wurde die angepasste Clustering-Methode EMAD als Prototyp implementiert, sodass dieser auf Grund des Ergebnisses eines hierarchischen Clustering-Verfahrens in CHAD ein Clustering basierend auf EM ausführt. Die Clustering-Verfahren werden dabei hintereinander auf die gleiche Datenbasis angewendet.

Mit Hilfe der Laufzeit und Qualität der Clustering-Modelle soll die Umsetzung der angepassten Clustering-Methode EMAD mit einer bereits bestehenden Umsetzung, die auf dem Clustering-Verfahren k-means basiert, verglichen werden.

Diese Arbeit erläutert die im Prototyp verwendeten Ansätze und begründet deren Auswahl. Weiters enthält sie die Ergebnisse der Umsetzung und analysiert die daraus erkennbaren Eigenschaften.

1.3 Aufbau der Arbeit

Nachdem in den vorangegangenen Abschnitten die Thematik dieser Arbeit ausgeführt wurde, gibt dieser Abschnitt einen kurzen Überblick über den inhaltlichen Aufbau der Arbeit.

In Kapitel 2 werden die relevanten Grundlagen von Knowledge Discovery in Databases und Data-Mining zusammengefasst, die für das Verständnis der anschließenden Kapitel notwendig sind, wobei im Besonderen auf das Clustering eingegangen wird.

Alternative Ansätze, die das gleiche Ziel wie EMAD, nämlich ein Clustering basierend auf EM bei großen Datenmengen, verfolgen, werden in Kapitel 3 beschrieben. Darin wird auch auf die Unterschiede zu EMAD eingegangen und erläutert, warum diese Ansätze nicht bei der Umsetzung des Prototyps berücksichtigt wurden.

In Kapitel 4 wird das Vorausschauende Data-Mining beschrieben, wobei zuerst auf die Gründe für diese Vorgehensweise eingegangen wird.

Das bestehende System CHAD wird in Kapitel 5 dargelegt, indem der Ansatz beschrieben und auf den Aufbau sowie auf die Verwendung der aggregierten Repräsentation der Daten eingegangen wird. Außerdem wird die verwendete Initialisierung von EMAD kurz vorgestellt.

Anschließend wird das Konzept und eine detaillierte Beschreibung des Algorithmus EMAD in Kapitel 6 vorgestellt, wobei im Besonderen auf die Anwendung der aggregierten Repräsentation der Daten im Algorithmus eingegangen wird.

In Kapitel 7 werden die Ergebnisse der Tests, die Laufzeit und die Qualität von EMAD, dargestellt und diskutiert. Das Kapitel 8 beinhaltet abschießende Betrachtungen zum Verfahren und stellt Empfehlungen für die Weiterentwicklung von EMAD vor.

2 Grundlagen

Die Grundlage für diese Arbeit stellt der Bereich Knowledge Discovery in Databases dar, der im folgenden Abschnitt ausführlich erläutert und dessen Prozess in Abschnitt 2.1.1 vorgestellt wird. Außerdem wird in Abschnitt 2.1.2 in ein Teilgebiet von Knowledge Discovery in Databases, das Data-Mining, eingeführt. Dabei werden neben einer Übersicht des Themas die Aufgaben des Data-Mining beschrieben.

In Abschnitt 2.2 wird auf das Clustering, das eine bestimmte Aufgabe des Data-Mining darstellt und Thema dieser Arbeit ist, eingegangen. Insbesondere werden in Abschnitt 2.2.1 die Ziele der Clustering-Methoden und in Abschnitt 2.2.2 der Begriff der Ähnlichkeit von Objekten erläutert. Neben einem generellen Überblick der Clustering-Methoden werden in Abschnitt 2.2.3 die in dieser Arbeit verwendeten Clustering-Methoden ausführlich beschrieben. Dazu zählen BIRCH, siehe Abschnitt 2.2.3.2, k-means, siehe Abschnitt 2.2.3.4, und Erwartungsmaximierung, siehe Abschnitt 2.2.3.5. Abschließend werden in Abschnitt 2.2.4 noch die Probleme bei der Wahl der geeigneten Clustering-Methode diskutiert.

2.1 Knowledge Discovery in Databases

Im Zeitalter der digitalen Informationen ist das Problem der Datenflut allgegenwärtig, was die Motivation für das Gebiet Knowledge Discovery in Databases, kurz KDD genannt, darstellt. Die Hardware und die Datenbanktechnologien ermöglichen eine effiziente und billige Speicherung der Daten sowie deren Zugriff. Allerdings kann man aus diesen gespeicherten Daten nicht sofort Wissen ableiten, denn sowohl die Anzahl der Daten als auch ihre Komplexität übersteigen die menschlichen Möglichkeiten für eine Analyse. KDD versucht deshalb das relevante Wissen mit Hilfe von computergestützten Techniken und Werkzeugen zu extrahieren.

KDD hat sich, und entwickelt sich noch immer, aus der Überschneidung von unterschiedlichen Forschungsgebieten. Zu den wichtigsten Disziplinen gehören nach Ester und Sander [ES00, Seite 1f]:

- Statistik: modellbasierte Inferenzen. Der Schwerpunkt liegt hier auf numerischen Daten. Eine gute Einführung der KDD aus Sicht der Statistik wird von [BH99] gegeben.

- Maschinelles Lernen: Suchverfahren. Der Schwerpunkt liegt auf symbolischen Daten. Die wichtigsten Verfahren des maschinellen Lernens, die oftmals auch relevant für KDD sind, werden bei [Mit97] dargestellt.

- Datenbanksysteme: Skalierbarkeit für große Datenmenge, neue Datentypen oder die Integration mit kommerziellen Datenbanksystemen. Bei [CHY96] ist eine gute Einführung in das Gebiet KDD aus Sicht der Datenbanksysteme zu finden.

2.1.1 Der KDD-Prozess

In diesem Abschnitt werden die einzelnen Schritte vorgestellt, aus denen der Prozess der Knowledge Discovery in Databases besteht.

Der KDD-Prozess wird nach Fayyad, Piatetsky-Shapiro und Smyth als nicht trivialer Prozess der Identifizierung von „gültigen, neuen, potentiell nützlichen und letztendlich verständlichen Mustern in Daten“ definiert [FPSS96b, Seite 29]. Dabei stellt ein Muster einen Ausdruck dar, der eine Teilmenge der Daten beschreibt. Der Prozess impliziert einen Ablauf, der in mehrere Schritte unterteilt ist. Die Suche nach Strukturen, Modellen und Muster stellt sich dabei als nicht trivial dar. Die gefundenen Muster sind dann gültig, wenn sie mit einer gewissen Sicherheit auf neue Daten anwendbar sind. Neben der Neuheit sollen die Muster für den Anwender oder die Aufgabe nützlich sein. Schließlich sind sie für den Menschen verständlich, was auch erst nach ein paar Nachbearbeitungsschritten zutreffen kann.

Der in Abbildung 3 skizzierte KDD-Prozess benötigt für den Ablauf menschlichen Input. Außerdem kann es zu mehreren Iterationen zwischen allen Schritten kommen, bevor der Prozess ein Endergebnis bestimmt.
Im Folgenden wird ein Überblick über die einzelnen Schritte des KDD-Prozesses nach Fayyad, Piatetsky-Shapiro und Smyth gegeben [FPSS96b]. Eine umfassendere Ausführung ist außerdem bei Frank und Witten [WF00] zu finden.

Selection: Bei der Selection wird ein Verständnis über die Anwendung und das bereits bekannte Wissen entwickelt. Zusätzlich werden die Ziele des KDD-

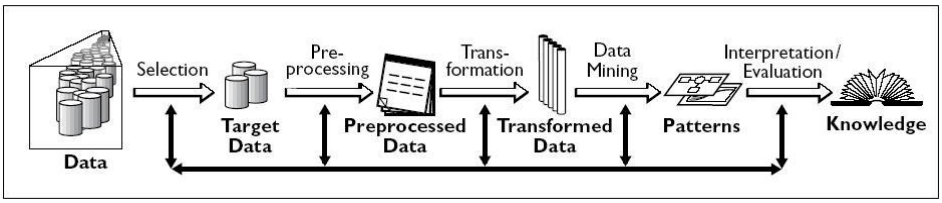

Abbildung 3: Überblick der Schritte im KDD-Prozess [FPSS96b, Seite 29]

Prozesses aus der Sicht der Benutzer definiert. Außerdem werden die Daten festgelegt, die im KDD-Prozess angewendet werden sollen.

Preprocessing: Ziel von Preprocessing ist es, die notwendigen Daten in konsistente Form zu bringen und zu vervollständigen. Werden unterschiedliche Quellen für die Daten verwendet, so sind diese noch zusätzlich in eine integrative Form zu bringen. Außerdem können die Daten noch verbessert werden, indem man das „Rauschen“ entfernt. Dieser Schritt kann einen erheblichen Aufwand im KDDProzess verursachen.

Transformation: Bei der Transformation werden nur jene Eigenschaften bzw. Variablen aus den Daten selektiert, die für die Ziele der Anwendung notwendig sind. Durch eine Reduktion der Dimensionalität oder Transformations-Methoden, wie der Attribut-Selektion, kann die Anzahl der Variablen verringert werden. Eine zu große Anzahl an Attributen kann nämlich die Effizienz und die Ergebnisse von Data-Mining Algorithmen negativ beeinflussen [WF00].

Es ist zu beachten, dass eine maximale Reduzierung der Dimensionalität bei minimalem Informationsverlust durch die Hauptkomponenten- oder Faktorenanalyse durchgeführt werden kann [KK00, Arm79]. Dabei werden ausgehend von den vorhandenen, korrelierenden Attributen neue, synthetische Merkmale generiert, die im höchsten Maße unkorreliert sind und ein Maximum an Information abbilden. Der Nachteil dieser Merkmale ist eine erschwerte Interpretierbarkeit. Insbesondere für methodisch unerfahrene Nutzer sind die Ergebnisse nicht anschaulich und schwer analysierbar, da ein direkter Zusammenhang zwischen Ergebnis und Datenbasis scheinbar nicht vorhanden ist.

Data-Mining: Bei Data-Mining wird die eigentliche Suche nach Mustern in den Daten mit der Hilfe von effizienten Algorithmen durchgeführt. Da dies den zentralen Teil im KDD-Prozess darstellt, wird er in Abschnitt 2.1.2 genauer erläutert.

Interpretation/Evaluation: Bei der Interpretation/Evaluation des KDD-Prozesses werden die gefundenen Muster visualisiert und interpretiert. Entsprechen diese Ergebnisse noch nicht den Zielen, so kann bei jedem Schritt des Prozesses eine weitere Iteration gestartet werden. Die endgültigen Ergebnisse werden im Anschluss dokumentiert und für nachfolgende KDD-Prozesse zu Verfügung gestellt.

2.1.2 Data-Mining

Nach Fayyad, Piatetsky-Shapiro und Smyth definiert sich Data-Mining als die Anwendung von speziellen Algorithmen, um damit Muster oder Modelle aus den zu untersuchenden Daten zu extrahieren [FPSS96a, Seite 39]. Es wird allerdings nicht selten mit Knowledge Discovery in Databases gleichgesetzt, obwohl Data-Mining nur einen Schritt des KDD-Prozesses darstellt [FPSS96b, Seite 28].

Die meisten Data-Mining Algorithmen bestehen nach [FPSS96b, Seite 31] letztendlich aus einem Mix von drei Komponenten, die auf Grund der Ziele einer Anwendung und der Daten für die Analyse zusammengestellt werden.

1. **Das Modell:** Das Modell besteht aus zwei Faktoren, einer Aufgabe des Modells, z.B. Clustering, und einer Art des Modells, z.b. Gauß'sche Wahrscheinlichkeitsdichte. Die Parameter, die das Modell genau spezifizieren, werden aus den zu analysierenden Daten gewonnen.

2. **Das Präferenzkriterium:** Das Präferenzkriterium stellt im Wesentlichen eine Funktion dar, die den Zusammenhang zwischen dem Modell und den Daten unterstützt, und somit Overfitting oder ein Modell mit zu vielen Freiheitsgraden unterbindet.

3. **Der Suchalgorithmus:** Der Algorithmus sucht schließlich in einer gegebenen Datenmenge ein passendes Modell oder eine passende Modellfamilie.

Im Folgenden werden die gebräuchlichsten Aufaben im Data-Mining nach Ester und Sander, die in dieser Arbeit auch als Verfahrensklassen bezeichnet werden, dargelegt [ES00, Seite 4f].

Klassifikation: Bei der Klassifikation sind die auftretenden Klassen in den Daten bereits bekannt, da diese durch Trainingsobjekte bestimmt wurden. Die einzelnen Datenobjekte werden diesen Klassen auf Grund der Attribute zugeordnet.

Clustering: Im Gegensatz zur Klassifikation stehen beim Clustering die Klassen noch nicht fest. Sie werden erst durch diese Methode aus den Daten bestimmt. Dabei wird nach Gruppierungen (Clustern) von Datenobjekten gesucht, die untereinander besonders ähnlich und gegenüber anderen Clustern besonders unähnlich sind. Diese Methode wird in Abschnitt 2.2 weiter ausgeführt.

Assoziationsregeln: Assoziationsregeln drücken häufig auftretende und starke Zusammenhänge auf Grund der Transaktionen aus einer Datenbank aus. Sie werden oft für die Warenkorbanalyse verwendet um interessante Transaktionsmuster aufzudecken, z.B. $A \wedge B \Rightarrow C$, wenn A und B dann folgt C.

Generalisierung: Bei der Generalisierung stehen nicht die detaillierten Daten im Vordergrund, sondern es wird versucht, eine kompaktere Beschreibung von diesen zu ermitteln. Dafür werden die Datensätze erheblich reduziert und die Attributwerte auf ein abstrakteres Niveau gesetzt.

Diese Arbeit beschäftigt sich ausschließlich mit Clustering, weshalb diese Verfahrensklasse im nächsten Abschnitt genauer beschrieben wird.

2.2 Clustering

Dieser Abschnitt führt in die Konzepte des Clusterings ein, das in vielen Gebieten verwendet wird. Neben Data-Mining [FPSSU96] wird es auch bei der statistischen Datenanalyse [KR89, BR93] sowie bei der Datenkompression [ZRL97] eingesetzt. Im Besonderen ist es im Data-Mining ein wichtiger Schritt, um Gruppierungen in den Daten zu finden.

In Abschnitt 2.2.1 und 2.2.2 werden zuerst die Grundbegriffe des Clusterings erläutert. Weiters wird in Abschnitt 2.2.3 ein Überblick der Clustering-Methoden gegeben und jene Methoden, die dieser Arbeit zugrunde liegen, werden ausführlich in den Abschnitten 2.2.3.2, 2.2.3.4 und 2.2.3.5 ausgeführt. Auf die Problematik bei der Auswahl der Clustering-Methode und deren Lösung wird im abschließenden Abschnitt 2.2.4 eingegangen.

2.2.1 Ziel von Clustering-Methoden

Das Ziel von Clustering-Methoden ist es, Daten (semi-)automatisch so in Gruppen bzw. Cluster einzuteilen, dass Objekte im gleichen Cluster möglichst ähnlich und Objekte aus unterschiedlichen Clustern möglichst unähnlich zueinander sind [KR90, Kapitel 1]. Die Einteilung der Cluster wird in dieser Arbeit auch als Clustering-Modell bezeichnet.

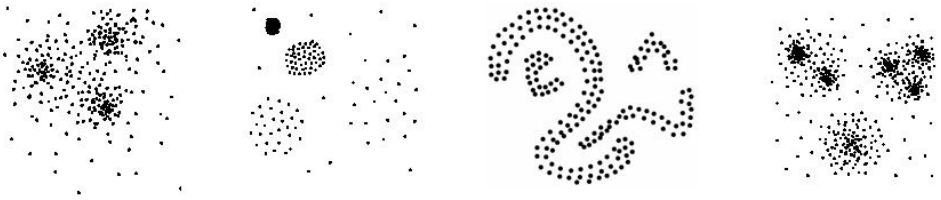

Abbildung 4: Beispiele für 2-dimensionale Clusterstrukturen mit verschiedenen Charakteristika [ES00, Seite 45]

Um die Clustering-Methode anwenden zu können, muss zuerst eine sinnvolle Modellierung für die Ähnlichkeit zwischen den Datenobjekten gefunden werden. Die einzelnen Clustering-Algorithmen sind dabei unterschiedlich gut für der Bestimmung der Cluster in den Daten anwendbar, da sich diese einerseits in Größe, Form und Dichte unterscheiden und andererseits ineinander verschachtelt sein können. Abbildung 4 veranschaulicht verschiedene Beispiele für 2-dimensionale Cluster mit den bereits genannten Eigenschaften. Das Merkmal für die Ähnlichkeit zwischen den Datenobjekten ist in diesen Beispielen der Abstand zwischen den Punkten.

2.2.2 Ähnlichkeit von Objekten

Angelehnt an Ester und Sander wird in diesem Abschnitt auf sie Bestimmung der Ähnlichkeit zwischen Objekten kurz eingegangen. Nähere Details dazu siehe [ES00, Abschnitt 3.1.2].

Die Ähnlichkeit von Objekten ist grundlegend für die Bestimmung der Cluster. Dabei wird oftmals eine Distanzfunktion *dist* herangezogen, welche die Distanz zwischen zwei Objekten basierend auf ihren direkten oder abgeleiteten Eigenschaften bestimmt. Die Distanz wird hier so interpretiert, dass die zwischen ähnlichen Objekten gering und zwischen unähnlichen Objekten hoch ist.

Die Wahl der Distanzfunktion wird stark von der Anwendung und den Objekten beeinflusst und bestimmt somit auch die Güte des Clustering-Modells. Es müssen jedoch grundsätzlich immer für alle Funktionen die folgenden Bedingungen für die Objekte $o_1, o_2, ..., o_n$ aus der Gesamtmenge der Objekte O gelten.

1. $dist(o_1, o_2) = d \in \mathbb{R}_+$;
   die Distanz zwischen zwei Objekte liegt im Wertebereich der positiven reellen Zahlen;

2. $dist(o_1, o_2) = 0 \Leftrightarrow o_1 = o_2$
   die Distanz zwischen zwei Objekten ist genau dann 0, wenn die beiden Objekte identisch sind;

3. $dist(o_1, o_2) = dist(o_2, o_1)$;
   die Distanz von Objekt o_1 zu Objekt o_2 entspricht der Distanz von Objekt o_2 zu Objekt o_1, es herrscht daher Symmetrie.

Ist zusätzlich noch die Dreiecksungleichung, siehe Abbildung 5, erfüllt, so gilt für alle $o_1, o_2, o_3 \in O$:

4. $dist(o_1, o_3) \leq dist(o_1, o_2) + dist(o_2, o_3)$;
   die Distanz zwischen den Objekten o_1 und o_3 ist nicht größer als die Summe der Distanzen von o_1 nach o_2 und o_2 nach o_3.

2.2.3 Überblick der Clustering-Methoden

Es werden in der Literatur eine große Menge von Clustering-Methoden angeführt, vergleiche dazu [ES00, Kapitel 3] und [HK01, Kapitel 8]. Im Folgenden wird eine

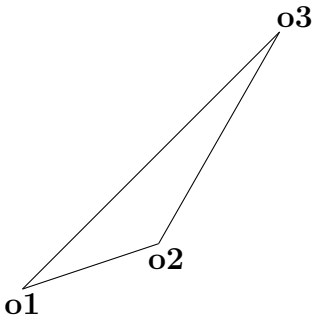

Abbildung 5: Dreiecksungleichung [ES00, Seite 46]

grundlegende Einteilung der Clustering-Methoden vorgenommen, und die in den anschließenden Kapiteln verwendeten Verfahren werden hier zugeordnet.

Die Clustering-Methoden können grundsätzlich in unterschiedliche Kategorien eingeteilt werden. Hier werden diese Methoden, entsprechend Ester und Sander [ES00, Kapitel 3], nur in hierarchische und partitionierende Verfahren unterteilt. Diese Einteilung begründet sich auf der Art der Cluster, die die Algorithmen in den Daten finden. Hierarchische Verfahren, vergleiche dazu Abschnitt 2.2.3.1, erzeugen eine Repräsentation der Daten, aus der man die Clusterstruktur ablesen kann. Die partitionierenden Verfahren hingegen zerlegen die Daten in unterschiedliche Cluster, näheres siehe Abschnitt 2.2.3.3.

Die Bestimmung der geeigneten Clustering-Methode hängt zum einen von den Daten und zum anderen von der Anwendung ab. Um die Problematik der Wahl der Clustering-Methode zu verringern, werden in Abschnitt 2.2.4 mögliche Vergleichskriterien vorgestellt.

2.2.3.1 Hierarchische Clustering-Verfahren

Bei hierarchischen Clustering-Verfahren wird eine hierarchische Repräsentation der Daten erzeugt, aus der man die Clusterstruktur ermitteln kann. Sie lassen sich dabei in die sogenannten agglomerativen und divisiven Verfahren unterteilen, die die Repräsentation der Daten entweder bottom-up oder top-down erstellen.

Die agglomerativen Verfahren gehen davon aus, dass zuerst jeder Punkt einen eigenen Cluster repräsentiert. Es werden anschließend immer zwei ähnliche Cluster

zusammengefügt, bis am Ende nur mehr ein einziger Cluster existiert oder eine Abbruchbedingung erfüllt ist. Es wird daher bottom-up vorgegangen.

Bei den divisiven Verfahren startet man mit einem einzigen Cluster, in dem alle Punkte enthalten sind. Im Anschluss wird immer ein Cluster gesplittet, bis jeder Cluster nur aus einem Punkt besteht oder eine Abbruchbedingung eintritt. Die Vorgangsweise ist hier top-down. Viele hierarchischen Verfahren arbeiten agglomerativ, da bei den divisiven Verfahren festgelegt werden muss, wie ein Cluster gesplittet werden soll.

Als Beispiel für ein hierarchisches Verfahren wird in Abschnitt 2.2.3.2 der Algorithmus BIRCH vorgestellt, da dieser vom Algorithmus CHAD, in dem EMAD eingebunden ist und der in Kapitel 5 genau beschrieben ist, herangezogen wird.

Ein sogenanntes Dendrogramm skizziert die hierarchische Repräsentation der Daten, aus der man die Clusterstruktur ermitteln kann. Dies ist nach Ester und Sander ein Baum, der die hierarchische Zerlegung der Datenmenge O in immer kleinere Teilmengen darstellt [ES00, Abschnitt 3.3]. Die Wurzel stellt einen einzigen Cluster dar, der die gesamte Datenmenge O beinhaltet. Die Blätter des Baumes repräsentieren Cluster, in denen sich ein einzelnes Objekt oder ein Aggregat aus mehreren Objekten der Daten befindet. Ein innerer Knoten repräsentiert die Vereinigung all seiner Kindknoten. Jede Kante zwischen einem Knoten und einem seiner Kindknoten beinhaltet noch die Distanz zwischen den beiden repräsentierten Datenmengen.

Eine graphische Darstellung des Dendrogramms, wie in Abbildung 6 dargestellt, ist üblich.

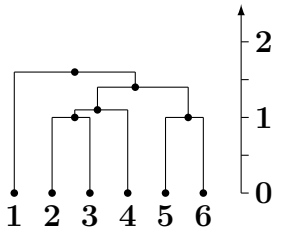

Abbildung 6: Dendrogramm

Im Dendrogramm repräsentiert jeder Knoten einen möglichen Cluster von der Datenmenge O. Für ein konkretes Clustering, also eine Zerlegung in mehrere Cluster, werden dann die notwendigen Knoten ausgewählt. Diese Auswahl kann durch manuelle Betrachtung und Analyse des Dendrogramms erfolgen [ES00, Seite 78]. Die Knoten können dabei auch von unterschiedlichen Ebenen des Baumes stammen.

Ein Dendrogramm kann für sehr große Datenmengen schnell unübersichtlich und schwer interpretierbar werden, was auch zur Motivation dieser Arbeit beiträgt.

2.2.3.2 BIRCH

Im diesem Abschnitt wird der Algorithmus BIRCH, der von Tian Zhang, Raghu Ramakrishnan und Mron Livny stammt, näher beschrieben [ZRL97]. BIRCH, Balanced Iterative Reducing and Clustering using Hierarchies, ist besonders für das Clustering von großen Datenmengen geeignet. Dieses Verfahren zählt zu den hierarchischen Verfahren beim Clustering und clustert inkrementell und dynamisch multidimensionale metrische Datenpunkte. Dabei ist für ein gutes Clustering nur ein einziger Scan der Datenpunkte notwendig. Die Qualität des Clustering-Modells wird von den verfügbaren Ressourcen, wie Zeit und Speicherplatz, beeinflusst.

Die Ausgangsbasis für dieses Clustering ist eine große Anzahl von multidimensionalen Datenpunkten, die nicht gleichmäßig verteilt sind. BIRCH bildet zunächst Beschreibungen, Clustering-Feature-Vektoren, dieser Datenpunkte. Die Clustering-Feature-Vektoren erlauben die Berechnung von allgemeinen Informationen über die Cluster, wie Centroid oder Radius, vergleiche dazu Abschnitt 2.2.3.2.1. Der benötigte Speicherplatz für diese ist gegenüber den gesamten Datenpunkten geringer.

Die Clustering-Feature-Vektoren werden anschließend in einem höhenbalancierten Baum (CF-Baum) abgespeichert, der nach Beendigung von BIRCH eine erweiterbare, im Bezug auf Ausreißer unsensibel und gegenüber den verfügbaren Ressourcen die bestmögliche Beschreibung der gesamten Datenpunkte darstellt [ZRL96].

Im folgenden Abschnitt werden die Clustering-Feature-Vektoren definiert. In Abschnitt 2.2.3.2.2 wird die Grundlage für das Rechnen mit Clustering-Feature-Vektoren dargelegt. Der CF-Baum wird in Abschnitt 2.2.3.2.3 und der genaue Ablauf von BIRCH wird in Abschnitt 2.2.3.2.4 erläutert.

2.2.3.2.1 Clustering-Feature-Vektor

Der Clustering-Feature-Vektor ist neben dem CF-Baum die Basis für die inkrementelle Berechnung und Wartung der Clustering-Informationen durch BIRCH.

Nach Zhang, Ramarkishnan und Lavny ist ein Clustering-Feature-Vektor ein Tripel, das die zusammengefassten Informationen eines Clusters enthält [ZRL96]. Es sind dabei N d-dimensionale Datenpunkte in einem Cluster C, $\{\vec{x_i}\}$, wobei $i = 1, 2, ..., N$, gegeben. Das Clustering-Feature-Vektor Tripel dieses Clusters C definiert sich daher als

$$CF = (N, \vec{LS}, SS). \tag{1}$$

Die Anzahl der Datenpunkte N im Cluster C ist

$$N = |C|. \tag{2}$$

Die lineare Summe $\vec{LS}$ der N Datenpunkte bestimmt sich durch

$$\vec{LS} = \sum_{i=1}^{N} \vec{x_i} \ . \tag{3}$$

Die Summe der Quadrate SS der N Datenpunkte ist daher

$$SS = \sum_{i=1}^{N} \vec{x_i}^2 \ . \tag{4}$$

Mit den Clustering-Feature-Vektoren können wichtige Informationen über einen Cluster bestimmt werden, wie Centroid bzw. Mittelpunkt $\vec{\mu_c}$, Radius R und Durchmesser Ø, die im Folgenden dargestellt sind.

$$\vec{\mu_c} = \frac{\sum_{i=1}^{N} \vec{x_i}}{N} = \frac{\vec{LS}}{N} \tag{5}$$

$$R = \sqrt{\frac{\sum_{i=1}^{N} (\vec{x_i} - \vec{\mu_c})^2}{N}} = \frac{SS}{N} - \vec{\mu_c}^2 \tag{6}$$

$$\emptyset = \sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{N}(\vec{x_i} - \vec{x_j})^2}{N(N-1)}} \tag{7}$$

2.2.3.2.2 Additivitätstheorem für Clustering-Feature-Vektoren

Die Möglichkeit die Clustering-Feature-Vektoren in einen höhenbalancierten Baum einzuordnen beruht auf dem Additivitätstheorem nach Tian Zhang [Zha96]. Dabei können Clustering-Feature-Vektoren, die ein oder mehrere Tupel repräsentieren, addiert oder subtrahiert werden. Dies ist gleichbedeutend mit einer Vereinigung oder Trennung der eigentlichen Tupel.

Werden zwei Clustering-Feature-Vektoren des CF-Baums $CF_1 = (N_1, \vec{LS_1}, SS_1)$ und $CF_2 = (N_2, \vec{LS_2}, SS_2)$, die von zwei unterschiedlichen Clustern stammen, addiert, so ergibt sich jener Clustering-Feature-Vektor, der beide Cluster vereinigt. Es gilt daher, dass

$$CF_1 + CF_2 = (N_1 + N_2, \vec{LS_1} + \vec{LS_2}, SS_1 + SS_2). \tag{8}$$

2.2.3.2.3 CF-Baum

In diesem Abschnitt werden die Eigenschaften des CF-Baums erläutert und es wird eine Beschreibung des Vorgangs beim Hinzufügen von neuen Datenpunkten gegeben.

Der CF-Baum nach Zhang, Ramarkishnan und Lavny ist ein höhenbalancierter Baum, der sich durch folgende Parameter bestimmen lässt [ZRL97, Abschnitt 4.2].

Branching Factor B:

Der Verzweigungsgrad B gibt die maximal mögliche Anzahl an Einträgen für Kindknoten in einem inneren Knoten an. Die Form dieser Einträge ist $[CF_i, child_i]$ mit $(i = 1, 2, ..., B)$, wobei $child_i$ einen Zeiger zu dem i-ten Kindknoten darstellt und CF_i der Clustering-Feature-Vektor des Sub-Clusters ist, der den i-ten Kindknoten repräsentiert. Ein innerer Knoten stellt einen Cluster dar, der aus allen Sub-Clustern bzw. Cluster-Feature-Vektoren besteht, die in diesem Knoten eingetragen sind.

Threshold T:

Der Schwellenwert T bestimmt die Größe des Baumes. Alle Einträge in den Blattknoten müssen diesen Schwellenwert erfüllen, sodass der Durchmesser bzw. der Radius des Clusters, der durch die Einträge repräsentiert wird, den Wert T nicht überschreitet. Je größer der Schwellenwert T ist, desto kleiner wird der CF-Baum.

Leaf Entries L:

Der Wert L bestimmt die maximale Anzahl an Einträgen von Clustering-Feature-Vektoren in den Blattknoten, $[CF_i]$ mit$(i = 1, 2, ..., L)$. Die Blattknoten haben neben den Clustering-Feature-Vektoren zusätzlich zwei Zeiger (prev und next), die alle Blattknoten für einen effizienten Scan miteinander doppelt verketten. Ein Blattknoten repräsentiert ebenfalls einen Cluster, der aus allen Sub-Clustern bzw. Cluster-Feature-Vektoren besteht, die in diesem Knoten eingetragen sind.

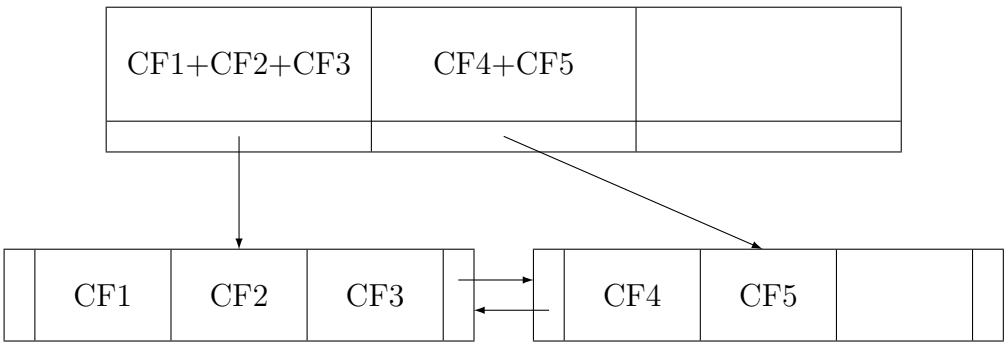

Abbildung 7: Überblick eines CF-Baums [ZRL96, Abbildung 1]

Der CF-Baum wird dynamisch durch das Einfügen eines Datenpunktes x aufgebaut. Dafür wird zuerst x in einen Clustering-Feature-Vektor $CF_x = (1, x, x^2)$ umgewandelt. Das Einfügen des CF_x in den CF-Baum, siehe Abbildung 7, erfolgt dann genauso wie bei einem B^+-Baum [ES00]. Die so entstehende Baumstruktur wird in dieser Arbeit auch als Dendrogramm bezeichnet.

Der CF-Baum ist eine sehr kompakte Repräsentation der Datensätze, da in den Blattknoten nicht nur einzelne Datenpunkte, sondern auch Sub-Cluster mit mehreren Datenpunkten enthalten sein können. Dies ist dann der Fall, wenn die gesamte

Datenmenge den zur Verfügung stehenden Hauptspeicher übersteigt. Die Blattknoten enthalten daher die kleinste mögliche Repräsentation der Datenpunkte, die mit den verfügbaren Ressourcen möglich ist.

Der Algorithmus zum Einfügen eines Clustering-Feature-Vektors CF_x, vergleiche dazu [ZRL97, Abschnitt 4.3], der einen einzelnen Datenpunkt oder einen Sub-Cluster repräsentiert, wird im Anschluss erläutert. Dabei sind die folgenden Schritte so abgestimmt, dass die Sensitivität des CF-Baums in Bezug auf die Reihenfolge der einzufügenden Datenpunkte so weit wie möglich gesenkt wird.

1. **Identifizieren des geeigneten Blattes:**
   Ausgehend von der Wurzel wird rekursiv im CF-Baum bis zu einem Blatt abgestiegen, indem jener Kindknoten ausgewählt wird, der gemäß der Distanzfunktion am nächsten beim CF_x liegt.

2. **Modifizieren des Blattes:**
   Im Blatt wird der nächstgelegene Eintrag E bestimmt und geprüft, ob dieser den Punkt aufnehmen kann, ohne die Schwellenwertbedingung zu verletzen. Falls dies zutrifft, wird der Eintrag E auf Grund des Additionstheorems zu $E := E + CF_x$ geändert. Falls dies nicht zutrifft, wird der Clustering-Feature-Vektor CF_x als neuer Eintrag in den Blattknoten eingefügt. Dabei kann es zu einem Überlauf kommen, wenn die maximale Anzahl L von Einträgen in einem Blattknoten überschritten wird. Dies macht den Splitt eines Blattknotens notwendig.

   Ein Knoten wird gesplittet, indem die am weitesten auseinanderliegenden Einträge als Basis für zwei neue Knoten bestimmt werden. Die restlichen Einträge werden jenem dieser Knoten zugeordnet, dem sie am nächsten liegen.

3. **Modifizieren des Pfades vom Blatt zur Wurzel:**
   Nach dem Einfügen des Clustering-Feature-Vektors CF_x in einen Blattknoten müssen all jene Clustering-Feature-Vektoren der inneren Knoten aktualisiert werden, die auf dem Pfad zwischen der Wurzel und dem Blattknoten liegen. Wenn kein Splitt beim Blattknoten durchgeführt wurde, werden dafür die inneren Knoten mit dem Clustering-Feature-Vektor CF_x addiert.

War allerdings ein Splitt des Blattknotens notwendig, wird der alte Eintrag im Vaterknoten durch zwei neue Einträge ersetzt, die zwei neue Blattknoten darstellen. Entsteht dabei kein Überlauf im Vaterknoten, der Verzweigungsgrad B wird also nicht überschritten, so werden alle Einträge auf dem Pfad zur Wurzel durch die Addition des Clustering-Feature-Vektors CF_x aktualisiert.

Im Falle eines Überlaufs muss der Vaterknoten geteilt werden, wobei genauso wie beim Splitt eines Blattknotens vorgegangen wird. Dieser Splitt kann sich bis zur Wurzel des CF-Baums fortsetzen, was zum Wachsen der Höhe des Baums um eins führt.

4. **Verfeinerung durch Verschmelzen:**
   Splitts werden durch die Blattgröße notwendig und sind von den Eigenschaften der Daten absolut unabhängig. Eine willkürliche Reihenfolge der Daten kann dadurch die Clustering-Qualität und die Speicherausnutzung beeinflussen. Eine zusätzliche Verschmelzung hilft diese Probleme abzumildern.

   Geht man davon aus, dass ein Blattknoten geteilt wurde und die Ausbreitung dieses Splitts beim inneren Knoten N_j endet, da dieser mit dem zusätzlichen Eintrag keinen Überlauf erzeugt hat, so wird bei der Verschmelzung der Knoten N_j untersucht und jene beiden Einträge daraus ermittelt, die am nächsten zueinander liegen. Wenn dies nicht das Paar ist, das durch den Splitt entstanden ist, wird versucht, sie und damit auch ihre Kindknoten zu verschmelzen. Durch diesen Schritt werden ein Knoten und ein Eintrag im Knoten N_j eingespart. Kommt es bei der Verschmelzung der Kindknoten jedoch zu einem Überlauf, werden diese wieder getrennt. Dabei wird allerdings die Verteilung der Einträge in beiden Kindknoten verbessert.

Ein CF-Baum kann nach Zhang, Ramarkishnan und Lavny mit einem Aufwand von $O(n)$ aufgebaut werden, da nur ein einziger Scan der Daten notwendig ist [ZRL97]. Der Parameter T bestimmt dabei vor allem die Größe des Baums. Größere Werte für T ergeben eher kleinere CF-Bäume. Ein CF-Baum CF_1 kann dadurch auch verkleinert werden, indem ein bestehender CF-Baum CF_2 in CF_1 eingefügt wird, wobei in CF_1 der Schwellenwert T gegenüber CF_2 höher ist [ES00, Seite 96].

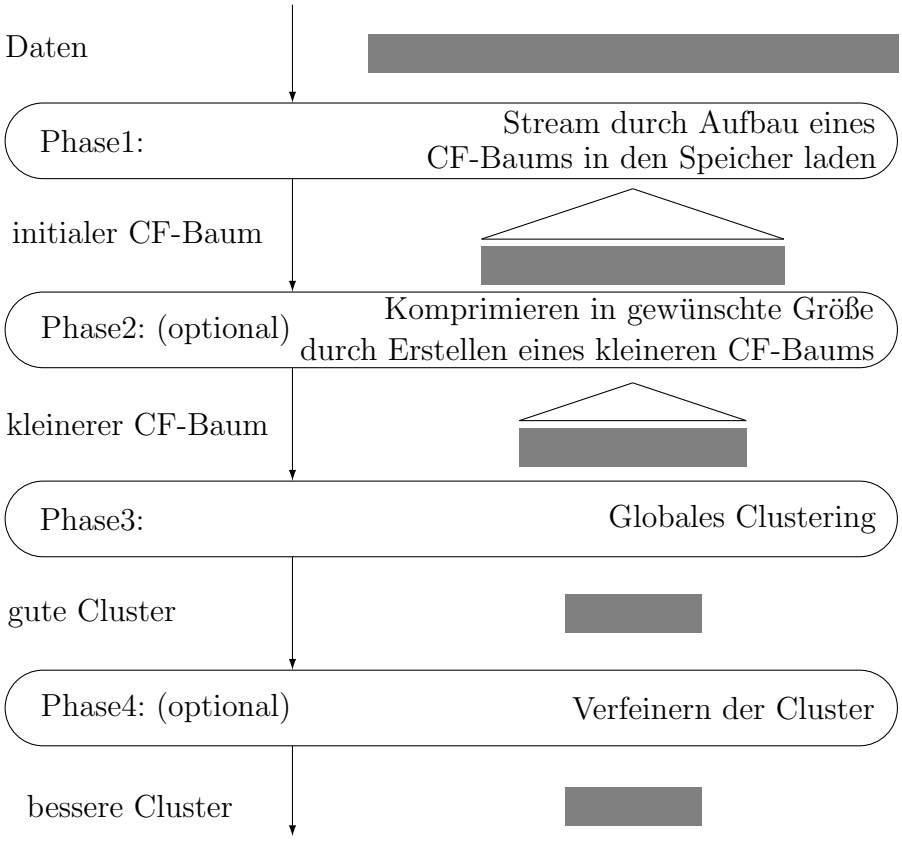

Abbildung 8: Übersicht der BIRCH Phasen [ZRL96, Abbildung 2]

2.2.3.2.4 BIRCH - Der Clustering-Algorithmus

In Abbildung 8 wird der Ablauf des Clustering-Algorithmus von BIRCH, bestehend aus vier Phasen, dargestellt und gemäß [ZRL97, Abschnitt 4.3] näher erläutert.

Die Hauptaufgabe von Phase 1 ist das Einlesen aller Daten und das Erstellen eines CF-Baums. Dieser Baum soll dabei die kleinste Granularität der Clustering-Informationen der Daten enthalten, die auf Grund der Ressourcen möglich ist.

Dafür beginnt BIRCH mit dem Einlesen der Daten und dem Aufbau eines CF-Baums, der einen initialen Schwellenwert T hat. Ist der Speicher für die gesamten Daten nicht ausreichend, so werden die bereits entstandenen Blattknoten des CF-Baums in einen kleineren CF-Baum übertragen, bei dem der Schwellenwert T höher

ist. Anschließend wird mit dem Einlesen der Daten fortgefahren. Das Erhöhen des Schwellenwerts wird so oft wiederholt, bis alle Daten eingelesen sind.

Der Schwellenwert T beeinflusst entscheidend die Granularität des CF-Baums. Bei der Initialisierung wird standardmäßig für den Schwellenwert der Wert 0 angenommen. Der Benutzer kann diesen Wert jedoch explizit festlegen. Um oftmaliges steigern des Schwellenwertes zu vermeiden, kann der Schwellenwert nicht um eins, sondern um einen größeren Wert, der auf einer Heuristik basiert, erhöht oder bereits mit einem gut gewählten Schwellenwert T begonnen werden. Ein zu hoher festgelegter initialer Wert bewirkt jedoch einen CF-Baum mit einer geringeren Granularität, als dies durch den Speicherplatz möglich wäre.

Zusätzlich kann Speicher für die Ausreißer reserviert werden. Beim Übertragen der Blattknoten in einen kleineren CF-Baum werden die potenziellen Ausreißer, also jene Einträge der Blätter, die viel weniger Datenpunkte als der Durchschnitt enthalten, gesondert abgespeichert. Auf Grund des veränderten Schwellenwertes oder einer Veränderung in der Verteilung durch nachträglich gelesene Daten kann ein Ausreißer seinen Status verlieren. Darum wird periodisch versucht, diese Ausreißer wieder in den Baum aufzunehmen, ohne diesen in seiner Größe zu verändern. Nachdem alle Punkte eingelesen wurden, wird noch einmal überprüft, ob sich nur Ausreißer im reservierten Speicher befinden. Diese werden im Anschluss gelöscht.

Auf Grund von Phase 1 sind die darauf folgenden Phasen

1. schnell, da (a) keine I/O Operationen nötig sind, und (b) das Problem des Clustering der Originaldaten zu einem kleineren Problem, dem Clustering der Blattknoten, reduziert wurde;

2. genau, da (a) viele Ausreißer eliminiert wurden, und (b) die verbliebenen Daten in ihrer größtmöglichen Granularität, die auf Grund des Speicherplatzes möglich ist, wiedergegeben werden;

3. weniger von der Reihenfolge der Daten abhängig, da die Einträge in den Blattknoten auf Grund der Position der Daten und nicht nach der eingelesenen Reihenfolge geordnet sind.

Die Phase 2 ist optional und abhängig von den Methoden, die in Phase 3 verwendet werden, da diese unterschiedliche Datenmengen für ein effizientes Clustering

brauchen. Deswegen wird in Phase 2 der CF-Baum aus Phase 1 in einer kleineren Form aufgebaut.

In Phase 3 werden die Blattknoten mit einem Clustering-Algorithmus geclustert. Dabei können unterschiedliche Algorithmen aus der Literatur verwendet werden, nachdem diese so angepasst wurden, dass sie die notwendigen Informationen für das Clustering aus den Clustering-Feature-Vektoren bestimmen können.

Nach der dritten Phase erhält man die Cluster der Daten, die noch geringfügige örtliche Ungenauigkeiten enthalten können, da zum einen nur eine grobe Zusammenfassung der Daten geclustert wurde und zum anderen die Gefahr von Anomalien besteht. Zu den Anomalien zählt die Möglichkeit, dass gleiche Datenpunkte auf Grund der Reihenfolge des Einlesens der Daten in unterschiedliche Blattknoten eingeteilt werden. In Phase 4, die optional ist, wird versucht diese Ungenauigkeiten z.B. mit Hilfe des Lloyd-kind Algorithmus nach [GG92] zu korrigieren. Dafür ist allerdings ein neuerliches Überprüfen der gesamten Daten notwendig.

2.2.3.3 Partitionierende Clustering-Verfahren

Die partitionierenden Clustering-Verfahren verteilen die Daten in mehrere Cluster, wobei in jedem Cluster mindestens ein Objekt enthalten ist und jedes Objekt nur einem Cluster zugeordnet wird. Das Ergebnis dieses Verfahrens wird iterativ ermittelt. Außerdem muss eine Initialisierung vorgegeben werden. Zu Vertretern dieses Verfahrens zählen k-means und die Erwartungsmaximierung, die bei der Umsetzung von EMAD verwendet wurde.

2.2.3.4 k-means

Beim k-means Verfahren nach MacQueen [Mac67] wird jeder Cluster durch einen zentralen Punkt, Centroid oder Mittelpunkt μ_c, repräsentiert. Die Zuordnung der Punkte erfolgt dann zu jenen Clustern, deren Mittelpunkte am ähnlichsten sind. Dafür wird die Distanz zwischen den Punkten und allen Mittelpunkten, oft mit Hilfe der euklidischen Distanz, bestimmt. Sobald keine Veränderungen mehr im Clustering durchgeführt werden, also keine Neuzuordnung der Punkte zu den Clustern erfolgt, wird der Algorithmus abgebrochen.

Der Mittelpunkt μ_c ist der Mittelwert aller Punkte im Cluster C und ist definiert als

$$\mu_c = (\bar{x}_1(C), \bar{x}_2(C), ..., \bar{x}_d(C)), \tag{9}$$

wobei

$$\bar{x}_j(C) = \frac{1}{n_C} \cdot \sum_{p \in C} x_j^p \tag{10}$$

ist, $\bar{x}_j(C)$ den Mittelwert der j-ten Dimension aller Punkte im Cluster C repräsentiert und n_C die Anzahl der Objekte im Cluster C darstellt [For65].

Bei k-means müssen im Vorhinein die Anzahl der Cluster k für die Zuordnung der Punkte sowie die initialen Mittelpunkte bekannt sein. Die Initialisierung der Mittelpunkte kann mit Hilfe mehrerer Stichproben erfolgen, aus denen die Mittelpunkte ermittelt werden. Jene mit dem besten Ergebnis wird dann als Ausgangssituation für das Clustering verwendet.

Der im Algorithmus 1 dargestellt Pseudocode von k-means wird im Folgenden kurz erläutert. K-means initialisiert zuerst leere Cluster und ordnet anschließend jedem Cluster jene Punkte zu, bei denen die Distanz zu den initialen Mittelpunkten M der Cluster minimal ist. Daraufhin werden die neuen Mittelpunkte μ_{c_i} der Cluster bestimmt. Im nächsten Schritt wird für jeden Punkt von jedem Cluster erneut die Distanz zu allen Mittelpunkten μ_{c_i} der Cluster ermittelt. Wird hier eine geringere Distanz zu einem anderen Cluster als den bestehenden gefunden, so wird der Punkt verschoben und die Mittelpunkte jener Cluster neu berechnet, die von dieser Verschiebung betroffen sind. Die Berechnung der Distanzen zwischen den Punkten und den Mittelpunkten wird so lange wiederholt, bis keine Verschiebung mehr erfolgt. Danach wird der Algorithmus beendet.

Algorithm 1 k-means (Punktmenge D, Integer k, Initiale Centroide M)

1: Initialisiere k leere Cluster $C = \{\dots, C_i, \dots\}$ mit $C_i = \{\}$
2: **for all** $\vec{x} \in D$ **do**
3: Bestimme für alle Mittelpunkte $\mu_{c_i} \in M$ die Distanzen $\Delta = \{\dots, d_i, \dots\}$ mit $d_i = d(\vec{x}, \vec{\mu}_{c_i})$ und bestimme den Index i, für die minimale Distanz aus Δ
4: Füge den Punkt $\vec{x}$ in den Cluster mit dem nähesten Mittelpunkt ein, d.h. $C_i \leftarrow \{\vec{x}\}$
5: **end for**
6: **for all** $i \in \{1, \dots, k\}$ **do**
7: $\mu_{c_i} \leftarrow (\sum_{\vec{x} \in C_i} \vec{x})/|C_i|$
8: **end for**
9: **repeat**
10: $M' \leftarrow M$
11: **for all** $C_i \in C$ **do**
12: **for all** $\vec{x} \in C_i$ **do**
13: Bestimme für alle Mittelpunkte $C_j \in C \backslash \{C_i\}$ die Distanzen $\Delta = \{\dots, d_j, \dots\}$ mit $d_j = d(\vec{x}, \vec{\mu}_{c_j})$
14: **if** $\exists d_j \in \Delta : d_j < d_i = d(\vec{x}, \vec{\mu}_{c_i}) \wedge \nexists d_k \in \Delta : d_k < d_j$ **then**
15: Verschiebe Punkt $\vec{x}$ und aktualisiere Mittelpunkte, d.h. $\vec{\mu}_{c_i} \leftarrow (|C_i|\vec{\mu}_{c_i} - \vec{x})/(|C_i| - 1)$; $\vec{\mu}_{c_j} \leftarrow (|C_j|\vec{\mu}_{c_j} + \vec{x})/(|C_j| + 1)$ $C_i \leftarrow C_i \backslash \{\vec{x}\}$; $C_j \leftarrow C_j \cup \{\vec{x}\}$
16: **end if**
17: **end for**
18: **end for**
19: **until** $M = M'$
20: **return** M

Zu den Eigenschaften von k-means zählen die folgenden:

Konvergenz in lokales Minimum: Es ist nicht garantiert, dass k-means das Optimum findet, d.h. die Lösung kann suboptimal sein.

Anzahl der Iterationen: Nach 5 bis 10 Iterationen ist das Clustering meist beendet, da keine weiteren Verschiebungen der Punkte mehr durchgeführt werden [ES00, Seite 54].

Laufzeitkomplexität einer Iteration: Der Aufwand für eine Iteration liegt bei O(n), da für jeden Punkt die Zuordnung zum Cluster bestimmt wird.

Hohe Sensibilität gegenüber Ausreißern: Da der Mittepunkt μ_c der Mittelwert aller Punkte eines Clusters ist, haben Ausreißer einen hohen Einfluss auf diesen und damit auf das gesamte Clustering.

Initiale Zerlegung wichtig: Die initiale Zerlegung ist ausschlaggebend für die Qualität des Ergebnisses und für die Anzahl der Iterationen [ES00, Seite 54].

Numerische Attribute anwendbar: Bei k-means sind nur numerische Attribute verwendbar, da die Distanzen bestimmt werden müssen.

Weitere Informationen über k-means findet man unter anderem bei [For65] und [Mac67].

2.2.3.5 Erwartungsmaximierung (EM)

Im Gegensatz zu k-means, das den Cluster durch einen Mittelpunkt beschreibt, wird bei der Erwartungsmaximierung, oder kurz EM, nach Dempster, Laird und Rubin [DLR77], der Cluster durch eine Wahrscheinlichkeitsverteilung beschrieben. Die gesamten Daten werden dabei als Mischung von k Wahrscheinlichkeitsverteilungen aufgefasst. Üblicherweise wird dafür eine multivariate Normalverteilung, oder auch Gaußverteilung genannt, verwendet. Gaußverteilungen werden benutzt, da sich jede Verteilung durch eine Mischung aus diesen approximieren lässt.

Bei EM geht man davon aus, dass die Daten in einem Clustering-Modell $M = \{G_1, \ldots, G_k\}$ auf Grund einer Mischung von k Wahrscheinlichkeitsverteilungen entstanden sind. Diese d-dimensionalen Gaußverteilungen eines jeden Clusters C sind dabei eindeutig definiert durch

- den Mittelpunkt μ_C aller Punkte eines Clusters und
- einer $d \times d$ Kovarianzmatrix Σ_C für alle Punkte im Cluster C.

Bei EM gehört jeder Punkt zunächst zu jedem Cluster, jedoch mit unterschiedlichen Wahrscheinlichkeiten. Deshalb ist die Wahrscheinlichkeit, mit der ein Punkt x bei einer bestimmten Gaußverteilung G eines Clusters C in den Daten vorkommt, nach der Gauß´schen Dichtefunktion,

$$P(x|G) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_C|}} e^{-\frac{1}{2}(x-\mu_C)^T (\Sigma_C)^{-1} (x-\mu_C)}. \tag{11}$$

Für k Gaußverteilungen lässt sich dann die Wahrscheinlichkeit bestimmen, mit der bei diesem Modell ein Punkt x in den Daten vorkommt. Hierbei ist W_i der Anteil der Daten, die der Gaußverteilung G_i zugeordnet werden. Daher gilt

$$P(x) = \sum_{i=1}^{k} W_i P(x|G_i). \tag{12}$$

Auf Grund der obigen Wahrscheinlichkeiten lässt sich schließlich die Wahrscheinlichkeit bestimmen, mit der ein Punkt x einem bestimmten Cluster C_i bzw. einer bestimmten Gaußverteilung G_i zugeordnet werden kann. Dabei gilt

$$P(G_i|x) = W_i \frac{P(x|G_i)}{P(x)}. \tag{13}$$

Die Güte eines Modells $M = \{G_1, ..., G_k\}$ lässt sich auf Grund der Wahrscheinlichkeiten für die gegebene Datenmenge als Wert E bestimmen. Der Wert E ist als die Summe der logarithmierten Einzelwahrscheinlichkeiten definiert, siehe Formel 14. Je größer dieser Wert ist, desto eher entspricht das angenommene Modell der tatsächlichen Verteilung der Daten.

$$E(M) = \sum_{x \in D} log(P(x)) \tag{14}$$

Bei der Neuberechnung des Modells $M = \{G_1, ..., G_k\}$ werden die Parameter W_i, μ_c und Σ_c der k Gaußverteilungen neu berechnet, wobei

$$W_i = \frac{1}{n} \sum_{x \in D} P(G_i|x), \tag{15}$$

$$\mu_c = \frac{\sum\limits_{x \in D} x \cdot P(G_i|x)}{\sum\limits_{x \in D} P(G_i|x)}, \text{ und} \tag{16}$$

$$\Sigma_c = \frac{\sum\limits_{x \in D} P(G_i|x)(x - \mu_c)(x - \mu_c)^T}{\sum\limits_{x \in D} P(G_i|x)}. \tag{17}$$

Beim Algorithmus von EM ist das initiale Modell wichtig, da es Auswirkungen auf die Qualität des Ergebnisses haben kann. Daher wird für die Initialisierung häufig k-means verwendet, wodurch das initiale Modell auf Grund der zu clustern-

den Daten und nicht zufällig entsteht. So können von k-means die Mittelpunkte direkt für die Gaußverteilungen übernommen werden. Außerdem entsprechen bei Clustern mit identischer Form gleiche Distanzen gleichen Wahrscheinlichkeiten.

Im Folgenden wird der unter Algorithmus 2 dargestellte Pseudocode von EM näher ausgeführt. Dabei wird als erster Schritt das Modell mit den k Gaußverteilungen auf Grund der Initialisierung bestimmt. Anschließend erzeugt der Algorithmus k leere Cluster in die später die Punkte eingeteilt werden. Für jeden Punkt werden anschließend die Wahrscheinlichkeiten gemäß den Formeln 11, 12 und 13 für alle Cluster ermittelt. Jenem Cluster mit der höchsten Wahrscheinlichkeit der Clusterzugehörigkeit $P(G_i|x)$ wird der Punkt schließlich zugeordnet. Nachdem alle Punkte den Clustern zugeordnet worden sind, werden die Parameter W_i, μ_c und Σ_c, die die Gaußverteilungen der Cluster bestimmen, neu berechnet. Die Güte des daraus entstehenden neuen Modells wird im Anschluss mit der Güte des vorangegangenen Modells verglichen. Unterscheiden sich diese mehr als die Gütedifferenz ϵ, das die Genauigkeit des Clustering-Modells bestimmt, so erfolgt eine erneute Iteration des Algorithmus, beginnend mit der Erstellung von k leeren Clustern. Andernfalls kann der Algorithmus abgebrochen werden.

Zu den Eigenschaften des EM Algorithmus zählen die folgenden:

Konvergenz in lokales Maximum: Es ist nicht garantiert, dass EM das beste Modell der Gaußverteilungen auf Grund der Daten bestimmt, d.h. die Lösung kann suboptimal sein.

Anzahl der Iterationen : Die Anzahl der Iterationen ist bei dieser Methode meistens sehr hoch [ES00, Seite 62].

Laufzeitkomplexität einer Iteration: Der Aufwand für eine Iteration liegt bei $O(n)$, da für jeden Punkt die Zuordnung zu allen Gaußverteilungen des Modells bestimmt wird.

Initiale Zerlegung wichtig: Die initiale Zerlegung ist ausschlaggebend für die Qualität des Ergebnisses und für die Anzahl der Iterationen [ES00, Seite 62].

Kontinuierliche und kategorische Attribute anwendbar: Bei EM sind sowohl kontinuierliche als auch kategorische Attribute verwendbar, da keine Distanzen bestimmt werden.

Algorithm 2 Erwartungsmaximierung(Punktmenge D, Integer k, Gütedifferenz eps)

1: Erzeuge ein initiales Modell $M' = \{G_1, \ldots, G_i, \ldots, G_k\}$ mit k Gaußverteilungen $G_i = (\vec{\mu}_i, \Sigma_i, W_i)$ für die Daten D;
2: **repeat**
3: Erzeuge k anfangs leere Cluster $C_i \leftarrow \{\}$ mit $i = \{1, \ldots, k\}$
4: $M \leftarrow M'$
5: **for all** $\vec{x} \in D$ **do**
6: **for all** $G_i \in M$ **do**
7: $P(\vec{x}|G_i) \leftarrow \phi((\vec{x} - \vec{\mu}_i)|\Sigma|)$
8: $P(\vec{x}) \leftarrow \sum_{i=1}^{k} W_i P(\vec{x}|G_i)$
9: $P(G_i|\vec{x}) \leftarrow W_i \frac{P(\vec{x}|G_i)}{P(\vec{x})}$
10: **end for**
11: suche Index i, für den die Wahrscheinlichkeit $P(G_i|\vec{x})$ maximal wird
12: $C_i \leftarrow C_i \cup \{\vec{x}\}$
13: **end for**
14: **for all** $i \in \{1, \ldots, k\}$ **do**
15: $W_i \leftarrow |C_i|$
16: $\vec{\mu}_i \leftarrow \frac{1}{W_i} \sum_{\vec{x} \in C_i} \vec{x}$
17: $\Sigma_i \leftarrow \frac{\sum_{\vec{x} \in D} P(G_i|\vec{x})(\vec{x}-\mu_i)(\vec{x}-\mu_i)^T}{\sum_{\vec{x} \in D} P(G_i|\vec{x})}$
18: **end for**
19: **until** $|E(M) - E(M') < eps|$
20: **return** M

Für eine bessere Übersicht, werden noch einmal die grundlegenden Eigenschaften von k-means und EM in der folgenden Tabelle gegenübergestellt.

Eigenschaft	k-means	EM
Mögliche Konvergenz in lokales:	Minimum	Maximum
Anzahl der Iterationen:	niedrig	hoch
Laufzeitkomplexität pro Iteration:	$O(n)$	$O(n)$
Initiale Zerlegung:	wichtig	wichtig
Verwendbare Attribute:	numerische	kontinuierliche und kategorische

Tabelle 1: Übersicht zu den Eigenschaften von k-means und EM

2.2.4 Problematik bei der Wahl der Clustering-Methode

Die grundlegende Problematik bei der Wahl der Clustering-Methode liegt darin, dass man die Cluster kennen muss, um die richtige Clustering-Methode auswählen zu können. Es ist deshalb notwendig ein Clustering-Verfahren mit variierenden Parametern mehrmals anzuwenden, bis eine zufriedenstellende Lösung gefunden wird.

Das in Kapitel 5 beschriebene Verfahren CHAD versucht dieser Problematik zu begegnen, da man dort zum einen mehr über die Struktur der Daten auf Grund einer hierarchischen Repräsentation weiß, und zum anderen mehrmals ein Verfahren mit unterschiedlichen Parametern auf diese Repräsentation angewendet werden kann.

Des Weiteren wurden von Fisher und Van Ness die folgenden Kriterien für die Wahl der Clustering-Methode definiert, wodurch ein besserer Vergleich der Methoden ermöglicht werden soll [FN71].

Die Form des Clusters: Nicht alle Clustering-Methoden können alle Formen der Cluster erkennen, da viele Methoden bestimmte Formen auf Grund der Ähnlichkeitsmessungen und der Gruppierungskriterien einfach annehmen.

Die Struktur der Daten: Die Struktur der Daten bestimmt die Cluster. Je mehr Informationen über die Daten bekannt sind, desto eher wird die wahre Struktur der Cluster erkannt.

Die Sensitivität der Clustering-Methode: Manche Methoden werden durch Ausreißer stark beeinflusst und andere können diese erkennen und somit das Clustering verbessern.

Auf Grund der Kriterien von Fisher und Van Ness kann keine konkrete Methode bestimmt werden, sie geben allerdings eine gewisse Richtung für die geeignetste Clustering-Methode an.

3 Alternative Ansätze

In diesem Kapitel werden aktuelle Ansätze für die Aufgaben des Data-Mining vorgestellt, die eine Verbesserung der Laufzeit bei großen Datenmengen zum Ziel haben, wobei im Besonderen das Clustering-Verfahren EM berücksichtigt wird.

Generell verfolgen die Ansätze zur Analyse von großen Datenmengen zwei unterschiedliche Strategien. Zu diesen zählt zum einen die Verwendung einer repräsentativen Stichprobe für die Analyse, um dadurch Aussagen über die gesamten Daten treffen zu können. Da bei EMAD und CHAD keine Stichproben zur Verringerung der Gesamtlaufzeit genutzt werden, wird hier nicht näher darauf eingegangen sondern für weitere Ausführungen auf Goller [Gol06, Abschnitt 3.2] verwiesen.

Die zweite Strategie beruht auf der Komprimierung der Daten, auf Grund derer im Anschluss die konkrete Analyse durchgeführt wird. Ein Überblick zu dieser Strategie wird in Abschnitt 3.1 gegeben.

In Abschnitt 3.2 wird ein dieser Arbeit ähnlicher Ansatz detailliert vorgestellt, das Skalierbare EM Clustering, das auf Grund komprimierter Daten ein Clustering basierend auf EM durchführt. Abschnitt 3.3 stellt einen weiteren zu EMAD ähnlichen Ansatz vor, den Algorithmus FREM, der neben der Verwendung von komprimierten Daten noch zusätzlich die Minimierung der Iterationen verfolgt. Diese beiden Ansätzen haben, genauso wie diese Arbeit, die Verbesserung der Laufzeit und der Qualität im Vergleich zum klassischen EM zum Ziel.

Für weitere Beschreibungen zu Ansätzen, die für große Datenmengen entwickelt wurden und nicht auf EM basieren, wird auf den Überblick von [BKKK04] verwiesen.

Abschließend wird in Abschnitt 3.4 ein Vergleich der hier ausführlich vorgestellten Ansätze, dem Skalierbaren EM Clustering und FREM, mit EMAD durchgeführt.

3.1 Komprimierung von Daten

Eine Komprimierung der Daten verbessert die Laufzeit der Verfahren bei großen Datenmengen, indem eine mengenmäßig kleinere alternative Repräsentation erstellt wird, wodurch sich die zu verarbeitende Datenmenge bei der Analyse stark

reduziert. Da viele Analyseverfahren sowohl mit den Statistiken, welche die Daten repräsentieren, als auch mit den Daten selbst ausgeführt werden können, ist diese Umwandlung möglich.

Für die Komprimierung der Daten werden oft Clustering-Feature-Vektoren, wie sie auch BIRCH verwendet, herangezogen. Dabei kann mit dem Mittelpunkt der Clustering-Feature-Vektoren wie mit einem n-fachem Tupel gerechnet werden. Die Berechnungen der Verfahren ändern sich dabei nicht wesentlich. Für eine Beschreibung der Clustering-Feature-Vektoren von BIRCH wird auf Abschnitt 2.2.3.2 verwiesen.

Eine Weiterentwicklung der Komprimierung, wie sie bei BIRCH erfolgt, stellen die sogenannten Data Bubbles von Breunig, Kriegel, Kröger und Sandar [BKK01] dar. Dabei werden neben den Statistiken der komprimierten Daten, wie lineare Summe, quadratischen Summe und der Anzahl der Tupel, noch zusätzliche Informationen, wie z.B. die durchschnittliche Distanz zum nächsten Nachbarn, aggregiert, sodass diese Komprimierung besonders für die Verwendung von hierarchischen Clustering-Verfahren nützlich ist.

Ein Überblick zu Verfahren, die bei großen Datenmenge die Komprimierung der Daten zur Bestimmung eines schnelleren Ergebnisses verwenden, wird bei [Gol06, Abschnitt 3.5] gegeben.

3.2 Skalierbares EM Clustering

Das Skalierbare EM Clustering verwendet den EM Algorithmus im Skalierbaren System von Bradley, Fayyad und Reina [BFR98b]. Dieser Ansatz ist besonders für große Datenmengen geeignet. Dessen Eigenschaften und der Ablauf des Algorithmus werden im Folgenden kurz beschrieben.

Das Skalierbare System hat die Eigenschaft, dass die Daten für das Clustering nur einmal gelesen werden müssen. Weiters ist das momentan „beste“ Ergebnis während der Ausführung des Algorithmus bestimmbar. Außerdem kann der Algorithmus jederzeit angehalten und später wieder gestartet werden, wobei neue Daten in das Clustering hinzugefügt werden können. Der Benutzer bestimmt die Größe des Speichers, in den eine Teilmenge der Datenpunkte gelesen wird, wodurch nicht die gesamten Punkte im Hauptspeicher sind, was die Verarbeitung von belie-

big großen Datenmengen und das Bestimmen von mehreren Clustering-Modellen parallel ermöglicht [BFR98b, Seite 2].

Nach Bradley, Fayyad und Reina können die Datenpunkte in drei Kategorien eingeteilt werden [BFR00, Kapitel 2]. In die erste Kategorie sind verwerfbare Datenpunkte, „discard set“ DS, einzuordnen, die durch Statistiken nachgebildet werden können und deshalb aus dem Speicher gelöscht werden. Die zweite Kategorie stellen die komprimierbaren Datenpunkte dar, „compress set“ CS, die in Gruppen zusammengefasst und durch Statistiken, die im Speicher verbleiben, repräsentiert werden. Zu der letzten Kategorie zählen jene Datenpunkte, „retained set“ RS, die vollständig im Speicher verbleiben.

In Abbildung 9 ist der Ablauf des Skalierbaren EM Clustering dargestellt, der in mehreren Iterationen ein Clustering-Ergebnis für die Daten bestimmt und im Folgenden kurz beschrieben wird. Die initialen Parameter, mit dem der Algorithmus startet, werden entweder vom Benutzer oder per Zufall bestimmt. Anschließend wird eine Teilmenge der Datenpunkte in den Speicher (RAM Buffer) geladen. Auf diese geladenen Daten und auf die bereits komprimierten Daten, die das bestehende Modell aufbauen, wird der EM Algorithmus angewandt um ein neues Modell von Gaußverteilungen zu bestimmen. Durch dieses Modell kann jeder einzelne Datenpunkte im Speicher den bereits genannten drei Kategorien zugeordnet werden, sodass dieser entweder verworfen, komprimiert oder gespeichert wird. Wird der Algorithmus auf Grund des bestehenden Stopp-Kriteria (Termination Criteria) nicht beendet, werden abermals Datenpunkte in den Speicher geladen und das Modell wird erneut bestimmt.

Ausführliche Erläuterungen zur Komprimierung der Daten und zum Erstellen der Statistiken sind bei [BFR98b, Seiten 7ff] zu finden. Neben der Clustering-Methode EM ist das skalierbare System auch mit k-means anwendbar, was bei [BFR98a] beschrieben wird.

Das Skalierbare EM Clustering verarbeitet die große Menge an Daten, ähnlich wie CHAD, durch eine Aggregation der Daten, die den Clustering-Feature-Vektoren von BIRCH entsprechen. Die Modelle von Gaußverteilungen, die bei großen Datenmengen bestimmt werden, sind, wie der Vergleich zum klassichen EM Algorithmus bei Bradley, Fayyad und Reina gezeigt hat, im Bezug auf die Qualität besser und brauchen nur 20 % der Laufzeit [BFR98b, Seite 18].

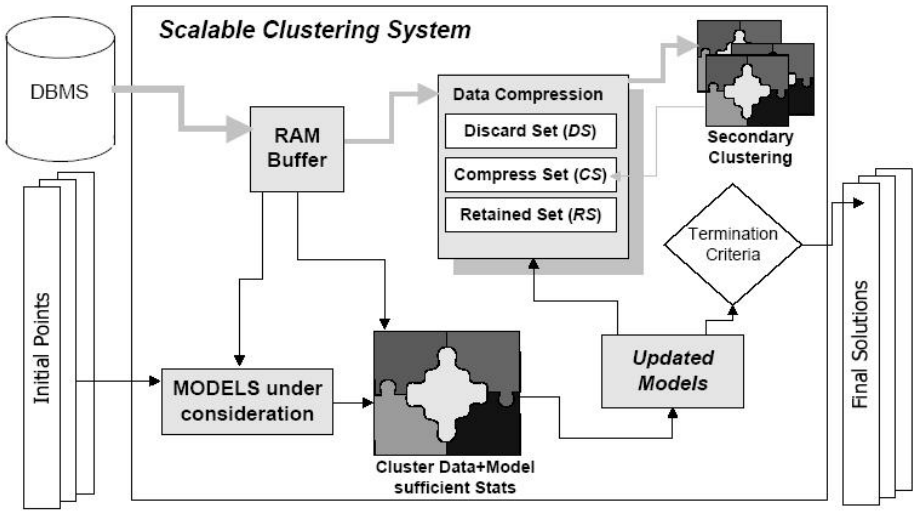

Abbildung 9: Skalierbares Clustering System [BFR98a, Abbildung 1]

3.3 FREM

In diesem Abschnitt soll der Algorithmus FREM, Fast and Robust EM, von Ordonez und Omiecinski vorgestellt werden, der auf dem EM Algorithmus basiert, und im Besonderen für große Datenmengen entwickelt worden ist [OO02]. Damit kann gegenüber dem klassischen EM Algorithmus die Qualität der Ergebnisse gesteigert und gleichzeitig die Laufzeit verkürzt werden. Eine Verbesserung der Laufzeit beruht hier nicht nur auf einer Aggregation der Daten, wie sie bei EMAD verwendet wird, sondern es wird auch versucht die Anzahl der Iterationen stark zu reduzieren. Im Folgenden wird auf die Eigenschaften von FREM eingegangen und der Ablauf des Algorithmus kurz beschrieben.

Nach Ordonez und Omiecinski kann FREM die Eigenschaften von EM, die sich bei großen Datenmengen negativ auswirken, vermeiden [OO02, Abschnitt 1.1]. Zu diesen Eigenschaften von EM zählt die Notwendigkeit einer guten Initialisierung um ein gutes Resultat zu erreichen. Ansonsten ist ein erneutes Ausführen des Algorithmus notwendig. Weiters kann das Ergebnis von EM nur zu einem lokalen Optimum führen, wodurch weitere Iterationen notwendig werden. Die Anzahl der Iterationen von EM können sich auf Grund dieser Eigenschaften erhöhen, was bei großen Datenmengen eine starke Auswirkung auf die Laufzeit hat.

Für die Senkung der Laufzeit bei FREM werden Zusammenfassungen über die einzelnen Cluster bestimmt, wodurch sich die Zeit für die Ein- und Ausgabe verringert. Diese Zusammenfassungen bestehen aus zwei Matrizen, die zum einen die Summe der Datenpunkte, und zum anderen die Summe der quadrierten Datenpunkte je Cluster enthalten. Die Dimension dieser Matrizen wird durch die Anzahl der Cluster und die Anzahl der Attribute je Datenpunkt bestimmt. Zusätzlich wird bei den Zusammenfassungen die Anzahl der Datenpunkte je Cluster gespeichert. Für die Bestimmung der Gaußverteilungen werden, ähnlich wie bei EMAD, die Berechnungen des klassischen EM mit Hilfe der Zusammenfassungen durchgeführt.

Für die Reduzierung der Iterationen von FREM wird die Anzahl der Neuberechnungen des gesamten Modells mit mehreren Gaußverteilungen, was beim klassischen EM Algorithmus nur einmal je Iteration erfolgt, erhöht. Bei kleinen Datenmengen entspricht die Anzahl der Neuberechnungen je Iteration der Anzahl der Cluster. Bei großen Datenmengen wird die Anzahl der Neuberechnungen je Iteration noch weiter erhöht. Durch diese Erhöhung kann die Sensitivität gegenüber der Reihenfolge bei wenigen Datenpunkten gemindert und die Kovergenz bei vielen Datenpunkten beschleunigt werden [OO02, Seite 593].

Für eine gute Qualität werden als Initialisierung bei FREM der Mittelpunkt und die Kovarianz aller Datenpunkte herangezogen, die mittels eines Scans der gesamten Daten bestimmt werden. Die Cluster haben bei der Initialisierung den gleichen Anteil an Datenpunkten sowie eine identische Kovarianzmatrix. Der Mittelpunkt je Cluster wird auf Grund einer bestimmten Streuung je Dimension, einer Zufallszahl und der Variable, die die Entfernung zum Mittelpunkt aller Datenpunkte überwacht, bestimmt. Dadurch liegen alle Mittelpunkte der Cluster um den Mittelpunkt aller Datenpunkte.

Zur Vermeidung eines lokalen Optimums werden bei FREM während der Neuberechnung des Modells die Cluster analysiert und gegebenenfalls geteilt oder verlegt. Unterschreitet der Anteil der Datensätze eines Clusters einen bestimmten Wert, so wird dieser Cluster versetzt, indem ein Cluster mit einem hohen Anteil an Datensätzen geteilt wird.

Der Algorithmus FREM kann auf Grund der bereits erwähnten Zusammenfassungen, Neuberechnungen, Analysen und der Initialisierung mit nur drei Iterationen ein Clustering-Ergebnis von guter Qualität bestimmen. Dabei haben die ersten

beiden Iterationen die Aufgabe eine gute Approximation für das Ergebnis zu bestimmen. Da auf Grund der oftmaligen Neuberechnungen des Modells die beiden ersten Iterationen sensibel auf die Reihenfolge der Daten reagieren, ist mindestens eine dritte Iteration, die dem Ablauf des klassischen EM entspricht, notwendig.

Der Vergleich zwischen FREM und dem klassischen EM bei Ordonez und Omiecinski hat gezeigt, dass die Laufzeit von FREM schon bei kleinen Datenmengen nur 30% ausmacht, bei einer steigenden Anzahl an Datensätzen linear und nicht exponentiell ansteigt, und die Qualität von EM übertroffen werden kann [OO02, Seite 598].

Der Aufwand für FREM liegt bei $O(kdn)$, da bei jedem Datenpunkt für jede Dimension d und jeden Cluster k Berechnungen durchgeführt werden. Im Vergleich dazu hat EMAD für das Clustering einen geringeren Aufwand von $O(kds)$, da nicht die gesamten Datenpunkte n sondern nur deren Aggregationen, die Sub-Cluster s, für das Clustering herangezogen werden.

Für den interessierten Leser ist eine detaillierte Beschreibung zum Ablauf und zu den Ergebnissen von FREM bei [OO02] zu finden.

3.4 Vergleich der Ansätze

Die Gemeinsamkeit, die EMAD mit den hier vorgestellten Ansätzen in Abschnitt 3.2 und 3.3 aufweist, ist die Verwendung von Aggregationen der Datenpunkte, da damit eine große Menge an Daten interaktiv behandelt werden kann.

Die Form der Aggregationen von diesen Ansätzen unterscheidet sich in gewissen Bereichen zu dem Ansatz CHAD, der in dieser Arbeit verwendet wird und in Kapitel 5 ausführlich beschrieben ist. So sind die Aggregationen beim Skalierbaren EM Clustering den Clustering-Feature-Vektoren von BIRCH ähnlich. FREM aggregiert die Datenpunkte jedoch in Matrizen.

Die in den vorangegangen Abschnitten ausführlich beschriebenen Ansätze werden für das Clustering mittels EMAD in der hier beschriebenen Form nicht direkt übernommen, da diese die Aggregation der Daten und das Clustering-Verfahren in einem Schritt ausführen, wodurch die Verwendung der Aggregationen durch andere Verfahren für weitere Anwendungen nicht möglich ist. Daher baut EMAD auf CHAD auf und übernimmt nicht die Aggregationen dieser Ansätze.

Die Bestimmung der Gaußverteilungen auf Grund der Aggregationen wird bei EMAD, FREM und dem Skalierbaren EM Clustering weitestgehend in der gleichen Form durchgeführt.

4 Vorausschauendes Data-Mining

Nachdem im Kapitel 2 auf die Grundlagen des Knowledge Discovery in Databases und im Speziellen auf das Clustering eingegangen wurde, soll in diesem Kapitel das Vorausschauende Data-Mining beschrieben werden, das den KDD-Prozess geringfügig verändert, um dadurch schneller zu einem Ergebnis zu kommen bzw. um die Qualität des Ergebnisses zu verbessern. Der im Rahmen dieser Arbeit entwickelte Algorithmus EMAD stellt eine Erweiterung des Vorausschauenden Data-Minings dar.

Im folgenden Abschnitt werden Fälle erläutert, die beim klassischen KDD-Prozess eine hohe Laufzeit verursachen. Durch das Vorausschauende Data-Mining soll diese verringert werden. In Abschnitt 4.2 wird das Vorausschauende Data-Mining definiert und näher ausgeführt.

4.1 Fälle des KDD-Prozesses mit hoher Laufzeit

In Abschnitt 2.1.1 wird Knowledge Discovery in Databases als iterativer Prozess, ausgehend von den Daten bis hin zum Wissen, definiert. Abbildung 10 zeigt den klassischen KDD-Prozess vereinfacht vom Problem bis hin zum Resultat.

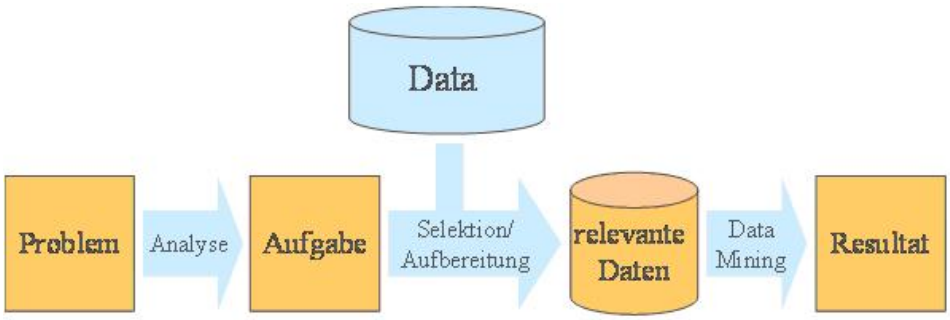

Abbildung 10: Klassische Vorgehensweise bei einer Data-Mining Aufgabe [Gol04, Seite 14]

Der in Abbildung 10 dargestellte KDD-Prozess kann in gewissen Situationen für ein gutes Ergebnis eine hohe Laufzeit erfordern. Eine Auswahl solcher Fälle wird im Folgenden kurz dargestellt.

Parameter-Wahl: Bei der Wahl der Parameter für eine Data-Mining Aufgabe muss sich der Benutzer auf seine Erfahrungen oder teilweise auch auf sein Glück verlassen, da diese nicht konkret definiert werden können. Im Gegensatz dazu können die Verfahrensklasse, z.B. Clustering, auf Grund des Problems bzw. der Aufgabe und das passende Verfahren der jeweiligen Klasse durch die Daten bestimmt werden.

Auf Grund der Unsicherheit, die sich durch die Wahl der Parameter ergibt, liefert das Data-Mining Verfahren nicht immer richtige bzw. nutzbare Resultate. Dadurch wird ein erneutes Ausführen des gleichen Verfahrens auf dieselben Daten mit veränderten Parametern notwendig. Da das Data-Mining im KDD-Prozess den größten Teil der Laufzeit beansprucht, wird durch das nochmalige Ausführen des Data-Mining Algorithmus die Gesamtlaufzeit des KDD-Prozesses drastisch erhöht.

Ähnliche Anwendungen: Es kann vorkommen, dass ähnliche Anwendungen die gleichen Daten verarbeiten. Dadurch werden mehrere KDD-Prozesse ausgeführt, deren Ziele sich nur geringfügig voneinander unterscheiden. Die KDD-Prozesse laufen dabei isoliert ab, wodurch für jeden eine eigenständige Initialisierung und die gesamte Verarbeitung durchgeführt werden muss. Die Ergebnisse der Prozesse nehmen dabei keinerlei Einfluss aufeinander und die Initialisierungen, die sich sehr ähnlich sind, erhöhen die Laufzeit jedes einzelnen KDD-Prozess unnötig [Für04, Seite 42].

Trial and Error: Das Data-Mining braucht für die Auswahl der Verfahrensklasse, die sich auf Grund der Aufgabe ergibt, oder der Auswahl des konkreten Verfahrens, das von den Daten abhängig ist, menschlichen Input. Minimiert man diesen Input wegen mangelnder Domainkenntnisse oder diffuser Rahmenbedingungen der Aufgabe, so kann man auch von einer Trial and Error Strategie sprechen, um Muster in den Daten zu finden.

Bei der Trial and Error Strategie wird ein Data-Mining Verfahren mit unterschiedlichen Parametern ausgeführt, und anschließend werden die ermittelten Ergebnisse dieses Verfahrens bewertet. Am Ende wird die beste Lösung als Endergebnis ausgewählt. Diese Vorgehensweise führt beim klassischen KDD-Prozess zu einem hohen und unnötigen Aufwand bei der Selektion und Aufbereitung [Für04, Seite 4].

Lokales Optimum: Data-Mining wird oft bei großen Datenbeständen eingesetzt, bei denen die Untersuchung des gesamten Lösungsraums auf Grund der langen Laufzeit nicht umsetzbar ist. Daher werden Algorithmen angewandt, die in angemessener Zeit ein Ergebnis liefern. Dazu werden beispielsweise Heuristiken und statistische Näherungsverfahren verwendet. Es besteht die Möglichkeit, dass diese auf Grund der Rahmenbedingungen allerdings nur zu einem lokalen Optimum führen.

Um das globale Optimum für eine Problemstellung zu ermitteln, wären zusätzliche Ansätze notwendig, die z.B. mit unterschiedlichen Initialisierungen und einer Bewertungsfunktion versuchen, dieses festzustellen [Für04, Seite 44]. Durch diese Vorgehensweise wird allerdings die Laufzeit des KDD-Prozesses zusätzlich erhöht.

Die beschriebenen Fälle zeigen, dass oftmalige Iterationen im klassischen KDD-Prozess zu einer hohen Laufzeit führen. Dies kann durch die Anwendung von Zwischenergebnissen vorangegangener Prozesse verringert werden. Das Vorausschauende Data-Mining setzt dies um, indem auf Grund universell aufbereiteter Daten mehrere unterschiedliche Anwendungen durchgeführt werden können.

4.2 Definition des Vorausschauenden Data-Mining

Die Idee des Vorausschauenden Data-Mining nach Goller ist die Berechnung von Zwischenergebnissen, die von mehreren noch nicht bekannten Prozessen später verwendet werden können [Gol06, Kapitel 4]. Dabei ist der dadurch entstehende, zusätzliche Aufwand für die Zwischenergebnisse gering, der daraus resultierende Vorteil für die nachfolgenden Prozesse jedoch hoch.

Das Vorausschauende Data-Mining unterteilt den KDD-Prozess in zwei Teile, sodass ein Teil anwendungsunabhängig und einer anwendungsabhängig ist. Dadurch kann ersterer mehrmals als Grundlage für unterschiedliche Anwendungen verwendet werden, was die Laufzeit der anwendungsabhängigen Teile verringert. Beide Teile beinhalten jeweils zwei Verarbeitungsschritte, wodurch sich das Vorausschauende Data-Mining insgesamt in vier Phasen einteilen lässt, die im Folgenden kurz beschrieben werden.

Phase 1: Die erste Phase des Vorausschauenden Data-Mining beinhaltet all jene Aufgaben des klassischen KDD-Prozesses, die anwendungsunabhängig sind, wie beispielsweise die Bereinigung oder Integration der Daten.

Phase 2: Phase zwei zählt genauso wie Phase eins zum anwendungsunabhängigen Teil des Vorausschauenden Data-Mining. Dabei werden Algorithmen des Data-Mining angewendet, die Zwischenergebnisse bzw. Aggregationen der Daten bestimmen, die von zukünftigen Anwendungen verwendet werden.

Phase 3: Die dritte Phase gehört zum anwendungsabhängigen Teil des Vorausschauenden Data-Minings. Dabei werden die Daten genauso wie beim klassischen KDD-Prozess für die spezifische Anwendung selektiert, projeziert und transformiert. Als Grundlage für diese Operationen werden die Zwischenergebnisse aus Phase zwei verwendet.

Phase 4: In der vierten und letzten Phase wird der Data-Mining Algorithmus auf die Daten aus Phase drei angewendet um Muster bzw. Modelle in den aufbereiteten Daten zu finden. Durch die aggregierten Daten ist die Laufzeit des Algorithmus gegenüber jenem im klassischen KDD-Prozess geringer.

Kapitel 5 erläutert das Konzept CHAD, das auf dem Vorausschauenden Data-Mining beruht, und die oben beschriebenen Phasen zwei bis vier umsetzt. Der in Kapitel 6 beschriebene Algorithmus EMAD wurde für CHAD entwickelt und wird in der vierten Phase des Vorausschauenden Data-Mining ausgeführt.

5 Bestehendes System: CHAD

In diesem Kapitel wird das bestehende System CHAD vorgestellt, in das der Algorithmus EMAD eingebunden ist, der im Rahmen dieser Arbeit entwickelt wurde. CHAD besteht aus drei Phasen, wobei in der ersten Phase die gesamten Daten aggregiert werden. In der zweiten Phase werden die Daten für die konkrete Anwendung in Phase drei angepasst. In der dritten Phase wird ein Data-Mining Verfahren angewendet.

Allgemeine Erläuterungen zu CHAD sind in Abschnitt 5.1 zu finden. Die erste und die zweite Phase werden in den Abschnitten 5.2 und 5.3 näher ausgeführt.

Für die Initialisierung von EMAD wurde eine Weiterentwicklung des K-Centroiden Clustering (KC) von Fürst [Für04, Abschnitt 4.5] verwendet, das ein Clustering basierend auf k-means durchführt und ebenfalls bei CHAD in der dritten Phase angewendet werden kann. In Abschnitt 5.4 wird diese Initialisierung dargestellt.

5.1 CHAD

Der Algorithmus CHAD, Clustering Hierarchically Aggregated Data, wurde am Institut für Wirtschaftsinformatik - Data & Knowledge Engineering - an der Johannes Kepler Universität Linz, im Zuge der Dissertation von Goller entwickelt [Gol06].

CHAD hat das Ziel, auf Grund von aggregierten Daten mit einem Data-Mining Verfahren ähnliche Ergebnisse zu bestimmen, die auch bei der Verwendung von nicht aggregierten Daten entstanden wären. Durch die Aggregation soll allerdings die Laufzeit des Clusterings reduziert werden.

CHAD setzt die zweite bis vierte Phase des Vorausschauenden Data-Mining, das in Abschnitt 4.2 näher erläutert wurde, um und kann damit verschiedene Verfahrensklassen des Data-Mining kombinieren. Im Zuge dieser Arbeit wurden zwei Clustering-Verfahren hintereinander ausgeführt. Zu diesen zählt zum einen ein Algorithmus, der im Wesentlichen auf BIRCH, siehe Abschnitt 5.2, basiert, und zum anderen der Algorithmus EMAD, der auf EM beruht. Durch diese Kombination wird versucht, die Laufzeit des zweiten Algorithmus durch die Ergebnisse des ersten

zu verbessern. Außerdem soll ein mehrmaliges Ausführen des zweiten Algorithmus ermöglicht werden.

CHAD besteht grundsätzlich aus 3 Phasen:

Phase 1: Die erste Phase entspricht dabei im Wesentlichen der ersten Phase des Algorithmus BIRCH, näheres dazu siehe Abschnitt 2.2.3.2. Dieser erzeugt eine hierarchische Repräsentation (Dendrogramm) der gesamten Daten mittels eines einzigen Scans. Da alle Daten berücksichtigt werden, entsteht eine anwendungsunabhängige Repräsentation. Die Clustering-Feature-Vektoren dieses Dendrogramms sind erweitert worden, um die Selektion, Projektion und Transformation in der zweiten Phase zu ermöglichen und werden in der Folge General Cluster Feature genannt. Das hier erzeugte Dendrogramm wird daher als General Cluster Feature Baum oder CFG-Baum bezeichnet. Weitere Ausführungen zu dieser Phase sind in Abschnitt 5.2 zu finden.

Phase 2: In der zweiten Phase wird das aus der ersten Phase erzeugte Dendrogramm für die jeweilige, spezifische Anwendung selektiert, projeziert und transformiert und das daraus resultierende Ergebnis für die erneute Verwendung in einem separaten Dendrogramm gespeichert. In der Folge wird dieses Dendrogramm als CFE-Baum bezeichnet. Diese separate „Verkleinerung“ des CFG-Baums durch den Benutzer ermöglicht es die Wiederverwendung des anwendungsunabhängigen CFG-Baums für mehrere unterschiedliche Anwendungen. Weitere Ausführungen zur zweiten Phase sind in Abschnitt 5.3 zu finden.

Phase 3: Bei der dritten Phase wird das jeweilige Data-Mining-Verfahren auf den anwendungsabhängigen CFE-Baum aus der zweiten Phase ausgeführt. In dieser Arbeit wird bei dieser Phase das angepasste EM Verfahren EMAD verwendet. Als Initialisierung für EMAD dient die Weiterentwicklung von KC. Es beinhaltet das Clustering-Verfahren k-means, das oft als Initialisierung für EM verwendet wird. Das KC wird in der dritten Phase von CHAD ausgeführt und ist in Abschnitt 5.4 näher erläutert.

Goller hat das Konzept sowie die zweite Phase von CHAD entwickelt und BIRCH für die erste Phase angepasst [Gol06, Kapitel 5]. Für die dritte Phase sind bislang

drei Algorithmen erstellt worden: Das K-Centroid Clustering von Fürst [Für04, Abschnitt 4.5], eine Naive Bayes Klassifikation von Goller [Gol06, Kapitel 6], sowie der im Rahmen dieser Arbeit entwickelte EMAD Algorithmus, siehe Abschnitt 6.

5.2 Anwendungsunabhängige Repräsentation der Daten - CFG-Baum

Bei der anwendungsunabhängigen Datengrundlage, dem CFG-Baum, sind die gesamten Daten mit all ihren Attributen aggregiert enthalten. Dieser CFG-Baum (Dendrogramm) resultiert aus der ersten Phase von CHAD, die im Wesentlichen der ersten Phase des hierarchischen Clustering Verfahren BIRCH entspricht.

Unterschiede zwischen CHAD und BIRCH sind hier beim Neueinfügen von Knoten und bei der Berechnung des Schwellenwertes T zu finden. Wenn ein Knoten voll ist, so muss dieser geteilt werden. Hierbei greift CHAD auf die Extremwerte der Sub-Cluster zurück, um die notwendigen Einträge für das Splitten zu bestimmen. Weitere Ausführungen zu diesen Unterschieden sind bei [Gol06, Abschnitt 5.1] zu finden.

Die im höhenbalancierten CFG-Baum enthaltenen General Cluster Feature, die Sub-Cluster repräsentieren, unterscheiden sich von den bei BIRCH verwendeten Clustering-Feature-Vektoren durch eine Erweiterung der Parameter, die in Abschnitt 5.2.1 näher beschrieben wird.

Der CFG-Baum repräsentiert die Daten anwendungsunabhängig und muss auch im Nachhinein verändert werden können. Dies ist notwendig, da er jederzeit erstellt werden kann, wobei sich die Daten bis zum eigentlichen Clustering-Verfahren noch verändern können. Zum Aktualisieren müssen sowohl neue Tupel eingefügt als auch bestehende verändert und gelöscht werden können. Dies stellt durch das Additionstheorem der Clustering-Feature-Vektoren, das auch für die General Cluster Features angewendet werden kann, kein Problem dar. CHAD kann dadurch auch bestehende Bäume schrittweise erweitern. Im Zuge des Löschens eines Tupels kann ein General Cluster Feature auch wieder aus dem CFG-Baum entfernt werden. Zum Ändern eines Tupels wird das bestehende Tupel gelöscht und das neue an dieser Stelle eingefügt. Weitere Ausführungen zur ersten Phase von CHAD sind bei [Gol06, Abschnitt 5.2] zu finden.

5.2.1 General Cluster Features

General Cluster Features müssen für spezielle Anwendungen, die in der dritten Phase von CHAD ausgeführt werden und die nicht alle Attribute benötigen, erstellt werden. Sie sind im CFG-Baum, der in der ersten Phase von CHAD entsteht und eine anwendungsunabhängige Repräsentation der Daten darstellt, enthalten.

Die Clustering-Feature-Vektoren $CF = (N, \vec{LS}, SS)$ von BIRCH, siehe Abschnitt 2.2.3.2.1, erlauben die Reduzierung der Attribute innerhalb des Clustering-Feature-Vektors nicht, da die Informationen, die aus der Summe der Quadrate SS berechnet werden können, nicht ausreichend sind. Daher werden die Clustering-Feature-Vektoren erweitert und infolgedessen als General Cluster Feature bezeichnet.

Goller definiert ein solches General Cluster Feature folgendermaßen [Gol06, Definition 5.2]:
Ein General Cluster Feature $cfg = (N, \vec{LS}, \vec{SS}, \vec{BL}, \vec{TR})$ *ist ein Quintupel, das mehrere Tupel* $C \subset D$ *repräsentiert, wobei* N, $\vec{LS}$ *und* $\vec{SS}$ *die Anzahl der Tupel, die lineare Summe und die Summe der Quadrate für jede Dimension darstellt. Die Vektoren* $\vec{BL}$ *und* $\vec{TR}$ *beschreiben ein begrenzendes Rechteck, dass alle Tupel des General Cluster Features enthält, wobei der Vektor* $\vec{BL}$ *(bottom left) das Minimum aller Dimensionen und* $\vec{TR}$ *(top right) das Maximum aller Dimensionen speichert.*

Es ist zu beachten, dass ein General Cluster Feature, wegen des zusätzlichen Vektors für die Summe der Quadrate $\vec{SS}$, etwa doppelt so viel Speicherplatz wie ein Clustering-Feature-Vektor von BIRCH benötigt. Die zusätzlichen Vektoren $\vec{BL}$ und $\vec{TR}$ werden für eine bessere Selektion der Attribute aufgenommen.

5.3 Anwendungsabhängige Repräsentation der Daten - CFE-Baum

Da der CFG-Baum, der durch die erste Phase von CHAD entstanden ist, die gesamten Daten mit all ihren Attributen enthält, ist eine direkte Verwendung dieses CFG-Baums für eine spezifische Anwendung nicht immer möglich. Es dürfen nämlich nur jene Attribute im Baum enthalten sein, die die Anwendung benötigt. Außerdem sind nicht alle Attribute vergleichbar.

Aus dem CFG-Baum wird daher ein neuer Baum, CFE-Baum genannt, erzeugt, der die Attribute für die spezifische Anwendung enthält und anpasst. Für das Erzeugen dieses Baums wird eine Selektionsoperation, eine Projektionsoperation und eine Transformationsoperation durchgeführt, die in den Abschnitten 5.3.1, 5.3.2 und 5.3.3 ausführlich erläutert werden.

Der auf Grund des CFG-Baums erstellte Baum wird CFE-Baum genannt, weil er Extended Cluster Features enthält. Die im CFE-Baum enthaltene Beschreibung über mehrere Tupel entspricht weder dem General Cluster Feature noch dem Clustering-Feature-Vektor von BIRCH, und wird, wegen der Erweiterung um eine Berechnung, die für die Verwendung von EMAD notwendig ist, als Extended Cluster Feature bezeichnet.

Die zusätzliche Berechnung in den Extended Cluster Feature stellt das Vektoraußenprodukt Γ, das auf Grund der Informationen von den Einträgen im Blattknoten des CFE-Baums berechnet wird, dar. Die Definition des Extended Cluster Feature sowie eine Beschreibung ist in Abschnitt 5.3.4 zu finden.

5.3.1 Selektionsoperation

Die Selektionsoperation extrahiert all jene Tupel aus dem CFG-Baum, der alle Tupel D enthält, die der Bedingung p, die in der Menge aller Bedingungen P enthalten ist, entsprechen. Sie wird nach Goller folgendermaßen definiert [Gol06, Definition 5.3].

Die Selektion eines General Cluster Feature $cfg_* \in CFG$, der einen Sub-Cluster $C \subset D$ repräsentiert und die Selektionsbedingung $p \in P$ verwendet, ist eine Funktion $\zeta : CFG \times P \rightarrow CFG$, die ein General Cluster Feature $cfg_{**} \in CFG$ liefert, sodass

1. *die Anzahl der Tupel N_{**}, die vom General Cluster Feature cfg_{**} repräsentiert wird, gleich der Anzahl der Tupel vom General Cluster Feature cfg_* ist, der die Selektionsbedingung p erfüllt,*

$$N_{**} = \left| \left\{ \vec{x} \in C \middle| p(\vec{x}) \right\} \right|. \tag{18}$$

2. *die lineare Summe $\vec{LS}_{**}$ der Tupel, die vom General Cluster Feature cfg_{**} repräsentiert wird, gleich der Vektorsumme aller Tupel im General Cluster*

Feature cfg_* ist, der die Selektionsbedingung p erfüllt,

$$\overrightarrow{LS}_{**} = \sum_{\overrightarrow{x} \in C : p(\overrightarrow{x})} \overrightarrow{x} \, . \tag{19}$$

3. *der Vektor SS_{**} für jedes Attribut a ein Element $ss_{**}[a]$ besitzt, dass die Summe der Quadrate von allen Attributewerten $\overrightarrow{x}\,[a]$ der Tupel vom General Cluster Feature cfg_* speichert, der die Selektionsbedingung p erfüllt,*

$$ss_{**}[a] = \sum_{\overrightarrow{x} \in C : p(\overrightarrow{x})} (\overrightarrow{x}\,[a])^2. \tag{20}$$

4. *der Vektor BL_{**} für jedes Attribut a ein Element $bl_{**}[a]$ besitzt, dass das Minimum aller Attributewerte $\overrightarrow{x}\,[a]$ der Tupel vom General Cluster Feature cfg_* speichert, der die Selektionsbedingung p erfüllt,*

$$bl_{**}[a] = \min_{\overrightarrow{x} \in C : p(\overrightarrow{x})} (\overrightarrow{x}\,[a]). \tag{21}$$

5. *der Vektor TR_{**} für jedes Attribut a ein Element $tr_{**}[a]$ besitzt, dass das Maximum aller Attributewerte $\overrightarrow{x}\,[a]$ der Tupel vom General Cluster Feature cfg_* speichert, der die Selektionsbedingung p erfüllt,*

$$tr_{**}[a] = \max_{\overrightarrow{x} \in C : p(\overrightarrow{x})} (\overrightarrow{x}\,[a]). \tag{22}$$

Die Selektionsbedingung $p \in P$ ist eine logische Bedingung im Bezug auf die Tupel $\overrightarrow{x}$ in disjunktiver Normalform. Sie besteht aus mehreren Ausdrücken t_i und kann definiert werden als

$$p = \bigvee_{i=1}^{r} t_i, r \in \mathbb{N}. \tag{23}$$

Der Ausdruck t_i besteht aus mehreren Literalen und bestimmt sich als

$$t_i = \bigwedge_{j=1}^{s} x[a] \theta c_j, \theta \in \{<, \leq, =, >, \geq\}, s \in \mathbb{N}, a \in A, c_j \in \mathbb{R}, j \in \{1, 2, \ldots, s\}. \tag{24}$$

Dabei kennzeichnet $x[a]$ den Wert des Attributs a vom Tupel, θ einen Operator, und c_j ist eine benutzerdefinierte Konstante.

Genauere Ausführungen zur Selektionsoperation der General Cluster Features wie z.B. die Verwendung des begrenzenden Rechtecks, dass durch das Minimum $\vec{BL}$ und das Maximum $\vec{TR}$ bestimmt wird, oder die Bestimmung von neuen Einträgen in den Blattknoten sind bei [Gol06, Abschnitt 5.4] zu finden.

5.3.2 Projektionsoperation

Bei der Projektionsoperation werden die General Cluster Feature vereinfacht und jene Attribute entfernt, die nicht zu den selektierten Attributen für die spezifische Anwendung gehören. Die Projektion hat dabei auf die Anzahl der Punkte im CFE-Baum keinen Einfluss.

Zur Vereinfachung wird der Vektor mit den Summen der Quadrate $\vec{SS}$ im General Cluster Feature, durch dessen Summe SS ersetzt, da nach Abschluss der Selektion, dieser Vektor nicht mehr benötigt wird.

Die Projektion eines General Cluster Feature $cfg = (N, \vec{LS}, \vec{SS}, \vec{BL}, \vec{TR})$, mit einer Menge von Attributen $A_a \subset A$, ist nach Goller eine Funktion $\pi : CFG \times A \rightarrow CF$, die ein Cluster Feature $cf = (N', \vec{LS'}, SS')$ liefert, das gültig für die Anwendung ist, da die Attribute $A_a \subset A$ die relevanten Attribute darstellen [Gol06, Definition 5.4]. Daher gilt

$$N' = N, \tag{25}$$

$$\forall a \in A_a : \vec{LS}'\,[a] = \vec{LS}\,[a], \tag{26}$$

$$SS' = \sum_{a \in A_a} \vec{SS}'\,[a]. \tag{27}$$

Dabei repräsentiert A_a die selektierten Attribute der spezifischen Anwendung und A alle Attribute.

Für die Anwendung von EMAD in der dritten Phase von CHAD wurde diese Projektion noch um die Attribute $\vec{BL}$ und $\vec{TR}$ erweitert, sodass

$$\forall a \in A_a : \vec{BL}'\,[a] = \vec{BL}\,[a], \tag{28}$$

$$\forall a \in A_a : \vec{LS}'\,[TR] = \vec{TR}\,[a]. \tag{29}$$

Nach der Projektionsoperation stellt sich daher das im CFE-Baum enthaltene Extended Cluster Feature als $CFE = (N, \vec{LS}, SS, \vec{BL}, \vec{TR}, \Gamma)$ dar, wobei das Vekto-

raußenprodukt Γ in der zweiten Phase von CHAD noch zusätzlich berechnet wird, wie in Abschnitt 5.3.4 beschrieben.

Bei der Projektion kann auf zwei unterschiedliche Arten vorgegangen werden. Zum einen kann der gesamte Baum, zum anderen können nach Bedarf einzelne General Cluster Features projeziert werden. Bei CHAD wird die erste Variante, also die Projektion des gesamten Baums, durchgeführt.

5.3.3 Transformationsoperation

Nicht alle Attribute sind auf Grund ihrer Werte vergleichbar, wodurch eine Transformation notwendig ist. Dabei werden die Attribute auf Grund ihrer Mittelpunkte und ihrer Streuungen transformiert.

Nach Goller bestimmt CHAD den Mittelpunkt μ und die Streuung $s[a]$ von jedem Attribut und führt anschließend eine Z-Normalisierung durch, wodurch der berechnete Wert z einen Mittelpunkt von 0 und eine Streuung von 1 aufweist [Gol06, Abschnitt 5.6].

$$z := \frac{x[a] - \mu[a]}{s[a]} \tag{30}$$

Da für die Normalisierung die Mittelpunkte und die Streuung der Attribute bekannt sein müssen, wäre ein zweimaliges Einlesen der gesamten Daten notwendig. CHAD vermeidet dies, indem auf Grund einer kleinen Stichprobe der Daten zu Beginn die Mittelpunkte der Attribute und deren Streuung bestimmt werden. Mit Hilfe dieser wird die Normalisierung durchgeführt und der Baum aufgebaut. Während des Aufbaus werden die Mittelpunkte und die Streuung den tatsächlichen Attributwerten im Baum angeglichen.

5.3.4 Extended Cluster Feature

Das Extended Cluster Feature $CFE = (N, \vec{LS}, SS, \vec{BL}, \vec{TR}, \Gamma)$, das im CFE-Baum enthalten ist, beinhaltet neben der Anzahl N, der linearen Summe $\vec{LS}$, der Summe der Quadrate SS, dem Minimum $\vec{BL}$ und dem Maximum $\vec{TR}$ der Tupel noch das Vektoraußenprodukt Γ, das die Summe der quadrierten Tupel je Eintrag im

Blattknoten, wie in Formel 31 beschrieben, repräsentiert.

$$\Gamma = \sum_{\vec{d} \in CFE} \vec{d}\,\vec{d}^{T} \tag{31}$$

Da die genauen Punkte, die in den Einträgen des Blattknotens vom CFE-Baum enthalten sind, auf Grund der Aggregation nicht mehr bestimmt werden können, wird als Approximation für das Vektoraußenprodukt eines Eintrags die lineare Summe $\vec{LS}$ mit dem Mittelpunkt μ_c des Eintrags multipliziert. Das Vektoraußenprodukt wird in der zweiten Phase von CHAD gemäß Formel 32 bestimmt,

$$\Gamma = \vec{LS}\vec{\mu_c}, \tag{32}$$

wobei

$$\mu_c = \frac{\vec{LS}}{N}. \tag{33}$$

Durch Addition können die Vektoraußenprodukte der Einträge für den Blattknoten und den Vaterknoten vereinigt werden, wodurch die Aggregation im CFE-Baum ermöglicht wird.

5.4 K-Centroid Clustering (KC)

Die Weiterentwicklung des K-Centroid Clustering (KC) nach Fürst [Für04, Abschnitt 4.5] wird von EMAD als Initialisierung verwendet, und basiert im Wesentlichen auf dem Clustering-Verfahren k-means, siehe Abschnitt 2.2.3.4. Es wurde jedoch für die Verwendung durch CHAD angepasst, sodass die Datengrundlage von KC der CFE-Baum der zweiten Phase von CHAD ist.

KC verwaltet die Knoten bzw. Sub-Cluster in einer sekundären Speicherstruktur, ähnlich der Speicherstruktur EMADRef, die in dieser Arbeit in Abschnitt 6.1 beschrieben wird. Diese Struktur besteht aus einer k Elemente langen Liste, welche die Cluster repräsentiert. Jedes Element beinhaltet eine weitere Liste, die Referenzen auf die Sub-Cluster des Clusters enthält.

Durch die Verwendung des CFE-Baums als Datengrundlage wird bei KC zusätzlich ein Splittkriterium eingeführt, wodurch Sub-Cluster geteilt und deren Kindknoten

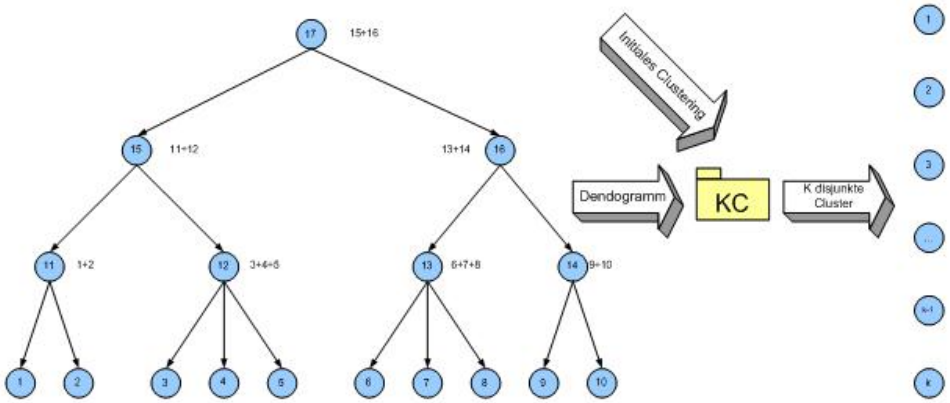

Abbildung 11: I/O des K-Centroids Clusterings [Für04, Abbildung 17]

einzeln geclustert werden können, was die Qualtiät des Clustering-Modells verbessern kann.

Genauso wie bei EMAD wirkt sich die Initialisierung auf das Ergebnis von KC aus. Nach Goller ist die beste von mehreren zufälligen Startlösungen eine gute Initialisierung für KC [Gol06, Abschnitt 5.10.2].

Im Folgenden wird der Ablauf von KC kurz beschrieben.

1. Für jeden Sub-Cluster, der durch eine Referenz in der sekundären Speicherstruktur enthalten ist, wird der nächstgelegene Mittelpunkt eines Clusters ermittelt und die Distanz dorthin berechnet.

2. Besitzt dieser Sub-Cluster Kindknoten so wird bestimmt, ob er gesplittet werden muss.

3. Ist ein Splitt notwendig, so wird die Referenz des Sub-Clusters in der sekundären Speicherstruktur durch die Referenzen der Kindknoten ersetzt. Anschließend wird mit Punkt 1 fortgesetzt, wobei dort der erste Kindknoten den aktuellen Sub-Cluster für die Berechnungen darstellt.

4. Ist kein Splitt notwendig, wird die Referenz des aktuellen Sub-Clusters zu jenem Cluster in der sekundären Speicherstruktur verschoben, der ihm am nächsten liegt. Dieser Schritt unterbleibt, wenn sich die Referenz des Sub-Clusters bereits im nächstgelegenen Cluster befindet.

5. Punkte 1-4 werden solange wiederholt, bis keine Verschiebung mehr durchgeführt wird.

Nachdem KC beendet ist, enthält die sekundäre Speicherstruktur das Clustering-Modell. Für den interessierten Leser ist eine ausführliche Beschreibung zum Ablauf von KC und zum Splittkriterium bei [Für04, Abschnitt 4.5] zu finden.

6 EMAD

Im diesem Kapitel wird der Algorithmus EMAD vorgestellt, der auf Grund von aggregierten Daten ein Clustering durchführt, und der im Rahmen dieser Diplomarbeit entwickelt wurde. EMAD basiert auf dem EM Algorithmus von Dempster, Laird und Rubin [DLR77], der für die Verwendung von aggregierten Daten angepasst wurde.

Die Datengrundlage von EMAD wird durch die erste und die zweite Phase von CHAD, siehe Kapitel 5, erstellt. Dabei bestimmt die erste Phase mit Hilfe eines hierarchischen Clustering-Verfahrens einen CFG-Baum, vergleiche Kapitel 5.2. Die zweite Phase, siehe Kapitel 5.3, selektiert, projeziert bzw. transformiert die Daten aus dem CFG-Baum und speichert sie im Baum mit den Extended Cluster Features, dem CFE-Baum, ab.

Das Ziel von EMAD ist das Erzeugen eines Clustering-Modells mit k disjunkten Clustern auf Grund des CFE-Baums in der dritten Phase von CHAD. Dabei werden nur die Knoten des CFE-Baumes, die Sub-Cluster repräsentieren, für das Clustering herangezogen. Da diese Knoten Daten in aggregierter Form enthalten, musste der EM Algorithmus angepasst werden.

Für das Zuordnen und Verschieben der Sub-Cluster zu den Clustern, das auf Grund des Clusterings notwendig ist, wird eine zweite Speicherstruktur, ein EMADRef Objekt, angelegt, welches für jeden Cluster eine Liste mit Referenzen enthält, die auf jene Knoten im CFE-Baum verweisen, die zum Cluster gehören. Der CFE-Baum bleibt beim Clustering mittels EMAD daher unverändert.

Wegen der hierarchischen Baumstruktur der aggregierten Daten muss in EMAD eine Splittbedingung eingeführt werden, die bestimmt, ob eine Referenz eines Knotens in der Speicherstruktur durch die Referenzen der Kindknoten zu ersetzen ist. Durch die niedrigere Aggregationsstufe der Kindknoten kann die Qualität des Clustering-Modells dadurch verbessert werden.

Der Aufbau des EMADRef Objekts, der Speicherstruktur für die Cluster, wird im folgenden Abschnitt näher ausgeführt. Darin wird im Speziellen auf das Vorgehen bei einem Splitt eines Knotens, siehe Abschnitt 6.1.1, und auf das Verschieben einer Referenz, vergleiche Abschnitt 6.1.2, eingegangen. Der genaue Ablauf des EMAD

Algorithmus wird in Abschnitt 6.2 beschrieben. Für die praktische Umsetzung von EMAD sind zusätzliche Berechnungen für die Kovarianzmatrizen, Inversen und Determinanten notwendig, die im Anhang A erläutert werden.

6.1 Speicherstruktur von EMAD - EMADRef Objekt

Da das Verschieben von Knoten in einem Baum einen hohen Rechenaufwand erfordert, wird aus Effizienzgründen bei EMAD eine alternative Speicherstruktur, ein sogenanntes EMADRef Objekt, eingesetzt. Bei diesem sind Referenzen auf die Knoten des CFE-Baums den Clustern zugeordnet. Der grundsätzliche Aufbau dieser Struktur entspricht jener, die Fürst als Sekundärstruktur bezeichnet [Für04, Abschnitt 4.5].

Ein EMADRef Objekt besteht aus k Elementen, welche die Ausprägungen der k disjunkten Cluster des Clustering-Modells enthalten. Deswegen werden in der Folge diese Elemente auch als Cluster-Elemente bezeichnet. Jedes Cluster-Element beinhaltet zum einen eine Liste mit Referenzen auf jene Sub-Cluster, die dem jeweiligen Cluster zugeordnet sind. Zum anderen enthält das Cluster-Element die Aggregation aller Sub-Cluster, die diesem Cluster angehören, in Form eines Extended Cluster Feature $CFE = (N, \vec{LS}, SS, \vec{BL}, \vec{TR}, \Gamma)$, näheres dazu siehe Abschnitt 5.3.4. Für die Aggregation werden alle Extended Cluster Features der Sub-Cluster vom Cluster aufsummiert.

In Abbildung 13 ist ein EMADRef Objekt dargestellt, das sich auf Grund des in Abbildung 12 dargestellten CFE-Baums ergeben kann. Das Cluster-Element C_1 repräsentiert dabei einen Cluster und beinhaltet die Summe der Extended Cluster Features der Sub-Cluster SC_{10} und SC_{11}, die sich auf Grund des Additionstheorems bestimmen lässt, sodass $CFE_{C_1} = CFE_{SC_{10}} + CFE_{SC_{11}}$. Das Cluster-Element C_1 besitzt außerdem eine Liste mit Referenzen $\{RSC_{10}, RSC_{11}\}$ auf jene Sub-Cluster im CFE-Baums, die dem Cluster zugeordnet sind.

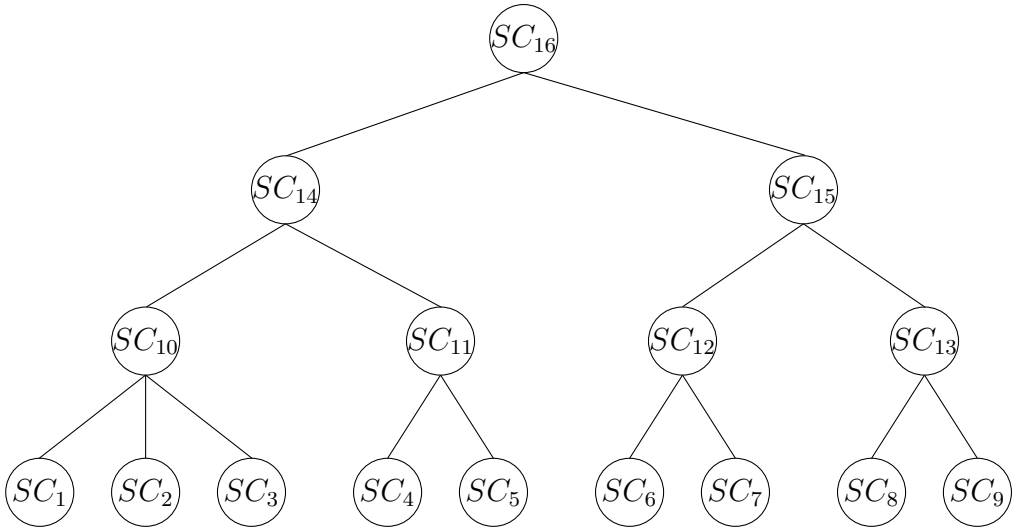

Abbildung 12: CFE-Baum

6.1.1 Splitt eines Sub-Clusters

Durch die Splittbedingung, näheres dazu siehe Abschnitt 6.3, kann es dazu kommen, dass ein Sub-Cluster während des Clusterings durch seine Kindknoten im EMADRef Objekt ersetzt werden muss. Dafür werden aus dem CFE-Baum die Kindknoten des aktuellen Sub-Cluster ausgelesen und deren Referenzen in die Speicherstruktur anstelle jener Referenz, die auf den aktuellen Sub-Cluster verweist, eingefügt. Die erste Referenz dieser Kindknoten wird anschließend als jene definiert, die auf den aktuellen Sub-Cluster verweist.

Abbildung 14 zeigt ein EMADRef Objekt, bei der die Referenz RSC_{10} auf den aktuellen Sub-Cluster SC_{10} verweist. Kommt es bei diesem Sub-Cluster zu einem Splitt, wird die Referenz RSC_{10} durch die Referenzen RSC_1, RSC_2 und RSC_3 ersetzt, die auf die Kindknoten des aktuellen Sub-Clusters verweisen, wie aus dem CFE-Baum aus Abbildung 12 ersichtlich.

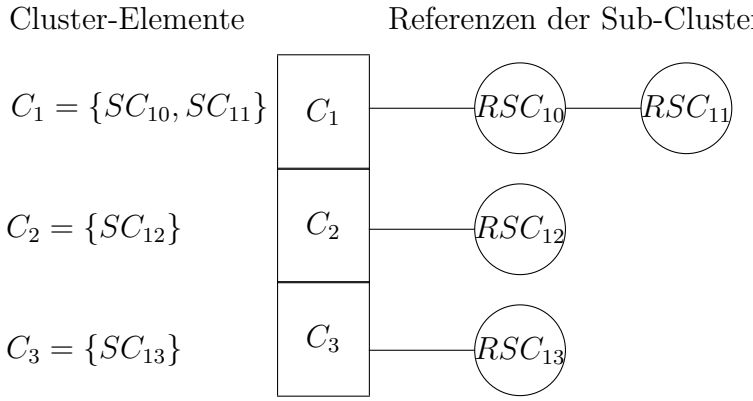

Abbildung 13: Aufbau eines EMADRef Objekts

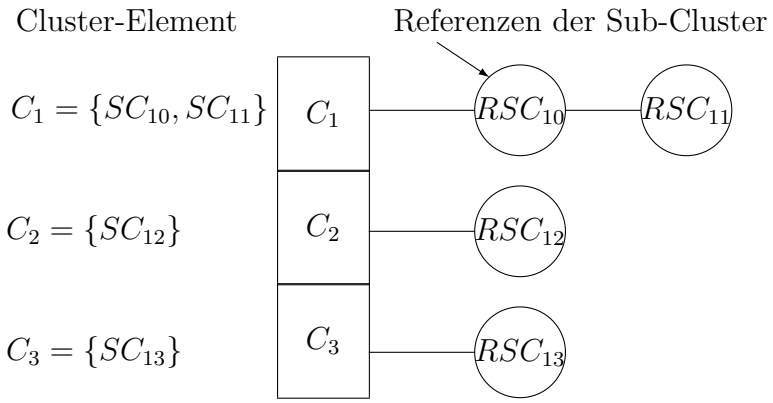

Abbildung 14: EMADRef Objekt vor dem Splitt des Knotens SC_{10}

In Abbildung 15 ist das EMADRef Objekt nach dem Splitt des Sub-Clusters SC_{10} dargestellt, wobei jene Referenz, die auf den ersten Kindknoten des Sub-Clusters SC_{10} verweist, hier RSC_1, die Referenz zum aktuellen Sub-Cluster im Clustering darstellt. Die Kindknoten werden nach dem Splitt als separate Sub-Cluster betrachtet und einzeln bei EMAD den Clustern zugeordnet.

Das Extended Cluster Feature von C_1 bleibt bei diesem Splitt unverändert, da die Summe der Extended Cluster Features der Kindknoten dem gelöschten Extended

Cluster Feature des Sub-Clusters SC_{10} entspricht,

$$CFE_{SC_{10}} = CFE_{SC_1} + CFE_{SC_2} + CFE_{SC_3},$$

sodass

$$CFE_{C_1} = CFE_{SC_1} + CFE_{SC_2} + CFE_{SC_3} + CFE_{SC_{11}}.$$

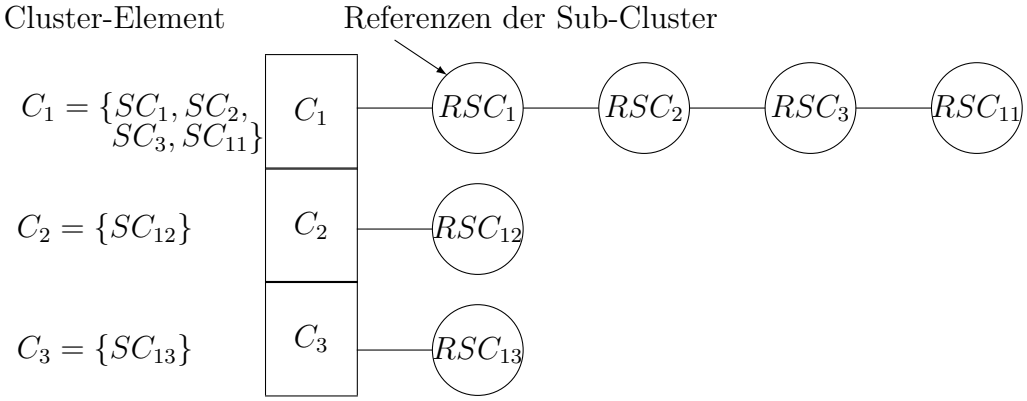

Abbildung 15: EMADRef Objekt nach dem Splitt des Knotens SC_{10}

6.1.2 Verschieben eines Sub-Clusters

In diesem Abschnitt wird auf das Verschieben der Referenzen von Sub-Cluster im EMADRef Objekt eingegangen, das auf Grund des Clusterings notwendig ist.

Bei EMAD werden die Sub-Cluster immer zu jenen Clustern verschoben, welche die höchste Wahrscheinlichkeit der Zugehörigkeit aufweisen, näheres dazu siehe Abschnitt 2.2.3.5. Dafür bestimmt EMAD diese Wahrscheinlichkeit für alle Cluster, und jenem Cluster mit der höchsten Wahrscheinlichkeit wird der Sub-Cluster zugeordnet.

Die Zuordnung zum Cluster kann auf Grund mehrerer Iterationen von EMAD mehrmals erfolgen, wobei sich der Cluster mit der höchsten Wahrscheinlichkeit der Clusterzugehörigkeit ändern kann. Es wird deshalb bei jeder Iteration geprüft, ob der Sub-Cluster dem Cluster mit der höchsten Wahrscheinlichkeit zugeordnet ist. Ist dies nicht der Fall, so muss der Sub-Cluster zu jenem Cluster mit der höchsten Wahrscheinlichkeit der Zugehörigkeit verschoben werden.

Die Veränderung des EMADRef Objektes der Verschiebung eines Sub-Clusters soll durch Abbildung 16 veranschaulicht werden. Dabei wird die Referenz RSC_2, die auf den Sub-Cluster SC_2 verweist, zuerst am Ende der Liste mit den Referenzen des Cluster-Elements C_2 angefügt. Anschließend wird es aus der Liste des Cluster-Elements C_1 gelöscht.

Durch die Addition bzw. Subtraktion des Extended Cluster Features CFE_{SC_2} mit den Extended Cluster Features C_2 bzw. C_1 werden die Ausprägungen der Cluster, die in den Cluster-Elementen gespeichert sind und die sich durch die Verschiebung verändert haben, wieder aktualisiert. Dadurch hat die Verschiebung eines Sub-Clusters Auswirkungen auf die betroffenen Cluster.

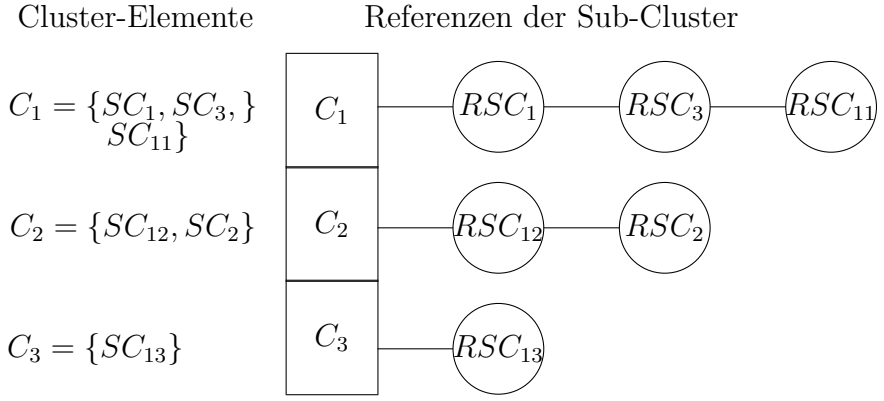

Abbildung 16: EMADRef Objekt nach dem Verschieben der Referenz SC_2

6.2 EMAD - Algorithmus

In diesem Abschnitt wird der Ablauf des Algorithmus von EMAD genau erläutert, der für das Clustering die Knoten des CFE-Baums verwendet, die im EMADRef Objekt verwaltet werden.

Im Folgenden werden die wichtigsten Zeilen von Algorithmus 5 auf Seite 66 genauer ausgeführt.

Zeile 1 Zu Beginn von EMAD wird die Initialisierung *init* von KC, siehe Abschnitt 5.4, in das EMADRef Objekt, hier *emadref* genannt, übernommen. Dabei werden k Cluster-Elemente $\{C_1, \ldots, C_i, \ldots, C_k\}$ angelegt, die zum

einen eine Liste mit Referenzen $\{RSC_1, \ldots, RSC_j, \ldots, RSC_l\}$ auf die Sub-Cluster $\{SC_1, \ldots, SC_j, \ldots, SC_l\}$ des CFE-Baumes *tree*, die dem Cluster bzw. Cluster-Element zugeordnet sind, enthält. Zum anderen beinhaltet jedes Cluster-Element ein Extended Cluster Feature CFE_i, das alle Sub-Cluster des Clusters aggregiert.

Zusätzlich werden bei der Initialisierung die Kovarianzmatrix Σ_i, deren Inverse Σ_i^{-1} und die Determinante $|\Sigma_i|$, sowie der Anteil der Tupel W_i und der Mittelpunkt μ_i für jedes Cluster-Element bestimmt, um die Gaußverteilung G_i berechnen zu können, die in einem Modell mit k Gaußverteilungen $M = \{G_1, \ldots, G_i, \ldots, G_k\}$ enthalten ist.

Für die Bestimmung der Kovarianzmatrix wird der Verschiebungssatz herangezogen. Mit Hilfe des Vektoraußenprodukts der linearen Summen $\vec{LS}_i$, des Vektoraußenprodukts von allen Tupeln Γ_i und der Anzahl der Tupel N_i von jedem Cluster-Element C_i kann diese bestimmt werden, sodass

$$\Sigma_i = \frac{1}{N_i}\left(\Gamma_i - \frac{1}{N_i}\vec{LS}_i\vec{LS}_i^{\,T}\right). \tag{34}$$

Für die Bestimmung des Anteils der Tupel W_i wird die Anzahl der Tupel eines Cluster-Elements N_i durch die Gesamtanzahl aller Tupel im Clustering N dividiert.

$$W_i = \frac{N_i}{N} \tag{35}$$

Der Mittelpunkt kann auf Grund des Extended Cluster Feature CFE_i im Cluster-Element bestimmt werden, da

$$\mu_i = \frac{\vec{LS}_i}{N_i}. \tag{36}$$

Die Determinante und Inverse der Kovarianzmatrix werden durch bestehende Rechenfunktionen, die in Anhang A näher definiert sind, bestimmt.

Zeile 2 Wird als Splittbedingung die Wahrscheinlichkeitsbedingung, vergleiche Abschnitt 6.3.2, herangezogen, so muss für das gesamte Clustering die inverse Normalverteilung auf Grund des Fehlerwerts *errval* bestimmt werden. Dafür wird der Algorithmus 10 aus Abschnitt 6.3.2.3 ausgeführt und des-

sen Ergebnis für die nachfolgende Bedingung in der Variable $errval_{PhiInv}$ gespeichert.

Zeile 4 Für jedes Cluster-Element wird der Radius r, die minimale Distanz zur Gleich-Wahrscheinlichkeitsdichte bestimmt, und im Vektor $\vec{r}$ gespeichert. Gemäß Algorithmus 3 werden für jeden Cluster des Cluster-Elements C_m die Punkte x_f ermittelt, die im Bezug zu allen anderen Clustern der Cluster-Elemente C_n die gleiche Wahrscheinlichkeit der Clusterzugehörigkeit aufweisen. Die geringste Distanz d_j zu diesen Punkten wird als Radius r für das Cluster-Element C_m gespeichert. Die Erläuterung und Definition der Berechnung der minimalen Distanz zur Gleich-Wahrscheinlichkeit ist in Abschnitt 6.3.3 zu finden.

Zeile 7 Auf jeden Sub-Cluster SC_j von allen Cluster-Elementen C_i wird eine Splittbedingung angewendet, die Extremwertbedingung oder die Wahrscheinlichkeitsbedingung. Trifft die Splittbedingung zu, so muss der Sub-Cluster nicht gesplittet werden. Das Ergebnis der Splittbedingung wird für die spätere Verwendung negiert und in der Variable *split* gespeichert.

Bei der Wahrscheinlichkeitsbedingung wird die Wahrscheinlichkeit, dass einzelne Tupel des Sub-Clusters SC_j außerhalb des Radius $\vec{r}\ [i]$ liegen, mit dem Ergebnis der inversen Normalverteilung $errval_{PhiInv}$ verglichen. Trifft die Bedingung, die im Algorithmus 4 dargestellt ist, zu, so muss kein Splitt vorgenommen werden. Der Radius r_μ bestimmt dabei die Distanz zwischen dem Mittelpunkt des Cluster-Elements μ_i und dem Mittelpunkt des Sub-Clusters μ_{SC_j}. Eine genaue Beschreibung der Wahrscheinlichkeitsbedingung ist in Abschnitt 6.3.2 gegeben.

Die Extremwertbedingung bestimmt die Notwendigkeit eines Splitts auf Grund der Extremwerte des Sub-Clusters SC_j, die einen Radius bestimmen, innerhalb diesem alle Tupel des Sub-Clusters liegen. Überschneidet sich der Radius $\vec{r}\ [i]$ des Cluster-Elements mit dem Radius der Extremwerte, so ist ein Splitt durchzuführen und die Extremwertbedingung trifft nicht zu.

Die Attribute des Minimums $\vec{BL}_{CFE_{SC_j}}$ und des Maximums $\vec{TR}_{CFE_{SC_j}}$, die im Extended Cluster Feature CFE_{SC_j} des Sub-Clusters enthalten sind, werden, gemäß Algorithmus 6, Zeile 3 bis 9, mit dem Mittelpunkt des Sub-

Clusters μ_{SC_j} verglichen. Jene Attributwerte vom Minimum oder Maximum, die die höchste Distanz zum Mittelpunkt ausweisen, werden im Vektor $\vec{cf_{extrem}}$ gespeichert.

Der Radius basierend auf den Extremwerten ist die Distanz zwischen dem Mittelpunkt des Sub-Clusters μ_{SC_j} und dem Vektor $\vec{cf_{extrem}}$. Die Extremwertbedingung trifft dann zu, wenn die Summe des Radius der Extremwerte und der Distanz r_μ, die zwischen dem Cluster-Element μ_i und dem Mittelpunkt des Sub-Clusters μ_{SC_j} liegt, den Radius $\vec{r}$ $[i]$ des Cluster-Elements C_i nicht überschreitet. Die Extremwertbedingung wird in Abschnitt 6.3.1 weiter ausgeführt.

Zeile 9 Ein Splitt des Sub-Clusters, der auf Grund der Splittbedingung aus Zeile 7 notwendig ist, kann nur ausgeführt werden, wenn dieser selbst Sub-Cluster besitzt. Trifft dies zu, so wird, wie in Abschnitt 6.1.1 erläutert, die Referenz des Sub-Clusters RSC_j durch jene der Kindknoten $\{RSC_{j_1}, \ldots, RSC_{j_l}\}$ im EMADRef Objekt ersetzt. Außerdem wird der erste Kindknoten als aktueller Sub-Cluster, $RSC_j \leftarrow RSC_{j_1}$, bestimmt. Der Algorithmus setzt dann mit dem Knoten, auf den die Referenz RSC_j verweist, in Zeile 7 fort, da diese auf den akutellen Sub-Cluster verweist.

Zeile 10 Besitzt ein Cluster-Element C_i nur eine einzige Referenz auf einen Sub-Cluster, der auf Grund der Splittbedingung nicht weiter geteilt werden muss, so unterbleiben die Berechnungen zur Ermittlung der Wahrscheinlichkeit der Zugehörigkeit zu einem Cluster und die Bestimmung einer Verschiebung, die in den Zeilen 11 bis 16 durchgeführt werden, da diese nicht notwendig sind.

Zeile 11 Für die Bestimmung der Wahrscheinlichkeit $P(SC_j|C_m)$, mit der ein Sub-Cluster SC_j in einer Gaußverteilung, die einen Cluster mit Hilfe des Cluster-Elements C_m beschreibt, enthalten ist, wird die Gauß'sche Dichtefunktion herangezogen. Diese wird, wie in Algorithmus 7 erläutert, für alle Cluster bestimmt und im Vektor $\vec{prob\_C}$ gespeichert.

Um die Dichtefunktion von EM, siehe Formel 11 aus Abschnitt 2.2.3.5, nicht auf einzelne Tupel sondern auf die Sub-Cluster anwenden zu können, wird für die Bestimmung der Wahrscheinlichkeit der Mittelpunkt μ_{SC_j} als Repräsentant des Sub-Clusters verwendet.

Zeile 12 Die Wahrscheinlichkeit $P(SC_j)$, mit der ein Sub-Cluster in einem Modell M mit k Gaußverteilungen vorkommt, wird durch die aus Zeile 11 bestimmten Wahrscheinlichkeiten, die in $\vec{prob\_C}$ gespeichert sind, und dem Anteil der Tupel W_i je Cluster-Element ermittelt, wie in Algorithmus 8 dargestellt.

Zeile 13 Die Wahrscheinlichkeit der Clusterzugehörigkeit $P(C_m|SC_j)$ eines Sub-Clusters SC_j wird für alle Gaußverteilungen, die auf den Cluster-Elementen C_m beruhen, wie in Algorithmus 9 erläutert, bestimmt. Sie ist von den berechneten Wahrscheinlichkeiten aus Zeile 11 und Zeile 12, sowie vom Anteil der Tupel W_i abhängig und wird im Vektor $\vec{prob\_emadref}$ mit der Länge k gespeichert.

Zeile 14 bis Zeile 16 Um festzustellen, ob eine Verschiebung des Sub-Clusters SC_j zu einem anderen Cluster-Element notwendig ist, wird jener Index des Vektors $\vec{prob\_emadref}$ bestimmt, der die höchste Wahrscheinlichkeit der Clusterzugehörigkeit enthält. Anschließend wird dieser Index m mit dem Index i verglichen, dem der Sub-Cluster SC_j derzeit zugeordnet ist. Stimmen diese nicht überein, werden beide Indizes und die Referenz des Sub-Clusters RSC_j zwischengespeichert.

Zeile 21 Nachdem für alle Sub-Cluster, die von den Cluster-Elementen referenziert werden, die Wahrscheinlichkeit der Clusterzugehörigkeit bestimmt und die Notwendigkeit einer Verschiebung in Zeile 14 überprüft wurde, werden mit Hilfe der gespeicherten Indizes und Referenzen aus Zeile 16 die Verschiebungen im EMADRef Objekt, wie in Abschnitt 6.1.2 beschrieben, durchgeführt.

Zeile 23 Da sich auf Grund der Verschiebungen aus Zeile 21 die Gaußverteilungen, die Cluster repräsentieren, verändert haben, werden diese für das neue Modell M' erneut bestimmt. Dazu werden auf Grund des *emadref'* Objekts die Kovarianzmatrix Σ'_i, deren Inverse $\Sigma_i^{-1'}$ und Determinante $|\Sigma'_i|$, sowie der Anteil der Tupel W'_i und der Mittelpunkt μ'_i je Cluster-Element, wie bereits bei der Initialisierung in Zeile 1, ermittelt.

Zeile 25 Die Güte des neuen Modells M' wird mit jener des vorherigen Modells M verglichen. Liegt deren Differenz unter einer festgelegten Gütedifferenz ϵ, so kann der Algorithmus beendet werden, ansonsten erfolgt eine weitere Iteration beginnend bei Zeile 4.

Die Güte eines Modells $M = \{G_1, ..., G_k\}$ wird auf Grund der Summe der logarithmierten Einzelwahrscheinlichkeiten der Sub-Cluster, die im EMADRef Objekt referenziert werden, bestimmt, sodass diese

$$E(M) = \sum_{SC_j \in emadref} \log(P(SC_j)) \tag{37}$$

ist.

Algorithm 3 Gleich-Wahrscheinlichkeitsdichte(EMADRef *emadref*)

Erstelle einen leeren Vektor $\vec{r}$ der Länge k;
for all $m \in \{1, \ldots, k\}$ **do**
  for all $n \in \{1, \ldots, k\}$ **do**
    if $(m <> n)$ **then**
      Bestimme bei *emadref* den Punkt x_f zwischen dem Cluster, der durch das Cluster-Element C_m definiert ist, und dem Cluster des Cluster-Elements C_n auf Grund der Gleichung
      $W_m \frac{1}{\sqrt{(2\pi)^d |\Sigma_m|}} e^{-\frac{1}{2}(x_f - \mu_m)^T (\Sigma_m^{-1})(x_f - \mu_m)} =$
      $W_n \frac{1}{\sqrt{(2\pi)^d |\Sigma_n|}} e^{-\frac{1}{2}(x_f - \mu_n)^T (\Sigma_n^{-1})(x_f - \mu_n)}$
      Bestimme die Distanz d_j zwischen x_f und μ_m;
      if $d_j < r[m]$ **then**
        $r[m] \leftarrow d_j$
      end if
    end if
  end for
end for
return $\vec{r}$;

Algorithm 4 Wahrscheinlichkeitsbedingung(Radius r, Radius r_μ, Sub-Cluster SC_j, inverse Normalverteilung $errval_{PhiInv}$)

return $\frac{r - r_\mu}{\sigma_{SC_j}} \geq errval_{PhiInv}$

Algorithm 5 EMAD (CFE-Baum *tree*, Initialisierung *init*, Fehlerwert *errval*, Gütedifferenz ϵ)

1: Erstelle ein EMADRef mit k Cluster-Elementen und initialisiere $emadref \leftarrow init$, sodass $emadref = \{C_1, \ldots, C_i, \ldots, C_k\}$, wobei $C_i = (CFE_i)$ und $C_i = \{SC_1, \ldots, SC_j, \ldots, SC_l\}$, die referenziert werden durch $C_i = \{RSC_1, \ldots, RSC_j, \ldots, RSC_l\}$; $\Sigma_i, \Sigma_i^{-1}, |\Sigma_i|, W_i, \mu_i$ für $G_i \in M$ bestimmen.
2: $errval_{PhiInv} \leftarrow \Phi^{-1}(1 - errval)$
3: **repeat**
4: $\vec{r} \leftarrow$ call Gleich-Wahrscheinlichkeitsdichte(emadref);
5: **for all** $C_i \in emadref$ **do**
6: **for all** $RSC_j \in C_i$ **do**
7: $split \leftarrow \neg$(call Extremwertbedingung($distance(\mu_{SC_j}, \mu_i)$, $CFE_{SC_j}, SC_j, r[i]$) bzw. Wahrscheinlichkeitsbedingung($r[i]$, $errval_{PhiInv}, distance(\mu_{SC_j}, \mu_i), SC_j,$);
8: **if** $(split \wedge \neg(SC_j = \{\}))$ **then**
9: Ersetze RSC_j in C_i durch die Referenzen der Kindknoten $\{RSC_{j_1}, \ldots, RSC_{j_l}\}$; $RSC_j \leftarrow RSC_{j_1}$;
10: **else if** $(|C_i| > 1)$ **then**
11: $\vec{prob\_C} \leftarrow$ call WahrscheinlichkeitProVerteilung($emadref, SC_j$)
12: $P(SC_j) \leftarrow$ call WahrscheinlichkeitProModell($\vec{prob\_C}$,$emadref$);
13: $\vec{prob\_emadref} \leftarrow$ call WahrscheinlichkeitderZugehörigkeit($\vec{prob\_C}$, $P(SC_j)$, $emadref$);
14: suche Index m in $\vec{prob\_emadref}$ für den $P(C_i|SC_j)$ maximal ist
15: **if** ($m \neq$ Index von C_i) **then**
16: Speichere die Indizes m, i und die Referenz RSC_j zwischen;
17: **end if**
18: **end if**
19: **end for**
20: **end for**
21: Verschiebe alle aus Zeile 16 gespeicherten RSC_j zu den Cluster-Elementen mit den Indizes m in $emadref$;
22: **for all** $C_i' \in emadref'$ **do**
23: $\Sigma_i' \leftarrow \frac{\Gamma_i - \frac{\vec{LS_i}\vec{LS_i^T}}{N_i}}{N_i}$; $W_i' \leftarrow \frac{N_i}{N}$; Bestimme $\Sigma_i^{-1'}$ und $|\Sigma_i|'$, sowie μ_i für G_i' im Modell M';
24: **end for**
25: **until** $|E(M') - E(M) < \epsilon|$
26: **return** $emadref$;

Algorithm 6 Extremwertbedingung(Radius r_μ, CFE CFE_{SC_j}, Sub-Cluster SC_j, Radius r,)

1: Erstelle einen leeren Vektor $\vec{cf_{extrem}}$ mit der Anzahl der Attribute $|A|$ als Länge;
2: Vergleich der Extremwerte $\vec{BL_{CFE_{SC_j}}}$ und $\vec{TR_{CFE_{SC_j}}}$ mit dem Mittelpunkt $\vec{\mu_{SC_j}}$;
3: **for all** $a \in \{1, \ldots, |A|\}$ **do**
4: **if** $distance(\vec{BL_{CFE_{SC_j}}}\ [a], \vec{\mu_{SC_j}}[a]) > distance(\vec{TR_{CFE_{SC_j}}}\ [a], \vec{\mu_{SC_j}}[a])$ **then**
5: $\vec{cf_{extrem}}[a] \leftarrow \vec{BL_{CFE_{SC_j}}}\ [a]$;
6: **else**
7: $\vec{cf_{extrem}}[a] \leftarrow \vec{TR_{CFE_{SC_j}}}\ [a]$;
8: **end if**
9: **end for**
10: **return** $(r_\mu + distance(\vec{cf_{extrem}}, \vec{\mu_{SC_j}})) \leq r$;

Algorithm 7 WahrscheinlichkeitProVerteilung(EMADRef *emadref*, Sub-Cluster SC_j)

1: Erstelle einen leeren Vektor $\vec{prob\_C}$ der Länge k;
2: **for all** $m \in \{1, \ldots, k\}$ **do**
3: $P(SC_j|C_m) \leftarrow \frac{1}{\sqrt{(2\pi)^d|\Sigma_m|}} e^{-\frac{1}{2}(\mu_{SC_j}-\mu_m)^T(\Sigma_m^{-1})(\mu_{SC_j}-\mu_m)}$
4: $\vec{prob\_C}\ [m] \leftarrow P(SC_j|C_m)$;
5: **end for**
6: **return** $\vec{prob\_C}$;

Algorithm 8 WahrscheinlichkeitProModell (WahrscheinlichkeitsdichteproVerteilung $\vec{prob\_C}$, EMADRef *emadref*)

1: **for all** $m \in \{1, \ldots, k\}$ **do**
2: $P(SC_j) \leftarrow \sum W_m\ \vec{prob\_C}\ [m]$
3: **end for**
4: **return** $P(SC_j)$;

Algorithm 9 WahrscheinlichkeitderZugehörigkeit (Wahrscheinlichkeitsdichteproverteilung $\vec{prob\_C}$, WahrscheinlichkeitProModell $P(SC_j)$, EMADRef *emadref*)

1: Erstelle einen leeren Vektor $\vec{prob\_emadref}$ der Länge k;
2: **for all** $m \in \{1, \ldots, k\}$ **do**
3: $P(C_m|SC_j) \leftarrow W_m \frac{\vec{prob\_C}[m]}{P(SC_j)}$;
4: $prob\_emadref[m] \leftarrow P(C_m|SC_j)$;
5: **end for**
6: **return** $\vec{prob\_emadref}$;

6.3 Splittbedingung

Die Splittbedingung bestimmt, ob ein Sub-Cluster in seine Kindknoten zerlegt (gesplittet) werden soll und diese anschließend bei EMAD einzeln den Clustern zugeordnet werden, wodurch sich die Genauigkeit des Clustering-Modells erhöhen kann. Durch diesen Splitt ist die Verwendung des CFB-Baums, in dem sich die Sub-Cluster befinden, als Datengrundlage für EMAD möglich. Die Splittbedingung kann zum einen mit Hilfe einer Wahrscheinlichkeitsbedingung und zum anderen mit einer Extremwertbedingung bestimmt werden.

Die Extremwertbedingung bestimmt mit Hilfe der Extremwerte eines Sub-Clusters jenen Radius, innerhalb dem sich alle Tupel des Sub-Clusters befinden. Liegt dieser Radius, ausgehend vom Mittelpunkt des Sub-Clusters, in jenem Bereich des Clusters, innerhalb dem die Tupel dem Cluster zugeordnet werden, so wird der Sub-Cluster nicht geteilt und es trifft die Extremwertbedingung zu. Weitere Erläuterungen sind in Abschnitt 6.3.1 zu finden.

Bei der Wahrscheinlichkeitsbedingung wird die Wahrscheinlichkeit berechnet, dass einzelne Tupel eines Sub-Clusters nicht zu jenem Cluster gehören, dem der gesamte Sub-Cluster zugeordnet ist. Liegt diese unterhalb eines Fehlerwerts, wird der gesamte Sub-Cluster einem Cluster zugewiesen, andernfalls muss der Sub-Cluster in seine Kindknoten zerlegt werden. In Abschnitt 6.3.2 wird diese Bedingung näher erläutert.

Im Prototyp von EMAD wurden zwei Splittbedingungen umgesetzt, und in den Abschnitten 7.5 und 7.6 sind deren Unterschiede beim Clustering im Bezug auf die Laufzeit und Qualität dargestellt.

In diesem Abschnitt wird auf die Unterschiede zwischen den zwei Arten von Splittbedingungen eingegangen. Die Definition der Extremwertbedingung ist in Abschnitt 6.3.1.1 zu finden, und die dafür notwendigen Berechnungen sind in Abschnitt 6.3.1.2 dargestellt. Auf die Wahrscheinlichkeitsbedingung wird in Abschnitt 6.3.2 eingegangen, und ihre Definition ist in Abschnitt 6.3.2.2 zu finden.

Der Abschnitt 6.3.2.3 geht auf die einzelnen Aspekte zur effizienten Berechnung der Wahrscheinlichkeitsbedingung ein, und in Abschnitt 6.3.2.4 werden die notwendigen Berechnungen für diese Splittbedingung näher erläutert.

Im abschließenden Abschnitt 6.3.3 wird die Berechnung des Radius r, der minimalen Distanz zur Gleich-Wahrscheinlichkeitsdichte, der für beide Splittbedingungen notwendig ist, dargelegt.

Die Extremwertbedingung und die Wahrscheinlichkeitsbedingung unterscheiden sich grundlegend in der Berechnung. Letztere bestimmt das Ergebnis durch eine Näherung mittels Standardisierter Normalverteilung und kann ab einem bestimmten Wert keine gültigen Lösungen mehr liefern. Dies tritt bei der Extremwertbedingung nicht ein, da sie von den existierenden Punkten im Sub-Cluster ausgeht. Im Gegensatz dazu verwendet die Wahrscheinlichkeitsbedingung eine Schätzung, die falsch sein kann.

Der Wert, ab dem die Wahrscheinlichkeitsbedingung nicht mehr gültig ist, liegt an jener Stelle, wo die Wahrscheinlichkeitsdichte des Clusters mit jener des Sub-Clusters ident ist und ab dem der Sub-Cluster „unterschätzt“ wird, d.h. die Dichtefunktion des Clusters jene des Sub-Clusters wieder übersteigt.

Liegt hingegen eine Überschätzung vor, so kann es zu einem vermeidbaren Splitt kommen. Der tatsächliche Fehler unterschreitet jedoch den maximal zulässigen Fehler, weshalb die Überschätzung vernachlässigt werden kann.

In Abbildung 17 werden die Funktionsgraphen von einem Cluster C_B und einem Sub-Cluster C_A mit ihren Mittelpunkten μ_{C_B} bzw. μ_{C_A} dargestellt. Der Wert, ab dem die Wahrscheinlichkeitsbedingung nicht mehr gültig ist, tritt hier bei x_2 ein.

Bei Punkt x_1 liegt ebenfalls eine Unterschätzung vor, die jedoch das Ergebnis der Wahrscheinlichkeitsbedingung nicht beeinflusst.

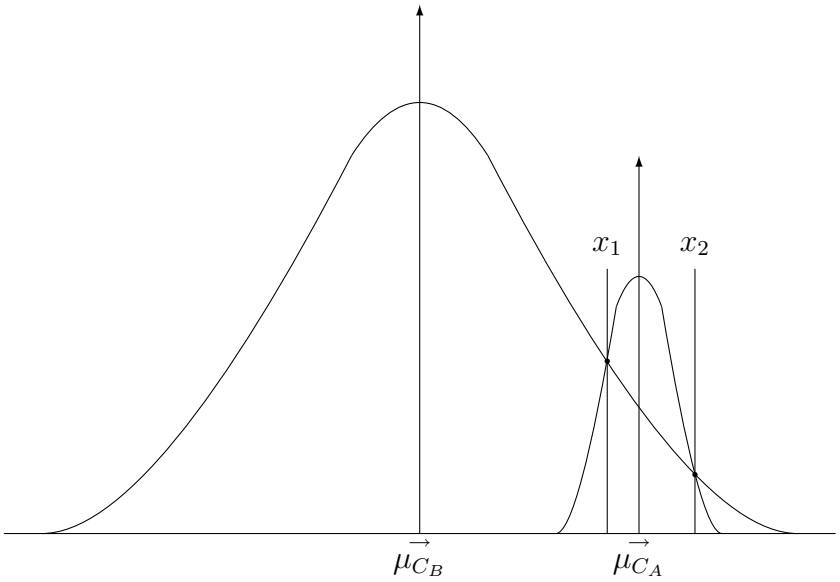

Abbildung 17: Wahrscheinlichkeitsdichte eines Clusters C_B und eines Sub-Clusters C_A

6.3.1 Extremwertbedingung

Bei der Extremwertbedingung, die in Abschnitt 6.3.1.1 definiert ist, wird für jeden Sub-Cluster SC_i geprüft, ob sich die Kugel um seinen Mittelpunkt $\vec{\mu_{sc}}$, mit dem Radius r_{sc}, in der sich alle Tupel des Sub-Clusters befinden, noch innerhalb der Kugel des Clusters mit dem Radius r liegt, da innerhalb dieser die Wahrscheinlichkeit für die Zugehörigkeit zu diesem Cluster größer ist als zu den anderen Clustern. Trifft dies zu, so muss der Sub-Cluster nicht gesplittet werden.

Für die Bestimmung des Radius r werden zwischen dem aktuellen Cluster und allen anderen Clustern jene Punkte berechnet, die eine gleich große Wahrscheinlichkeitsdichte besitzen. Der minimale Abstand zwischen dem Mittelpunkt $\vec{\mu_c}$ des aktuellen Clusters und dieser Punkte wird schließlich als Radius r bezeichnet. Somit gehören, ausgehend vom $\vec{\mu_c}$ des aktuellen Clusters, all jene Punkte, die sich innerhalb des Radius r befinden, mit einer höheren Wahrscheinlichkeit zum aktuellen Cluster als zu anderen Clustern. Die Berechnung des Radius r wird in Abschnitt 6.3.3 ausführlich ausgeführt.

Der Radius r_{sc} wird durch die Extremwerte des Sub-Clusters bestimmt, die im Extended Cluster Feature $CFE = (N, \vec{LS}, SS, \vec{BL}, \vec{TR}, \Gamma)$ des Sub-Clusters enthalten sind. Innerhalb des Radius r_{sc}, ausgehend vom Mittelpunkt $\vec{\mu_{sc}}$, sind darin alle Tupel des Sub-Clusters enthalten. Eine ausführliche Beschreibung zur Bestimmung des Radius r_{sc} ist in Abschnitt 6.3.1.2 zu finden.

6.3.1.1 Definition der Extremwertbedingung

Um die Extremwertbedingung zu bestimmen, ist die Distanz zwischen dem Mittelpunkt des Sub-Clusters $\vec{\mu_{sc}}$ und dem Mittelpunkt des Clusters $\vec{\mu_c}$ entscheidend, da diese für den Vergleich der Radien r und r_{sc} herangezogen wird. Diese Distanz wird in der Folge als r_μ bezeichnet und definiert sich folgendermaßen:

$$r_\mu = \mid \vec{\mu_c} - \vec{\mu_{sc}} \mid \tag{38}$$

Bei der Extremwertbedingung wird der Radius r mit der Summe der Distanzen r_{sc} und r_μ verglichen. Ist der Radius r hier größer, so ist die Extremwertbedingung erfüllt und es muss kein Splitt durchgeführt werden. Die Extremwertbedingung ist demnach definiert als:

$$(r_\mu + r_{sc}) \leq r \tag{39}$$

Der in Abbildung 18 dargestellte Sub-Cluster SC mit dem Radius r_{sc} liegt innerhalb des Cluster C mit dem Radius r, da die Summe aus der Distanz r_μ, die zwischen dem Mittelpunkt des Clusters $\vec{\mu_c}$ und dem Mittelpunkt des Sub-Cluster $\vec{\mu_{sc}}$ liegt, und der Distanz r_{sc} den Radius r nicht übersteigt.

Tritt der Fall ein, dass die Summe der Radien r_{sc} und r_μ den Radius r übersteigt, so schneiden sich die Radien r und r_{sc}. Für all jene Punkte des Sub-Clusters, die dabei außerhalb des Radius r liegen, ist dadurch die Zugehörigkeit zu dem aktuellen Cluster, dem der Sub-Cluster zugeordnet ist, nicht mehr gegeben. Die Extremwertbedingung trifft hier nicht zu, weshalb der Sub-Cluster gesplittet wird und seine Kindknoten einzeln den Clustern zugeordnet werden. Abbildung 19 zeigt einen Sub-Cluster, dessen Radius r_{sc} den Radius des Clusters r schneidet.

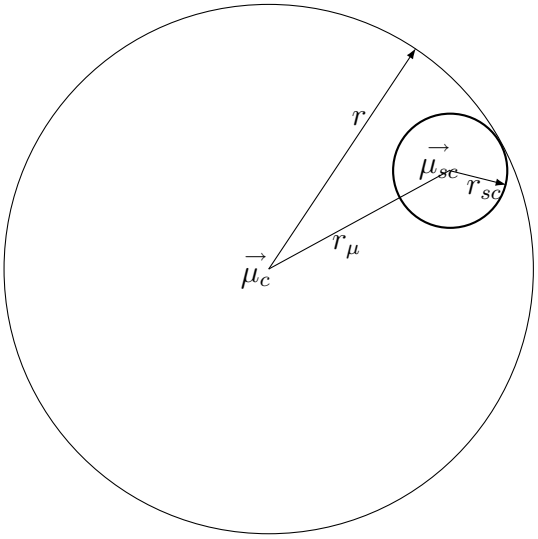

Abbildung 18: Sub-Cluster mit dem Radius r_{sc} liegt innerhalb des Cluster mit dem Radius r

6.3.1.2 Berechnung des Radius r_{sc} bei der Extremwertbedingung

Bei der Extremwertbedingung wird der Radius r_{sc} eines Sub-Clusters verwendet, innerhalb dem sich alle Punkte des Sub-Clusters befinden. Der Radius r_{sc} beruht auf den Extremwerten des Sub-Clusters, die im Extended Cluster Feature $CFE = (N, \vec{LS}, SS, \vec{BL}, \vec{TR}, \Gamma)$ enthalten sind. Diese sind das Maximum $\vec{TR}$ und das Minimum $\vec{BL}$ aller Attribute von allen Tupeln. Die Extended Cluster Features sind in Abschnitt 5.3.4 erläutert.

Aus dem Maximum $\vec{TR}$ und dem Minimum $\vec{BL}$ wird der Punkt $\vec{cf_{extrem}}$ bestimmt, dessen Attribute gegenüber dem Mittelpunkt des Sub-Clusters $\vec{\mu_{sc}}$ am weitesten entfernt sind. Bildet man eine Kugel um den Mittelpunkt $\vec{\mu_{sc}}$ mit dem Abstand von $\vec{cf_{extrem}}$ und dem Mittelpunkt $\vec{\mu_{sc}}$ als Radius, so kann man gewährleisten, dass alle Punkte des Sub-Clusters sich innerhalb dieser Kugel befinden.

Für die Bestimmung von $\vec{cf_{extrem}}$ wird jedes Attribut von $\vec{BL}$ und $\vec{TR}$ einzeln mit dem Attribut des Mittelpunkts $\vec{\mu_{sc}}$ verglichen, und dasjenige mit der größeren Distanz zum Mittelpunkt wird als Attribut in $\vec{cf_{extrem}}$ gespeichert. Dieser Vergleich

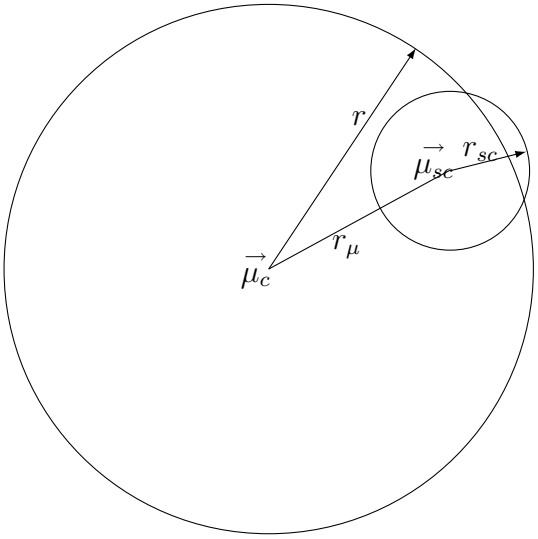

Abbildung 19: Sub-Cluster mit dem Radius r_{sc} liegt außerhalb des Cluster mit dem Radius r

wird für jedes Attribut a durchgeführt und kann folgendermaßen definiert werden:

$$\vec{cf_{extrem}}\left[a\right] = max(distanz(\vec{BL}\left[a\right], \vec{\mu_{sc}}\left[a\right]), distanz(\vec{TR}\left[a\right], \vec{\mu_{sc}}\left[a\right])) \tag{40}$$

Der Radius r_{sc} bestimmt sich aus der Distanz zwischen dem Punkt $\vec{cf_{extrem}}$ und dem Mittelpunkt des Sub-Clusters $\vec{\mu_{sc}}$ und wird wie folgt definiert.

$$r_{sc} = \left| \vec{cf_{extrem}} - \vec{\mu_{sc}} \right| \tag{41}$$

6.3.2 Wahrscheinlichkeitsbedingung

Bei der Wahrscheinlichkeitsbedingung wird im Vergleich zur Extremwertbedingung nicht das Minimum bzw. Maximum betrachtet, sondern für den Sub-Cluster SC_i wird geprüft, ob die Wahrscheinlichkeit einer Fehlklassifikation der Punkte im Sub-Cluster kleiner als der festgelegte maximale Fehlerwert $errval$ ist.

Der Sub-Cluster SC_i wird durch das Extended Cluster Feature $CFE = (N, \vec{LS}, SS, \vec{BL}, \vec{TR}, \Gamma)$ repräsentiert, wodurch nur die Anzahl der Punkte, die lineare Summe, die Summe der Quadrate, das Minimum, das Maximum und das Vektoraußenprodukt von den im Sub-Cluster enthaltenen Punkten bekannt sind. Damit

lassen sich wichtige Maßzahlen wie Centroid, Kompaktheits- und Distanzmaße für die Menge von Punkten im Sub-Cluster berechnen. Eine ausführliche Bestimmung der Lage der Punkte ist allerdings mit diesen Informationen nicht möglich, da durch die Aggregation nicht mehr die einzelnen Punkte der Daten sondern nur mehr deren Aggregate vorhanden sind.

Für die Wahrscheinlichkeitsbedingung wird wegen der fehlenden Information über die Verteilung eine Kernschätzung mit Gauß'schem Kern durchgeführt, die der Dichtefunktion der Standardisierten Normalverteilung entspricht.

Mit Hilfe des Extended Cluster Feature des Sub-Clusters lassen sich die notwendigen Informationen für die Standardisierte Normalverteilung berechnen. So kann der Mittelpunkt oder Centroid $\vec{\mu_{sc}}$ eines Sub-Clusters sowohl mit den Punkten $\vec{x_i}$, als auch mit der Linearsumme der Punkte $\vec{LS}$ berechnet werden, siehe Formel 36.

Die Standardabweichung σ_{sc} des Sub-Clusters kann durch das Extended Cluster Feature unter Zuhilfenahme des Verschiebungssatzes für Varianzen dargestellt werden als

$$\sigma_{sc} = \sqrt{\frac{\sum_{i=1}^{N}(\vec{X_i} - \vec{\mu_{sc}})^2}{N}} = \sqrt{\frac{SS}{N} - \vec{\mu_{sc}}^2}. \tag{42}$$

6.3.2.1 Exkurs: Wahrscheinlichkeitsrechnung — Standardisierte-Normalverteilung

Da die Verteilung der Tupel innerhalb eines Sub-Clusters nicht bekannt ist, wird eine Kernschätzung mit Gauß'schem Kern durchgeführt, die der Dichtefunktion der Standardisierten Normalverteilung entspricht. Diese Annahme soll in einem kurzen Exkurs über die Grundlagen der Standardisierten Normalverteilung belegt werden, der auf die Bücher von [Bas94] und [BH05, Kapitel 3] aufbaut.

Bei der Dichtefunktion der Gauß-Verteilung

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{43}$$

ist μ eine beliebige reelle Zahl und σ eine positive reelle Zahl. Die Gauß-Verteilung ist normalverteilt und wird mit $\mathrm{N}(\mu, \sigma)$ notiert. Die Kurve dieser Funktion, siehe

Abbildung 20, ist symmetrisch zu μ, besitzt im Punkt μ ihr Maximum und nähert sich für $x \to \infty$ und für $x \to -\infty$ null an. Dabei stellen μ den Erwartungswert und σ die Streuung dar.

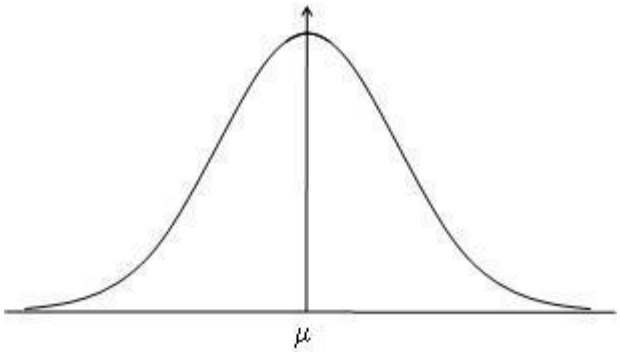

Abbildung 20: Graph der Dichtefunktion der Normalverteilung

Der Flächeninhalt $F(x)$ unter f entspricht zwischen $-\infty$ und $+\infty$ dem Wert 1 und wird mittels der folgenden Formel berechnet:

$$F(x) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dt \tag{44}$$

Durch den Transformationssatz, siehe Formel 46, kann die Berechnung der Verteilungsfunktion auf eine einzige Normalverteilung reduziert werden, der Standardisierten Normalverteilung, deren Dichtefunktion als φ, siehe Formel 45, deren Verteilungsfunktion als Φ bezeichent wird und die als Erwartungswert $\mu = 0$ und als Streuung $\sigma = 1$ besitzt.

$$\varphi_{0,1}(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \tag{45}$$

$$F(x) = \Phi\left(\frac{x-\mu}{\sigma}\right) \tag{46}$$

Die Transformation in eine standardisierte Verteilung wird durch die Standardisierung der Zufallsvariable X erreicht.

$$Z \equiv \frac{X - \mu}{\sigma} \tag{47}$$

Die Wahrscheinlichkeit einer Zufallsvariable kann auch mit einer Standardisierten Normalverteilung beschrieben werden, sodass

$$P(X \leq x) = P\left(\frac{X - \mu}{\sigma} \leq \frac{x - \mu}{\sigma}\right) = P\left(Z \leq \frac{x - \mu}{\sigma}\right) = \Phi\left(\frac{x - \mu}{\sigma}\right). \tag{48}$$

6.3.2.2 Definition der Wahrscheinlichkeitsbedingung

Bei der Wahrscheinlichkeitsbedingung wird festgelegt, dass die Wahrscheinlichkeit einer Fehlklassifikation $errClass(\vec{x}_i, SC_i)$ der Tupel $\vec{x}_i$ $(i = \{1, ..., N\})$ des Sub-Clusters SC_i nur kleiner als ein festgelegter maximaler Fehlerwert $errval$ sein darf. Dies gilt, falls

$$P(errClass(\vec{x}_i, SC_i)) < errval \tag{49}$$

ist, was dazu führt, dass kein Splitt durchgeführt werden muss.

Diese Wahrscheinlichkeitsbedingung muss für alle Sub-Cluster SC_i eines Clusters C ermittelt werden,

$$SC_i = \{SC_1, ..., SC_k\}. \tag{50}$$

Die Wahrscheinlichkeitsdichte der Tupel $\vec{x}_i$ im Cluster C wird in Formel 43 dargestellt, also die erwartete Konzentration der Tupel um den Mittelpunkt $\vec{\mu_{sc}}$. Die Flächen gleicher Dichte bilden (Hyper-)Kugeln um den Mittelpunkt. Dies gilt für alle Cluster bzw. Sub-Cluster und eben auch bei jenem Cluster mit dem Mittelpunkt $\vec{\mu_c}$, der an der Stelle $\vec{\mu_{sc}}$ die größte Wahrscheinlichkeitsdichte von allen Clustern aufweist. Daher wird der Sub-Cluster mit dem Mittelpunkt $\vec{\mu_{sc}}$ genau diesem Cluster zugeordnet.

Lässt sich nun eine Kugel um $\vec{\mu_{sc}}$ finden, innerhalb der alle Punkte eine höhere Wahrscheinlichkeit für den Mittelpunkt $\vec{\mu_c}$ als den Mittelpunkt eines anderen Clusters aufweisen, so kann für diese Tupel die Zugehörigkeit zu einem anderen Cluster

ausgeschlossen werden. Der größte Radius dieser Kugel, für den diese Bedingung gerade noch gilt wird als Radius r_m bezeichnet. Eine ausführlichere Erläuterung zur Bestimmung des Radius r_m ist in Abschnitt 6.3.2.4 zu finden.

Im Gegenzug dazu kann die Zugehörigkeit zu anderen Clustern bei den Tupeln x_i außerhalb der Kugel nicht ausgeschlossen werden. Diese Tupel erfüllen die folgende Bedingung:

$$| \vec{x_i} - \vec{\mu_{sc}} | > r_{sc} \tag{51}$$

Der Anteil der Tupel außerhalb der Kugel entspricht dabei der Wahrscheinlichkeit, dass ein Tupel außerhalb der Kugel liegt. Ist dieser Anteil kleiner als der Fehlerwert *errval*, dann ist die Wahrscheinlichkeitsbedingung erfüllt. Die Wahrscheinlichkeitsbedingung aus Formel 52 definiert sich daher:

$$P(| \vec{x_i} - \vec{\mu_{sc}} | > r_m) < errval \tag{52}$$

Um die Verteilungsfunktion Φ für die Berechnung der Wahrscheinlichkeit einer Fehlklassifikation anwenden zu können, müssen die Abstände der Punkte $\vec{x}_i$ zum Mittelpunkt des Sub-Clusters $\vec{\mu_{sc}}$, also $| \vec{x}_i - \vec{\mu_{sc}} |$, standardisiert werden.

$$\vec{z} = \frac{\vec{x} - \vec{\mu_{sc}}}{\sigma_{sc}} \tag{53}$$

Durch diese Transformation gilt für $\vec{z}$ die Standardisierte Normalverteilung,

$$\vec{z} \sim N(0, 1). \tag{54}$$

Damit ergibt sich für die Wahrscheinlichkeitsbedingung:

$$P(| \vec{z} | > \frac{r_m}{\sigma_{sc}}) < errval \tag{55}$$

Bezieht man nun die Verteilungsfunktion Φ in die Wahrscheinlichkeitsbedingung ein, so erhält man für die Wahrscheinlichkeit einer richtigen Klassifizierung von $\vec{x}_i$ folgendes:

$$\Phi\left(\frac{r_m}{\sigma_{sc}}\right) \geq (1 - errval) \tag{56}$$

Gemäß der in Formel 56 angeführten Bedingung muss für jeden Sub-Cluster SC_i die Wahrscheinlichkeit einer richtigen Klassifizierung berechnet werden. Da dies eine aufwendige Berechnung darstellt, wird die Bedingung um die inverse Verteilungsfunktion erweitert.

$$\frac{r_m}{\sigma_{sc}} \geq \Phi^{-1}(1 - errval) \tag{57}$$

Die Effizienz der Berechnung wird dadurch gesteigert, da die inverse Verteilungsfunktion von $(1 - errval)$, obwohl deren Berechnung ebenfalls sehr aufwendig ist, nur einmal für das gesamte Clustering berechnet werden muss. Dies begründet sich dadurch, dass der definierte Fehlerwert *errval* für alle Sub-Cluster und Cluster, gemäß Formel 57, konstant bleibt. Die inverse Verteilungsfunktion der Normalverteilung, sowie der in dieser Arbeit verwendete Algorithmus zur Berechnung eben dieser Funktion, werden im folgenden Abschnitt erläutert.

6.3.2.3 Inverse Verteilungsfunktion der Normalverteilung

Die inverse Normalverteilung, auch bekannt als inverse Gauß-Verteilung, wurde erstmal im Jahre 1915 von Schrödinger und Smoluchowsky verwendet. Sie ist eine nicht-lineare Funktion, für die keine geschlossene Lösung existiert. Die Funktion ist stetig, monoton steigend, unendlich differenzierbar und bildet das offene Intervall (0,1) auf dem Wertebereich der reellen Zahlen ab. Der Graph dieser Funktion ist in Abbildung 21 skizziert:

Die Berechnung der inversen Gauß-Verteilung beansprucht viel Zeit, weshalb man in der Statistik oft Tabellen verwendet, welche die Ergebnisse der Verteilung innerhalb eines bestimmten Intervall enthalten. Die Genauigkeit dieser Tabellen hängt vom Algorithmus des Nährungsverfahrens ab, mit dem sie erstellt werden. Die Laufzeit steht dabei in einem direkten Zusammenhang zur Genauigkeit bzw. der Fehlertoleranz. Im folgenden Abschnitt wird jener Algorithmus vorgestellt, der für die Berechnung der inversen Gaußverteilung bei der Umsetzung des Prototyps verwendet wurde.

6.3.2.3.1 Algorithmus zur Berechnung der inversen Gaußverteilung

Der verwendete Algorithmus ist von Peter J. Acklam, der diesen auf seiner Homepage [Ack06] ohne rechtliche Beschränkungen in mehreren Programmiersprachen zur Verfügung stellt und der in diesem Abschnitt näher vorgestellt wird.

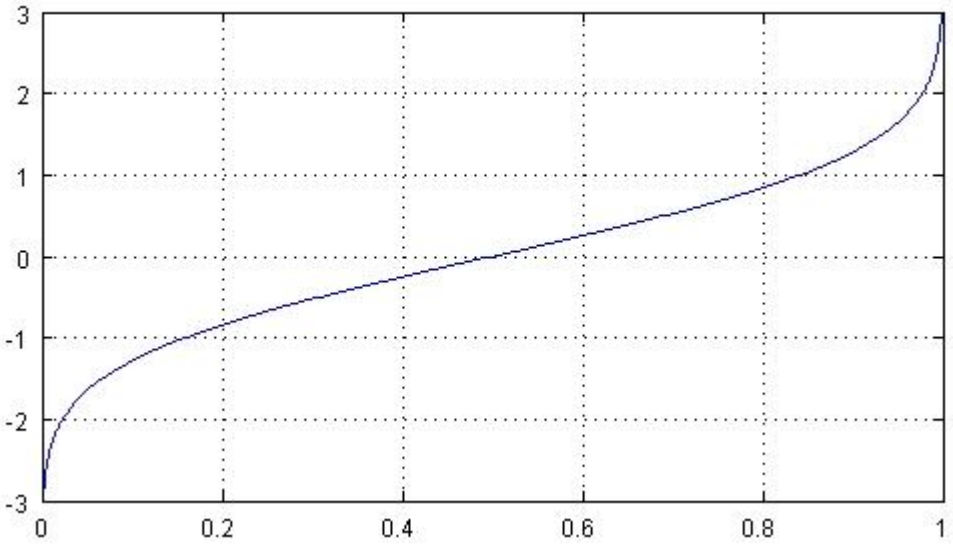

Abbildung 21: Umkehrfunktion der normierten Verteilungsfunktion der Normalverteilung [Ack06]

Der relative Fehler liegt beim Algorithmus von Acklam bei weniger als $1.50 * 10^{-9}$, was den in dieser Arbeit gestellten Ansprüchen vollkommen genügt und deshalb bei der Wahrscheinlichkeitsbedingung verwendet wird.

Acklam nutzt bei seinem Algorithmus die Tatsache aus, dass der gespiegelte Wert der Inversen der Normalverteilung, $\Phi^{-1}(p)$, als $\Phi^{-1}(p+1/2)$ definiert werden kann. Daher geht der Algorithmus grundsätzlich von zwei unabhängigen relationalen Näherungen aus, wobei eine für die zentrale Region und die andere für die Ränder der Dichtefunktion verwendet wird. Die zentrale Region gilt, wenn $0.02425 \leq p \leq 1-0.02425 = 0.97575$ ist, wofur eine direkte rationale Näherung durchgeführt wird. Der Bereich der Ränder gilt, wenn $p < 0.02425$ oder $p > 1 - 0.02425 = 0.97575$ ist. Hierbei wird zuerst eine nichtlineare Transformation durchgeführt, um die Funktion der Linearität anzunähern, bevor die Näherung durchgeführt wird.

Der Algorithmus 10 nach [Ack06] soll die Berechnung der Inversen Normalverteilung verdeutlichen.

Algorithm 10 PhiInv (Wert p)

{Randpunkte definieren:}
$p\_low \leftarrow 0.02425$;
$p\_high \leftarrow (1 - p\_low)$;
{Rationale Näherung im unteren Rand:}
if $0 < p < p\_low$ **then**
  $q \leftarrow \sqrt{-2 * \log(p)}$
  $x \leftarrow \frac{((((((c(1)*q+c(2))*q+c(3))*q+c(4))*q+c(5))*q+c(6))}{((((d(1)*q+d(2))*q+d(3))*q+d(4))*q+1)}$
end if
{Rationale Näherung in der zentralen Region:}
if $p\_low <= p <= p\_high$ **then**
  $q \leftarrow (p - 0.5)$
  $r \leftarrow (q^2)$
  $x \leftarrow \frac{((((((a(1)*r+a(2))*r+a(3))*r+a(4))*r+a(5))*r+a(6))*q}{((((((b(1)*r+b(2))*r+b(3))*r+b(4))*r+b(5))*r+1)}$
end if
{Rationale Näherung im oberen Rand:}
if $p\_high < p < 1$ **then**
  $q \leftarrow \sqrt{-2 * log(1 - p)}$
  $x \leftarrow \frac{((((((c(1)*q+c(2))*q+c(3))*q+c(4))*q+c(5))*q+c(6))}{((((d(1)*q+d(2))*q+d(3))*q+d(4))*q+1)}$
end if
return x

Eine Erhöhung der Genauigkeit bzw. eine Senkung der Fehlertoleranz ist mit einer Fehlerfunktion durchführbar. Die Implementierung einer solchen Fehlerfunktion wird von W. J. Cody [Cod06] zur Verfügung gestellt. Diese Präzisierung ist auf Grund der Aufgabenstellung für die Umsetzung des Prototypen nicht erforderlich und wird deshalb nicht miteinbezogen.

Weiterführende Informationen zum Algorithmus von Acklam, wie Fehlerplots und Referenzen zu anderen Algorithmen, sind auf dessen Homepage [Ack06] zu finden.

6.3.2.4 Berechnung des Radius r_m bei der Wahrscheinlichkeitsbedingung

Bei der Wahrscheinlichkeitsbedingung wird der Radius r_m mit dem Abstand zwischen dem Punkt $\vec{x_i}$ und dem Mitteplunkt des Sub-Clusters $\vec{\mu_{sc}}$ verglichen und so bestimmt, ob eine Fehlklassifikation des Punktes tatsächlich vorliegt.

Der Radius r_m ist dabei abhängig von der Lage des Sub-Clusters SC_i innerhalb seines zugehörigen Clusters und dem Radius r von diesem Cluster. Für die Beschreibung und Berechnung des Radius r wird auf den Abschnitt 6.3.3 verwiesen.

Abbildung 22 zeigt einen Cluster und die Lage des Sub-Clusters SC_i mit dem Mittelpunkt $\vec{\mu_{sc}}$ und dem Radius r_{sc} im zweidimensionalen Raum. Durch die Lage des Sub-Clusters innerhalb des Clusters und dem Radius r_{sc} kann der Fall eintreten, dass sich die Radien r und r_{sc} schneiden. Dadurch haben jene Punkte des Sub-Clusters, die sich außerhalb des Radius r befinden, eine höhere Wahrscheinlichkeit der Zugehörigkeit zu einem anderen Cluster als zu dem derzeitigen.

Bei der Wahrscheinlichkeitsbedingung wird nur jene Wahrscheinlichkeit bestimmt, die den Anteil der Punkte angibt, die richtig zugeordnet worden sind. Dafür wird der Radius r_m verwendet, welcher der maximale Radius des Sub-Cluster SC_i ist, der sich nicht mit dem Radius r schneidet. Alle innerhalb des Radius r_m gelegenen Punkte besitzen daher für den aktuellen Cluster die höchste Wahrscheinlichkeit der Zugehörigkeit.

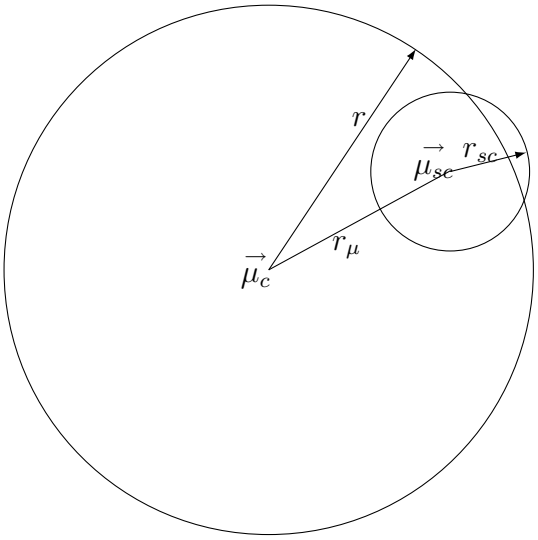

Abbildung 22: Cluster mit Sub-Cluster

Die in Abbildung 22 veranschaulichte Distanz r_μ, wie in Formel 38 definiert, ist der Abstand zwischen dem Mittelpunkt des Clusters $\vec{\mu_c}$ und dem Mittelpunkt des Sub-Clusters $\vec{\mu_{sc}}$.

Für die Berechnung des Radius r_m wird die Differenz zwischen den Radien r_μ und r des Clusters ermittelt.

$$r_m = r - r_\mu \tag{58}$$

bzw.

$$r_m = r - \mid \vec{\mu_c} - \vec{\mu_{sc}} \mid. \tag{59}$$

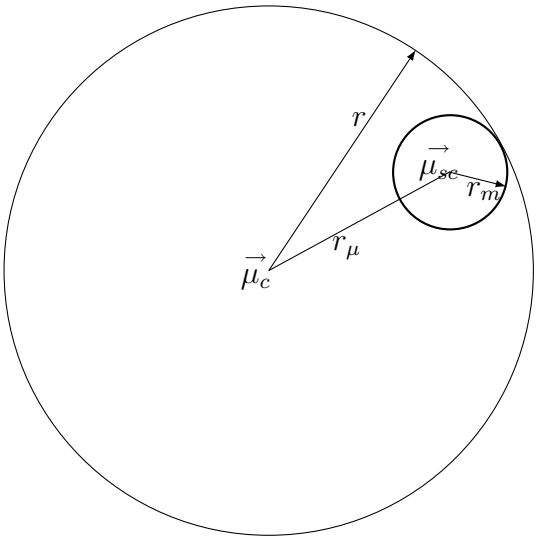

Abbildung 23: Cluster mit Sub-Cluster und berechnetem Radius r_m

Der Radius r_m stellt einen weiteren Radius für den Sub-Cluster SC_i dar, der ohne den tatsächlichen Radius r_{sc} des Sub-Clusters berechnet wird. Dieser neue Radius kann bei der Wahrscheinlichkeitsbedingung in Abschnitt 6.3.2.2 dazu führen, dass Punkte auch als falsch-zugeordnet definiert werden, die sich innerhalb des Radius r befinden und damit als richtig-zugeordnet gelten sollten. Dies ist der Fall wenn gilt:

$$r_m < r_{sc} \tag{60}$$

Trifft die Ungleichung aus Formel 60 zu, so gelten all jene Punkte, die sich zwischen den Radien r_m und r_{sc} befinden, als falsch-zugeordnet und dazu zählen auch jene, die innerhalb des Radius r gelegen sind. Zur Veranschaulichung zeigt Abbildung 24 jenen Bereich im Sub-Cluster schwarz, in dem die Punkte als richtig-zugeordnet gelten. Die Punkte im nicht schwarz markierten Bereich des Sub-Clusters gelten bei der Wahrscheinlichkeitsbedingung als falsch-zugeordnet.

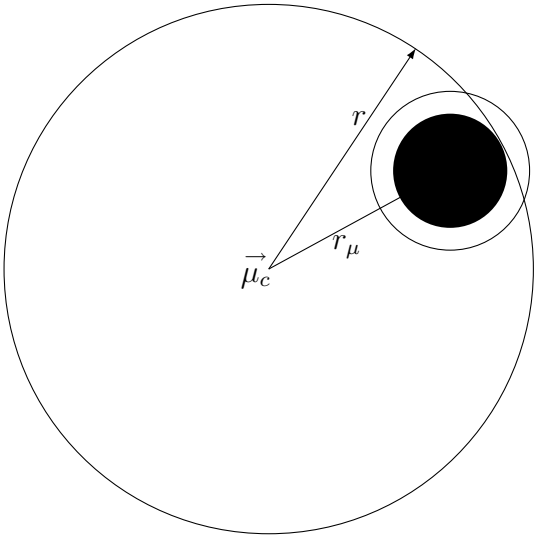

Abbildung 24: Cluster und Sub-Cluster mit markierter Richtig - und Falschzuordnung

Der Radius r_m in der Wahrscheinlichkeitbedingung, der eine mögliche falsche Zuordnung von Punkten des Sub-Clusters zu den Clustern nach sich zieht, wird trotzdem verwendet, da die genaue Berechnung des Anteils jener Punkte, die mit einer höheren Wahrscheinlichkeit zu einem anderen Cluster gehören, sehr aufwendig ist. Da man diese Berechnung bei EMAD ikn-mal ausführen müsste - wobei i für die Anzahl der Iterationen, k für die Anzahl der Cluster und n für die durchschnittliche Anzahl von Sub-Cluster in einem Cluster steht - könnte es zu einer negativen Beeinträchtigung der Effizienz kommen. Die zusätzliche Rechenzeit rechtfertigt den minimalen Genauigkeitsgewinn nicht, da auch mit einem angepassten Fehlerwert $errval$ bei der Wahrscheinlichkeitsbedingung dieser Fehler gemildert werden kann. Außerdem wird durch einen Splitt des Sub-Clusters dieser Fehler ebenfalls verkleinert, da die Radien der Sub-Cluster kleiner und diese damit auch exakter werden.

Es kann auch der Fall eintreten, dass ein Sub-Cluster mit dem Radius r_{sc} komplett innerhalb des Radius r liegt, und damit nach der Berechnung von r_m gilt:

$$r_m \geq r_{sc} \tag{61}$$

Bei der Wahrscheinlichkeitsbedingung mit der Ungleichung 61 kann sowohl mit dem berechneten Radius r_m als auch mit dem Radius r_{sc} das gleiche Ergebnis erzielt werden, und es tritt somit kein Fehler auf, da kein einziger Punkt des Sub-Clusters mit einer höheren Wahrscheinlichkeit zu einem anderen Cluster zugeordnet gehört.

6.3.3 Berechnung des Radius r - der minimalen Distanz zur Gleich-Wahrscheinlichkeitsdichte

Bei den Splittbedingungen wird der Radius r eines Sub-Clusters verwendet um zu bestimmen, ob die im Sub-Cluster enthaltenen Punkte einem anderen Cluster zugeordnet werden sollen. Der Radius r stellt dabei die geringste Distanz, ausgehend vom Mittelpunkt $\vec{\mu_{C_A}}$ des Clusters C_A, dar, bei dem die Wahrscheinlichkeitsdichte zu einem der anderen Cluster C_B ($B = \{1, ..., k|\neg A\}$) gleich groß ist, wie in Abbildung 25 dargestellt. Auf der Linie zwischen dem Cluster C_A zu allen anderen Clustern C_B gibt es einen fiktiven Punkt, für den die Wahrscheinlichkeit der Zugehörigkeit gleich groß ist. Im weiteren Verlauf wird dieser Punkt als x_f bezeichnet und dessen Bestimmung wird in Abschnitt 6.3.3.1 näher ausgeführt.

Der Abstand zu den unterschiedlichen x_f zwischen dem Cluster C_A und allen anderen Clustern C_B ist unterschiedlich groß, da die Lage der gleichen Wahrscheinlichkeitsdichte zwischen zwei Clustern von deren Eigenschaften, insbesondere deren Kovarianzmatrizen, abhängt.

Bei der Bestimmung des Radius r ist immer nur jenes x_f relevant, das den geringsten Abstand zu dem Mittelpunkt $\vec{\mu_{C_A}}$ des Clusters C_A besitzt.

Der gesamte Bereich um einen Cluster, der die Clusterzugehörigkeit begrenzt, kann unterschiedliche Formen aufweisen. In den meisten Fällen ist es ein Ellipsoid [Wei06].

Es kann allerdings auch der Fall eintreten, dass zwischen zwei Clustern auch zwei Punkte existieren, bei denen die Wahrscheinlichkeitsdichte gleich groß ist. Dies tritt dann ein, wie in Abbildung 26 dargestellt, wenn der Bereich der Clusterzugehörigkeit eines Clusters einen anderen Cluster einschließt.

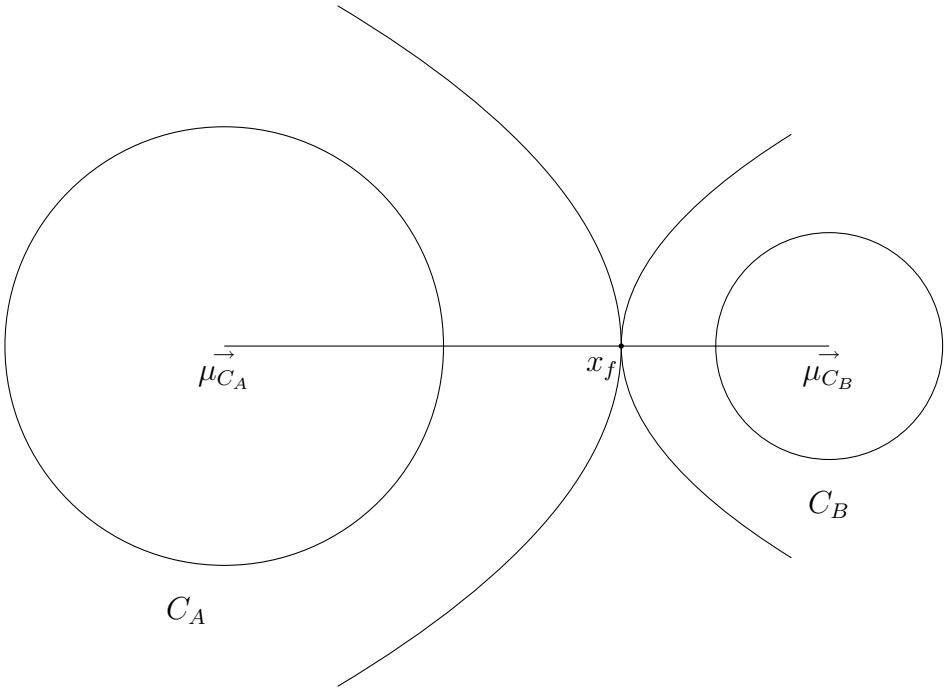

Abbildung 25: Cluster C_A und C_B mit dem Punkt der Gleich-Wahrscheinlichkeitsdichte x_f

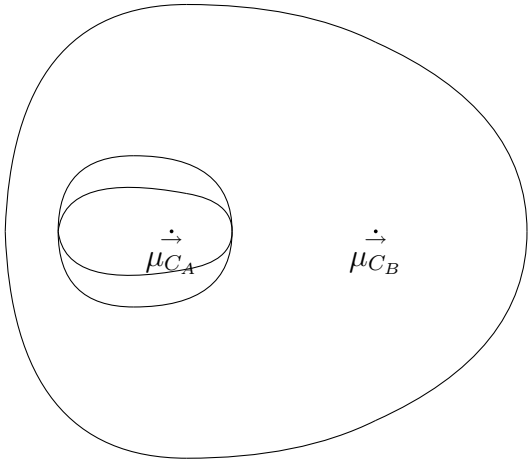

Abbildung 26: Cluster C_B umschließt mit dem Bereich der Clusterzugehörigkeit Cluster C_A

Stellt man diesen Bereich als Dichtefunktion dar, wie in Abbildung 27, so zeigen sich ebenfalls die beiden Schnittpunkte, hier x_1 und x_2, die sich durch das

Überschneiden der Funktionsgraphen der beiden Cluster ergeben. Der Graph von Cluster C_A liegt dabei komplett innerhalb des Graphen von Cluster C_B.

Für die Bestimmung von x_f ist jedoch nur jener Schnittpunkt relevant, der zwischen dem Mittelpunkt $\vec{\mu_{C_A}}$ und dem Mittelpunkt $\vec{\mu_{C_B}}$ liegt, da die Distanz zu beiden Mittelpunkten minimal sein muss.

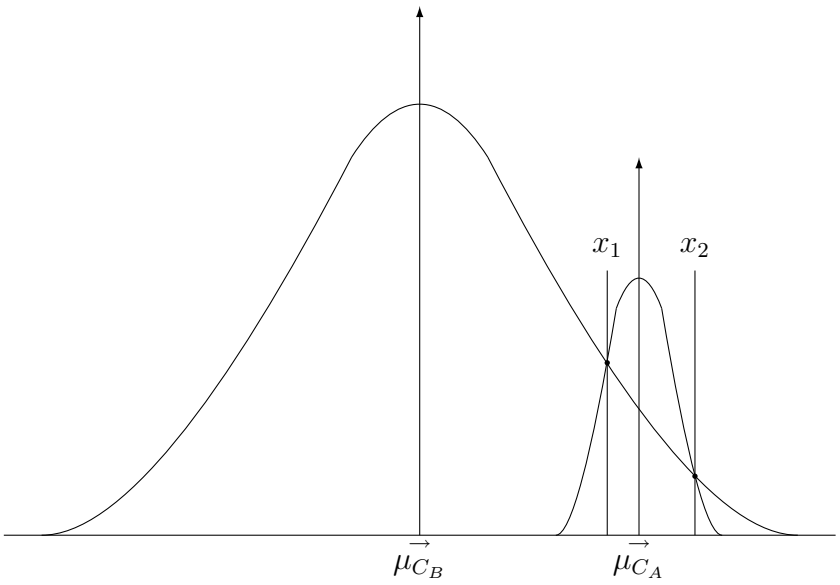

Abbildung 27: Cluster mit ihren unterschiedlichen Wahrscheinlichkeitsdichten

Einen Sonderfall stellt die Situation dar, wenn die beiden Cluster, zwischen denen das x_f bestimmt werden soll, bis auf den Mittelpunkt identisch sind. In diesem Fall ist der Bereich für die Clusterzugehörigkeit bei beiden Clustern kein Ellipsoid. Eine Gerade, die sich genau zwischen den Mittelpunkten $\vec{\mu_{C_A}}$ und $\vec{\mu_{C_B}}$ der Cluster befindet, bestimmt die Zuordnung der Punkte zu den Clustern. In diesem Sonderfall sind k-means und EM identisch, da gleiche Wahrscheinlichkeiten gleichen Distanzen entsprechen.

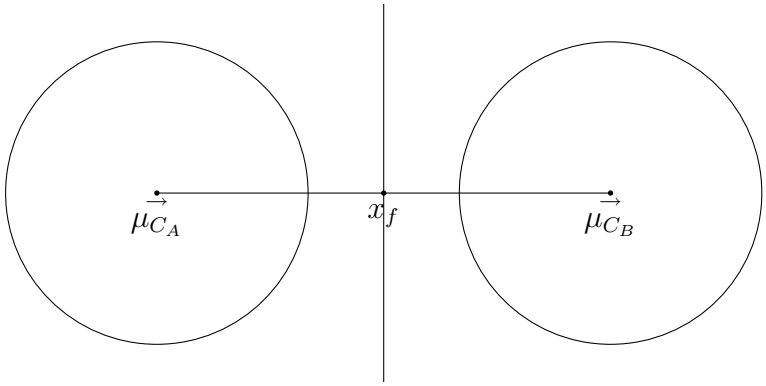

Abbildung 28: Identische Cluster mit der Gleich-Wahrscheinlichkeitsdichte x_f

6.3.3.1 Bestimmung des Punktes x_f

Für die Bestimmung des Punktes x_f wird die Wahrscheinlichkeit für die Clusterzugehörigkeit von zwei Clustern, hier Cluster C_1 und C_2, mit den Anteilen der Datenmenge W_1 und W_2 gleichgesetzt.

$$W_1 \frac{P(x_f|C_1)}{P(x_f)} = W_2 \frac{P(x_f|C_2)}{P(x_f)} \tag{62}$$

Nach dem Einfügen der vollständigen Formel der Wahrscheinlichkeitsdichte für einen Cluster, siehe Abschnitt 2.2.3.5, in die Gleichung 62 ergibt sich folgendes:

$$\begin{aligned} W_1 \frac{1}{\sqrt{(2\pi)^d |\Sigma_{C_1}|}} e^{-\frac{1}{2}(x_f - \vec{\mu_{C_1}})^T (\Sigma_{C_1})^{-1} (x_f - \vec{\mu_{C_1}})} = \\ W_2 \frac{1}{\sqrt{(2\pi)^d |\Sigma_{C_2}|}} e^{-\frac{1}{2}(x_f - \vec{\mu_{C_2}})^T (\Sigma_{C_2})^{-1} (x_f - \vec{\mu_{C_2}})} \end{aligned} \tag{63}$$

Zur Vereinfachung der Schreibweise werden jene Terme, in denen x_f nicht enthalten ist, durch die Variablen z_1 und z_2 ersetzt.

$$z_1 = W_1 \frac{1}{\sqrt{(2\pi)^d |\Sigma_{C_1}|}} \tag{64}$$

$$z_2 = W_2 \frac{1}{\sqrt{(2\pi)^d |\Sigma_{C_2}|}} \tag{65}$$

Nach der Substitution der Terme durch die Variablen z_1 und z_2 definiert sich die Gleichung als:

$$\frac{z_1}{z_2} = \frac{e^{-\frac{1}{2}(x_f - \vec{\mu_{C_2}})^T (\Sigma_{C_2})^{-1} (x_f - \vec{\mu_{C_2}})}}{e^{-\frac{1}{2}(x_f - \vec{\mu_{C_1}})^T (\Sigma_{C_1})^{-1} (x_f - \vec{\mu_{C_1}})}} \tag{66}$$

Gleichung 66 kann durch Umformung auch dargestellt werden als:

$$-2ln\Big(\frac{z_1}{z_2}\Big) = (x_f - \vec{\mu_{C_2}})^T (\Sigma_{C_2})^{-1} (x_f - \vec{\mu_{C_2}}) - (x_f - \vec{\mu_{C_1}})^T (\Sigma_{C_1})^{-1} (x_f - \vec{\mu_{C_1}}) \tag{67}$$

Der Punkt x_f muss genau zwischen den Mittelpunkten von zwei Clustern gelegen sein, um den geringsten Abstand zu dem Mittelpunkt $\vec{\mu_{C_1}}$ und dem Mittelpunkt $\vec{\mu_{C_2}}$ zu besitzen. Darum definiert sich der Abstand bzw. die Distanz $dist()$ zwischen $\vec{\mu_{C_1}}$ und $\vec{\mu_{C_2}}$ auch als der Abstand zwischen $\vec{\mu_{C_1}}$ und x_f und der Abstand zwischen $\vec{\mu_{C_2}}$ und x_f.

$$dist(\vec{\mu_{C_1}}, \vec{\mu_{C_2}}) = dist(\vec{\mu_{C_1}}, x_f) + dist(x_f, \vec{\mu_{C_2}}) \tag{68}$$

Verwendet man die allgemeine Formel für einen Punkt auf der Geraden zwischen $\vec{\mu_{C_1}}$ und $\vec{\mu_{C_2}}$ so lässt sich Formel 69 folgern, wobei $0 \leq p \leq 1$ gilt, und p in der Folge als Skalierungsfaktor bezeichnet wird.

$$p\ \vec{\mu_{C_1}} + (1-p)\ \vec{\mu_{C_2}} = x_f \tag{69}$$

Die in Gleichung 67 verwendeten Terme mit x_f lassen sich mit Hilfe des Skalierungsfaktors auch folgendermaßen darstellen:

$$\begin{aligned} (x_f - \vec{\mu_{C_1}}) &= (1-p)(\vec{\mu_{C_2}} - \vec{\mu_{C_1}}) \\ (x_f - \vec{\mu_{C_2}}) &= p(\vec{\mu_{C_1}} - \vec{\mu_{C_2}}) \end{aligned} \tag{70}$$

Gleichung 67 kann daher auch mit dem Skalierungsfaktor p dargestellt werden:

$$\begin{aligned} -2ln(\frac{z_1}{z_2}) = (p(\vec{\mu_{C_1}} - \vec{\mu_{C_2}}))^T (\Sigma_{C_2})^{-1} (p(\vec{\mu_{C_1}} - \vec{\mu_{C_2}})) - \\ ((1-p)(\vec{\mu_{C_2}} - \vec{\mu_{C_1}}))^T (\Sigma_{C_1})^{-1} ((1-p)(\vec{\mu_{C_2}} - \vec{\mu_{C_1}}) \end{aligned} \tag{71}$$

Durch Umformen von Gleichung 71 lässt sich folgern:

$$-2ln(\frac{z_1}{z_2}) = p^2(\vec{\mu_{C_1}} - \vec{\mu_{C_2}})^T(\Sigma_{C_2})^{-1}(\vec{\mu_{C_1}} - \vec{\mu_{C_2}}) - (1-p)^2(\vec{\mu_{C_2}} - \vec{\mu_{C_1}})^T(\Sigma_{C_1})^{-1}(\vec{\mu_{C_2}} - \vec{\mu_{C_1}}) \quad (72)$$

Zur Vereinfachung der Schreibweise werden folgende Terme der Gleichung 72 durch neue Variablen ersetzt.

$$a = (\vec{\mu_{C_1}} - \vec{\mu_{C_2}})^T(\Sigma_{C_2})^{-1}(\vec{\mu_{C_1}} - \vec{\mu_{C_2}}) \quad (73)$$

$$b = (\vec{\mu_{C_2}} - \vec{\mu_{C_1}})^T(\Sigma_{C_1})^{-1}(\vec{\mu_{C_2}} - \vec{\mu_{C_1}}) \quad (74)$$

$$c = 2ln(\frac{z_1}{z_2}) \quad (75)$$

Auf Grund der neuen Variablen stellt sich Formel 72, wenn sie null gesetzt wird, wie folgt dar:

$$p^2a - (1-p)^2b + c = 0 \quad (76)$$

bzw.

$$(a-b)p^2 + (2b)p + (c-b) = 0 \quad (77)$$

Wendet man die Formel zur Lösung quadratischer Gleichungen an, so ergibt sich für die Bestimmung des Skalierungsfaktors p folgendes:

$$p_{1,2} = \frac{-2b \pm \sqrt{2b^2 + 4(a-b)(b-c)}}{2(a-b)} \quad (78)$$

und in weiterer Folge:

$$p_{1,2} = \frac{-b \pm \sqrt{ab - ac + bc}}{a-b} \quad (79)$$

Für die Berechnung des Radius r wird aus Gleichung 79 nur jener Wert von p verwendet, der im Wertebereich $0 \leq p \leq 1$ liegt, da nur dieser, nach Gleichung 69, die geringste Distanz zu den beiden Mittelpunkten der Cluster $\vec{\mu_{C_1}}$ und $\vec{\mu_{C_2}}$ aufweist.

6.3.3.2 Berechnung des Radius r eines Clusters

Die Kugel mit dem Radius r liegt innerhalb des Bereichs der Clusterzugehörigkeit eines Clusters. Obwohl dieser Bereich meist ein Ellipsoid bzw. eine undefinierte Form ist [Wei06], wird der Einfachheit halber von einer Kugel ausgegangen, die den größtmöglichen Radius besitzt um noch komplett in diesem Ellipsoid enthalten zu sein.

Für die Berechnung des Radius r eines Clusters wird der Punkt x_f herangezogen. Dieser Punkt besitzt die gleiche Wahrscheinlichkeit der Zugehörigkeit zum aktuellen Cluster C_A und zu einem der anderen Cluster C_B und liegt am wenigsten weit vom Mittelpunkt $\vec{\mu_{C_A}}$ des aktuellen Clusters entfernt, siehe Abbildung 25.

Bei der Berechnung des Radius r wird die euklidische Distanz zwischen dem Mittelpunkt $\vec{\mu_{C_A}}$ und dem Punkt x_f bestimmt.

$$r = dist(\vec{\mu_{C_A}}, x_f) \tag{80}$$

7 Evaluierung

Ziel dieses Kapitel ist die Analyse des in Kapitel 6 vorgestellten EMAD Algorithmus. Dazu wird als Qualitätskriterium der Silhoutten-Koeffizient verwendet, der die Qualität der erstellten Clustering-Modelle bestimmt. Als Quantitätskriterium für die Evaluierung dient die Laufzeit des Algorithmus.

Mit Hilfe mehrerer Testläufe, der eine unterschiedliche Anzahl von Datensätzen zugrundeliegen, soll gezeigt werden, dass die Laufzeit von EMAD bei großen Datenmengen nicht linear ansteigt. Außerdem wird das Verhalten von EMAD mit verschiedenen Parametern, z.B. Anzahl der Cluster, analysiert, um den unterschiedlichen Aufwand zur Bestimmung der Endergebnisse darzustellen.

Weiters soll der Qualitätsunterschied der Clustering-Modelle von der verwendeten Initialisierung KC und EMAD untersucht werden, um die Abhängigkeit von EMAD zur Initialisierung und die Qualitätsveränderung beim Hintereinanderausführen von zwei Clustering-Methoden zu zeigen.

Im folgenden Abschnitt wird das Qualitätskriterium und in Abschnitt 7.2 das Quantitätskriterium näher beschrieben. Desweiteren werden in Abschnitt 7.3 die verwendeten Testdaten und in Abschnitt 7.4 der genaue Testplan mit den Testläufen und Testfällen vorgestellt, die die Grundlage für die Analysen darstellen. Abschließend werden in Abschnitt 7.5 die Ergebnisse der Evaluierung in Bezug auf das Qualitätskriterium bzw. in Abschnitt 7.6 die benötigten Laufzeiten von EMAD dargestellt und interpretiert. Im letzten Abschnitt wird ein Überblick der Testergebnisse gegeben.

7.1 Qualitätskriterium - Silhoutten-Koeffizient

Zur Bestimmung der Qualität eines Clustering-Modells wird der Silhouetten--Koeffizient [KR90, Kapitel 2] berechnet, der diese unabhängig von der Anzahl der Cluster bestimmt. Dabei wird von einem Clustering mit k Clustern ausgegangen, sodass $C_M = \{C_1, ..., C_k\}$. Diesen ist eine Menge von Datenobjekten x zugeordnet, dadurch gilt $C \in C_M$ und $x \in C \subset D$.

Für den Silhouetten-Koeffizienten ist die Bestimmung des durchschnittlichen Abstandes zwischen einem Datenobjekt x zu einem Cluster $C_i \in C_M$ essentiell. Dieser

wird als durchschnittliche Distanz zu den Punkten d eines Clusters definiert, das heißt als

$$dist(x, C_i) = \left(\sum_{d \in C_i} dist(x, d) \right) / |C_i|. \tag{81}$$

Bei der Berechnung der Silhouette des Objektes x, auf die der Silhouetten-Koeffizienten aufbaut, werden zwei unterschiedliche Distanzen berechnet. Zum einen $a(x)$, die durchschnittliche Distanz zwischen dem Objekt x und allen anderen Objekten in „seinem“ Cluster C_i, also dem Cluster, dem das Objekt x momentan zugeordnet ist.

$$a(x) = dist(x, C_i) \tag{82}$$

Die andere Distanz stellt $b(x)$ dar, der durchschnittliche Abstand zwischen dem Objekt x und allen anderen Objekten in jenem Cluster, dem x zugeordnet worden wäre, wenn es „seinen“ Cluster C_i nicht gegeben hätte. Dieser Cluster C_j wird als Nachbar-Cluster bezeichnet.

$$b(x) = min_{C_j \in C_M, C_j \neq C_i} dist(x, C_j) \tag{83}$$

Bei der Berechnung der Silhouette $s(x)$ ist zu berücksichtigen, dass diese 0 ist, wenn $a(x)$ und $b(x)$ identisch sind, also $a(x) = b(x)$, oder der Cluster, dem das Objekt derzeit zugeordnet ist, nur ein Objekt enthält, $|C_i| = 1$.

Ansonsten kann die Silhouette wie folgt berechnet werden:

$$s(x) = \frac{b(x) - a(x)}{max\{a(x), b(x)\}} \tag{84}$$

Die Silhouette misst, wie gut das Objekt x zu dem Cluster passt, dem es gerade zugeordnet ist. Die Werte von $s(x)$ liegen dabei zwischen -1 und 1 und können folgendermaßen interpretiert werden.

- $s(x) \approx 0$, also $a(x) \approx b(x)$: Das Objekt x liegt circa zwischen dem eigenen und dem Nachbar-Cluster.
- $s(x) \approx -1$, also $b(x) < a(x)$: Das Objekt x wurde dem Cluster schlecht zugeordnet.
- $s(x) \approx 1$, also $b(x) > a(x)$: Das Objekt x wurde dem Cluster gut zugeordnet.

Als Silhouettenweite von C_i wird der durchschnittliche Wert der Silhouetten $s(x)$ aller Objekte x von einem Cluster C_i bezeichnet. Sie misst die Qualität des Clusters

und wird definiert als

$$s(C_i) = \left(\sum_{x \in C_i} s(x) \right) / |C_i|. \tag{85}$$

Der Silhouetten-Koeffizient schließlich ist ein Maß für die Qualität des gesamten Clustering-Modells C_M. Er ist die Silhouettenweite aller Cluster C_M und wird im Folgenden als Durchschnitt aller Silhouetten definiert.

$$s(C_M) = \frac{\sum_{C \in C_M} \sum_{x \in C} s(x)}{|D|} \tag{86}$$

Je besser das Clustering C_M ist, desto höher ist der Silhouetten-Koeffizient $s(C_M)$. Als Interpretation schlagen Kaufman und Rousseeuw [KR90, Kaptitel 2] die folgende vor:

- $0.70 \leq s(C_M) \leq 1.00$: Eine starke Struktur des Clustering-Modells, bei dem die Cluster sehr gut unterschieden werden können bzw. die Objekte sehr nahe an den Mittelpunkten ihrer Cluster liegen.
- $0.50 \leq s(C_M) \leq 0.70$: Eine brauchbare Struktur des Clustering-Modells, bei dem die Objekte klar den Clustern zugeordnet werden können.
- $0.25 \leq s(C_M) \leq 0.50$: Eine schwache Struktur des Clustering-Modells, bei dem die Mittelpunkte der Cluster gefunden werden. Allerdings ist eine beachtliche Menge an „Rauschen“ vorhanden, also Objekte, die nicht eindeutig den Clustern zugeordnet werden können.
- $s(C_M) \leq 0.25$: Eine unbrauchbare Struktur des Clustering-Modells, bei dem keine signifikanten Mittelpunkte der Cluster gefunden werden. Die Mehrheit der Objekte kann nicht eindeutig zugeordnet werden.

7.2 Quantitätskriterium - Laufzeit

Für den Vergleich der Clustering-Modelle von EMAD mit unterschiedlichen Parametern oder mit Clustering-Modellen von anderen Clustering-Methoden wird die Laufzeit, die für das Clustering aufgewendet wird, herangezogen.

Damit das Verhalten von EMAD besser beurteilt werden kann, setzt sich die Messung der Laufzeit aus mehreren Teilen zusammen. Zum einen wird bei unterschied-

lichen Datenmengen die Dauer für den Aufbau des CFG-Baumes in der ersten Phase von CHAD, siehe Abschnitt 5.2, ermittelt. Zum anderen wird die Zeit für das Clustering mit KC und die Zeit für das Erstellen der Clustering-Modelle mittels EMAD gemessen.

Die Clustering-Modelle wurden auf einem Intel Duo Prozessor mit 2,00 GHz und 1024 MB Hauptspeicher erstellt.

7.3 Testdaten

Für die Evaluation des EMAD Algorithmus wurden synthetisch generierte Testdaten verwendet, die in diesem Kapitel beschrieben werden. Diese Daten wurden mit Hilfe eines Testgenerators von Goller erzeugt und können in k unabhängige, normalverteilte Cluster eingeteilt werden [Gol06].

Beim Ausführen des Testgenerators werden die Anzahl der Cluster, die Gesamtanzahl der zu erzeugenden Punkte und die Mittelpunkte der Cluster definiert. Weiters werden die Abweichung ausgehend vom Mittelpunkt des Clusters und der prozentuelle, mengenmäßige Anteil der Daten je Cluster angegeben. Die somit erstellten Cluster besitzen die Eigenschaft, dass sie die Form einer Kugel aufweisen.

Die Testdaten, die zur Evaluierung von EMAD verwendet werden, bestehen aus 5 Millionen Datensätzen, die in 5 Cluster verteilt sind und 10 Dimensionen besitzen. Von diesen Datensätzen sind 30 % nicht direkt einem Cluster zuordenbar, sodass diese „Rauschen“ darstellen. Die spezifischen Eigenschaften der 5 Cluster werden in Tabelle 2 näher ausgeführt, wobei anzumerken ist, dass sich der Anteil der Datenpunkte auf jene Datensätze bezieht, die nicht dem Rauschen zugeordnet werden.

Cluster Nummer	Anteil der Datenpunkte	Mittelpunkt	Abweichung
1	20%	1;1;2;0;1;1;0;0;0;0;	25%
2	20%	-1;2;1;0;-1;1,2;-1;-1;0;1;	25%
3	20%	0;-1;-2;0;0;3;1,2;-2;0;0;	25%
4	20%	2;-1,2;1;1;1,5;-2;3;1;0;0;	25%
5	20%	-2;0;-1;2;2;-1,8;5;1,1;0;1,5;	25%

Tabelle 2: Eigenschaften der generierten Testdaten

In Abbildung 29 sind die ersten 1000 Datensätze der Testdaten in der ersten und zweiten Dimension dargestellt.

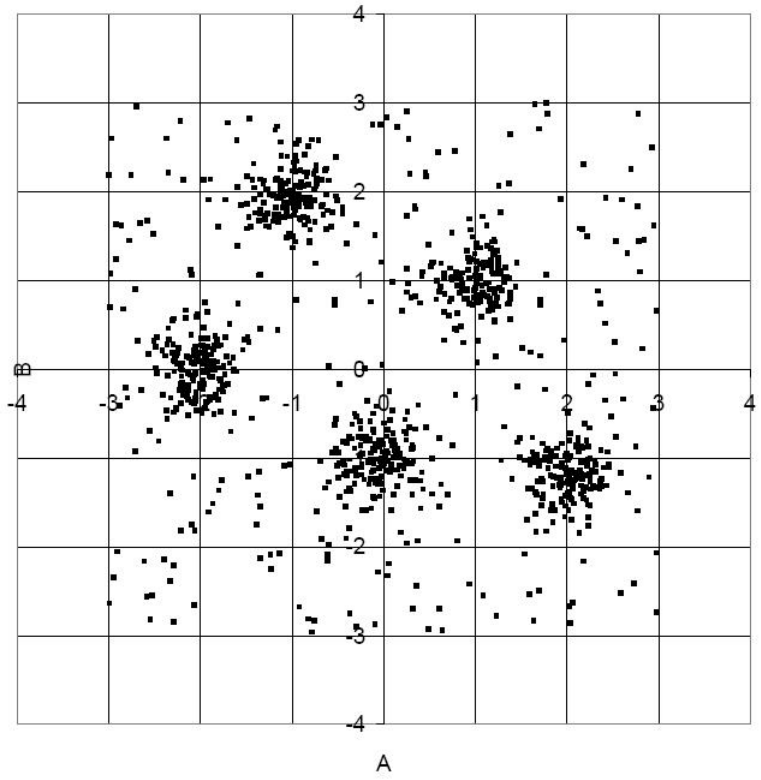

Abbildung 29: Die ersten 1000 Datensätze der Testdaten in den ersten zwei Dimensionen [Gol06, Seite 225]

7.4 Testplan

Um das Verhalten von EMAD in Bezug auf das Qualitätskriterium und das Quantitätskriterium zu verdeutlichen, werden mehrere Testläufe mit unterschiedlichen Datenmengen und mehrere Testfälle je Testlauf mit unterschiedlichen Parametern durchgeführt. Im folgenden Abschnitt werden diese näher erläutert.

Für die Evaluation werden 14 Testläufe mit unterschiedlichen Mengen an Datensätzen durchgeführt, die in Tabelle 3 aufgelistet sind. Die unterschiedlichen Mengen stellen dabei Stichproben aus den Testdaten dar. Basierend auf den Da-

tensätzen in den Testläufen wird in der ersten Phase von CHAD ein CFG-Baum, vergleiche dazu Abschnitt 5.2, erstellt.

Testlauf	Anzahl der Datensätze
1	100.000
2	200.000
3	300.000
4	400.000
5	500.000
6	600.000
7	700.000
8	800.000
9	900.000
10	1.000.000
11	2.000.000
12	3.000.000
13	4.000.000
14	5.000.000

Tabelle 3: Beschreibung der Testläufe

Bei jedem Testlauf werden mittels EMAD mehrere Clustering-Modelle mit unterschiedlichen Parametern erstellt. Diese Kombinationen der Parameter sind in den Tabellen 4, 5 und 6 aufgelistet und werden in der Folge als Testfälle bezeichnet. Zusätzlich wird für jeden Testfall das KC, siehe Abschnitt 5.4, durchgeführt.

Bei den Testfällen verändern sich die Parameter ϵ und k. Bei der Gütedifferenz ϵ handelt es sich um die maximale Differenz der Güten von zwei Clustering-Modellen. Je höher dieser Wert ist, desto eher kann das Clustering abgebrochen werden. Eine ausführlichere Beschreibung zur Gütedifferenz ϵ ist in Kaptiel 2.2.3.5 zu finden. Die Gütedifferenzen ϵ liegen bei den Testfällen zwischen 0,1 und 0,001.

Mit dem Parameter k wird die Anzahl der Cluster festgelegt, in welche die Daten beim Clustering eingeteilt werden. Die Anzahl der Cluster liegt bei diesen Testfällen bei 3, 4 bzw. und 5 Cluster.

In den Testfällen werden immer alle Ausprägungen der Parameter kombiniert, sodass sich, wie in Tabelle 4 aufgelistet, neun Testfälle bestimmen lassen.

ϵ	k
0.1	3
0.1	4
0.1	5
0.01	3
0.01	4
0.01	5
0.001	3
0.001	4
0.001	5

Tabelle 4: Testfälle für variierende Parameter ϵ und k

Zusätzlich wird EMAD noch bei Testfällen ausgeführt, die Datenpunkte mit einer unterschiedlichen Anzahl an Dimensionen verwenden. Dafür bleiben die Parameter ϵ und k konstant. Die Anzahl der Dimensionen bei diesen Testfällen, die in Tabelle 5 dargestellt sind, liegt zwischen 3 und 9.

ϵ	k	Dimensionen
0,01	5	3
0,01	5	6
0,01	5	9

Tabelle 5: Testfälle bei unterschiedlichen Dimensionen der Datenpunkte

Zuletzt werden noch die beiden Splittbedingungen, die Wahrscheinlichkeitsbedingung und die Extremwertbedingung, die in Abschnitt 6.3 erläutert werden und die bei EMAD möglich sind, verglichen. Die bereits beschriebenen Testfälle werden alle mit der Extremwertbedingung durchgeführt. Daher wird zum Vergleich für die Wahrscheinlichkeitsbedingung jener Testfall aus Tabelle 4 mit den Parametern $\epsilon = 0,01$ und $k = 5$ herangezogen.

Die Wahrscheinlichkeitsbedingung benötigt beim Clustering mittels EMAD einen zusätzlichen Parameter, den Fehlerwert *errval*, der die Genauigkeit der Wahrscheinlichkeitsbedingung bestimmt. Je kleiner dieser Wert ist, desto eher muss ein Splitt durchgeführt werden. Bei den Testfällen, siehe Tabelle 6, variiert *errval* zwischen $\frac{1}{32}$ und $\frac{1}{2048}$.

ϵ	k	$errval$
0,01	5	$\frac{1}{32}$
0,01	5	$\frac{1}{128}$
0,01	5	$\frac{1}{512}$
0,01	5	$\frac{1}{2048}$

Tabelle 6: Testfälle mit dem Fehlerwert $errval$

7.5 Qualitative Analyse der Ergebnisse

Die Clustering-Methoden erstellen auf Grund der Parameter der Testfälle Clustering-Modelle, die in diesem Abschnitt mit Hilfe des Silhouetten-Koeffizienten, der die Qualität eines Modells ausdrückt und in Abschnitt 7.1 beschrieben ist, analysiert werden. Untersucht werden dabei jene Modelle, die durch das KC und EMAD bestimmt wurden. Außerdem wird der Qualitätsunterschied der Clustering-Modelle von zwei verschiedenen Splittbedingungen untersucht, da bei EMAD die Wahrscheinlichkeitsbedingung und die Extremwertbedingung verwendet werden können.

7.5.1 Qualitätsvergleich zwischen KC und EMAD

Der Vergleich der Silhouetten-Koeffizienten mehrerer Modellen, die mittels KC und EMAD erstellt wurden, hat gezeigt, dass EMAD keine Modelle bestimmen kann, deren Qualität über jener liegt, welche die Modelle von KC aufweisen.

Die niedrige Qualität ist zum einen auf die ungenaue Bestimmung des Vektoraußenprodukts Γ zurückzuführen, das erst bei der Erstellung des CFE-Baums, der die aggregierten Daten für eine spezifische Anwendung enthält, mit der linearen Summe $\vec{LS}$ und der Anzahl der Datenpunkte N bestimmt wird. Weitere Ausführungen zur Bestimmung des Vektoraußenprodukts sind in Abschnitt 5.3.4 zu finden. Da diese Berechnung mit Hilfe der aggregierten Daten und nicht den eigentlichen Datenpunkten durchgeführt wird, sind Rechenungenauigkeiten enthalten.

Das Vektoraußenprodukt Γ wird für die Berechnung der Kovarianzmatrix und der Inverse von den Gaußverteilungen verwendet, wodurch sich hier ebenfalls Rechenungenauigkeiten ergeben, die sich in weiterer Folge bis zur Bestimmung der Clusterzugehörigkeit der Sub-Cluster bei EMAD, siehe Abschnitt 6.2, fortsetzen.

Das gesamte Modell der Gaußverteilungen weist daher einen Fehler auf, wodurch die Qualität der Ergebnisse von EMAD sinkt.

Die Initialisierung von EMAD ist ein weiterer Grund für die niedrige Qualität. Diese erfolgt durch KC, das auf k-means basiert und ein Clustering auf Grund der Distanzen, die mit den aggrgierten Daten korrekt ermittelt werden können, bestimmt. Dadurch treten keine Rechenungenauigkeiten auf. Wird eine Initialisierung von guter Qualität bei EMAD verwendet, wie es bei diesen Testdaten der Fall ist, kann die Qualität auf Grund der Rechenungenauigkeiten des Vektoraußenprodukts nicht verbessert werden. Hat die Initialisierung von KC allerdings eine schlechte Qualität, so ist ebenfalls keine Verbesserung möglich, da wegen des Modells von KC die Gaußverteilungen bei EMAD beeinträchtigt werden, und damit nur ein schlechtes Clustering-Modell bestimmt werden kann.

Abbildung 30 zeigt die Silhouetten-Koeffizienten jener Clustering-Modelle von KC und EMAD, welche ein bis fünf Millionen Datensätze in fünf Cluster einteilen. Dabei liegt die Qualität von EMAD immer unterhalb jener von KC, was auf die Rechenungenauigkeiten zurückzuführen ist. In dieser Abbildung wird außerdem die Abhängigkeit von EMAD zur Initialisierung gezeigt. Durch eine schlechtere Qualität von KC, wie es bei vier Millionen Datensätzen der Fall ist, fällt die Qualität von EMAD ebenfalls ab.

Der Silhouetten-Koeffizient ist generell ziemlich schlecht, da 30 % der Daten „Rauschen“ sind und sich die Cluster zum Teil überlappen. Unter besten Voraussetzungen ist eine schwache Struktur von $\approx 0{,}25$ zu erwarten.

7.5.2 Qualitätsvergleich der Splittbedingungen

Die Art der Splittbedingung, die auf Grund der aufbereiteten Daten im CFE-Baum notwendig ist, beeinflusst ebenfalls die Qualität des Clustering-Modells, das mittels EMAD bestimmt wird. Zum einen kann als Splittbedingung, welche die Teilung eines Knotens im Baum bestimmt, die Extremwertbedingung und zum anderen die Wahrscheinlichkeitsbedingung herangezogen werden.

Die Extremwertbedingung verwendet für die Entscheidung des Splitts die Extremwerte der Extended Cluster Features von den Sub-Clustern, vergleiche dazu Ab-

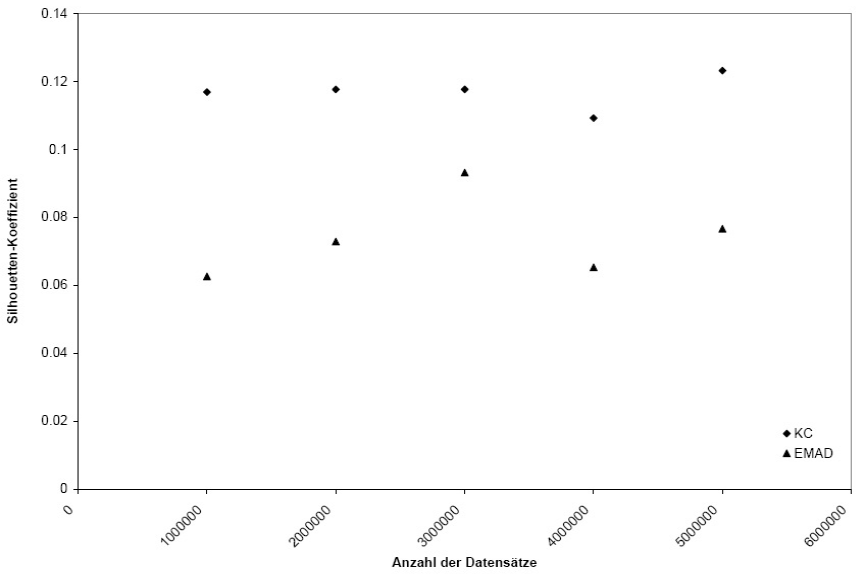

Abbildung 30: Silhouetten-Koeffizienten von KC und EMAD

schnitt 6.3.1. Dabei werden alle im Sub-Cluster enthaltenen Datenenpunkte für den Splitt berücksichtigt, sodass die Genauigkeit hoch ist.

Die Wahrscheinlichkeitsbedingung ist gegenüber der Extremwertbedingung ungenauer, da sie den Splitt auf Grund des Fehlerwerts *errval* annähert, der die maximale Wahrscheinlichkeit der Fehlklassifikation festlegt. Für die Bestimmung der Wahrscheinlichkeit wird nicht von der tatsächlichen Verteilung der Datenpunkte ausgegangen, da diese nicht bekannt ist. Stattdessen wird eine Annäherung mit Hilfe der standardisierten Normalverteilung durchgeführt, die ab einem bestimmten Wert keine gültige Lösung mehr ermitteln kann. Eine ausführliche Beschreibung der Wahrscheinlichkeitsbedingung ist in Abschnitt 6.3.2 zu finden.

Der Unterschied der Genauigkeit zwischen den beiden Splittbedingungen wirkt sich auch auf die Qualität der Clustering-Modelle aus, wie Abbildung 31 zeigt. Die Modelle, die mit Hilfe der Extremwertbedingung bestimmt werden, weisen dabei eine höhere Qualität auf, als jene, die mit der Wahrscheinlichkeitsbedingung ermittelt werden.

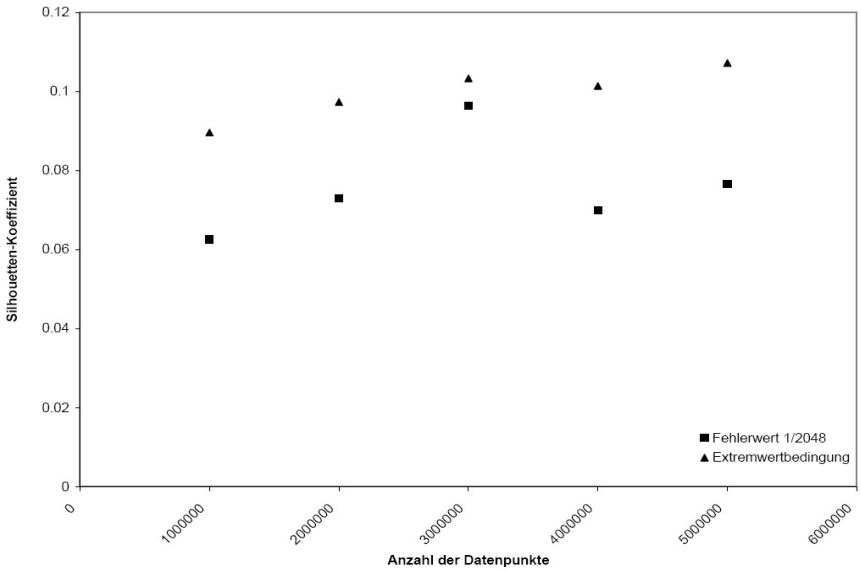

Abbildung 31: Silhouetten-Koeffizienten bei Modellen mit Extremwertbedingung und Wahrscheinlichkeitsbedingung

7.6 Quantitative Analyse der Ergebnisse

In diesem Abschnitt werde die Laufzeiten von EMAD und der ersten Phase von CHAD, die in Abschnitt 7.2 als Quantitätskriterium angeführt werden, analysiert. Dabei wird im folgenden Abschnitt auf die Laufzeit für den Aufbau des CFG-Baums eingegangen. In Abschnitt 7.6.2 wird die Laufzeit von EMAD mit jener von KC verglichen.

Die anschließenden Abschnitte vergleichen Clustering-Modelle, die von EMAD bestimmt wurden, und auf unterschiedlichen Parametern und unterschiedlichen Datenmengen basieren. So wertet Abschnitt 7.6.3 das Verhalten von EMAD bei der Bestimmung einer unterschiedlichen Anzahl an Cluster in den gleichen Daten aus. Abschnitt 7.6.4 geht auf die Gesamtlaufzeit bei unterschiedlichen Abbruchbedingungen von EMAD ein, und Abschnitt 7.6.5 analysiert die Laufzeit je Iteration bei einer unterschiedlichen Anzahl an Dimensionen der Datenpunkten. Abschließend werden in Abschnitt 7.6.6 die Laufzeiten der beiden Splittbedingungen, der Wahrscheinlichkeitsbedingung und der Extremwertbedingung, verglichen.

7.6.1 Laufzeit für den Aufbau des CFG-Baums

In diesem Abschnitt werden die Laufzeiten für den Aufbau des CFG-Baums in der ersten Phase von CHAD analysiert. Bei jedem Testlauf wird ein CFG-Baum, der die gesamten Daten enthält aufgebaut. Diese Testläufe verwenden, wie in Tabelle 3 aufgelistet, unterschiedliche Datenmengen. Abbildung 32 stellt die benötigte Laufzeit in Sekunden für den Aufbau des CFG-Baums bei allen Testläufen dar.

Da bei jedem Testlauf die gesamten Daten eingelesen werden, steigt die Laufzeit bei steigender Datenmenge linear an, wodurch in der ersten Phase von CHAD der Aufwand $O(n)$ beträgt.

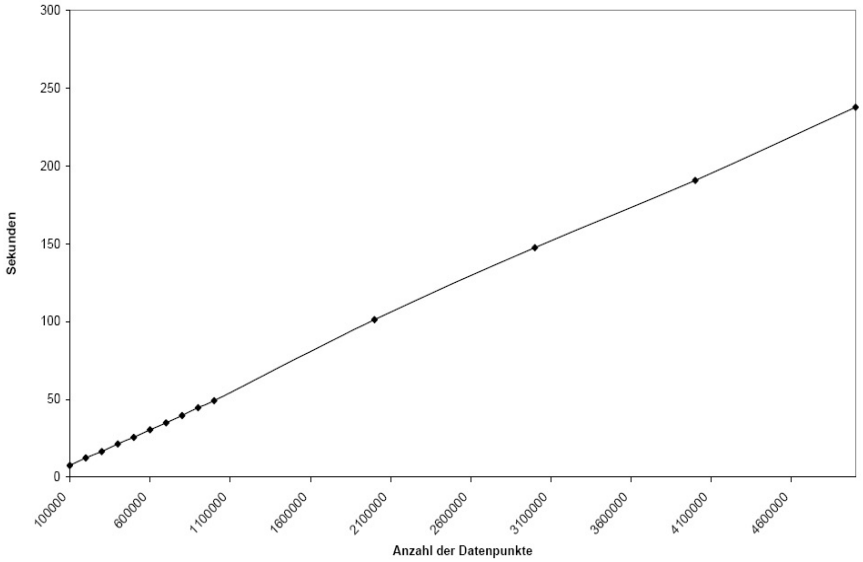

Abbildung 32: Laufzeit für den Aufbau des CFG-Baums bei unterschiedlichen Datenmengen

7.6.2 Laufzeit für das KC

In diesem Abschnitt wird die Laufzeit von EMAD mit einer anderen Clustering-Methode verglichen. Dazu wird KC herangezogen, das auf dem k-means Algorithmus basiert und in der dritten Phase von CHAD, genauso wie EMAD, ein Clustering erstellt. KC weist selbst eine gute Skalierbarkeit bei großen Datenmengen auf.

Da das KC auch die Initialisierung für EMAD darstellt, werden für den Vergleich der Clustering-Methoden nur jene Laufzeiten berücksichtigt, die für das eigentliche Clustering und nicht für die Initialisierung aufgewendet werden.

Abbildung 33 zeigt, dass die Gesamtlaufzeit für das Clustering von KC grundsätzlich unter jener von EMAD liegt und damit schneller zu einem Ergebnis kommt. Dies ist darauf zurückzuführen, dass KC für jeden Sub-Cluster den nächstgelegenen Mittelpunkt bestimmt, während EMAD für jeden Sub-Cluster die Wahrscheinlichkeit der Clusterzugehörigkeit zu allen Clustern bestimmt. Dadurch hat EMAD mehr Rechenoperationen auszuführen als KC, was sich auf die Laufzeit auswirkt.

Das Sinken der Laufzeit von EMAD bei jenem Testlauf mit drei Millionen Datenpunkten ist darauf zurückzuführen, dass in der Abbildung die Gesamtlaufzeiten und nicht die Laufzeiten je Iterationen dargestellt werden. EMAD muss bei diesem Testlauf auf Grund einer guten Initialisierung durch KD weniger Iterationen für das Endergebnis durchführen, wodurch die Gesamtlaufzeit sinkt.

Die hier analysierten Laufzeiten beruhen auf einem Clustering, das sich aus vier Clustern zusammensetzt, wobei alle zehn Dimensionen berücksichtigt werden. Als Abbruchbedingung wird für beide Algorithmen die Gütedifferenz $\epsilon = 0,01$ verwendet.

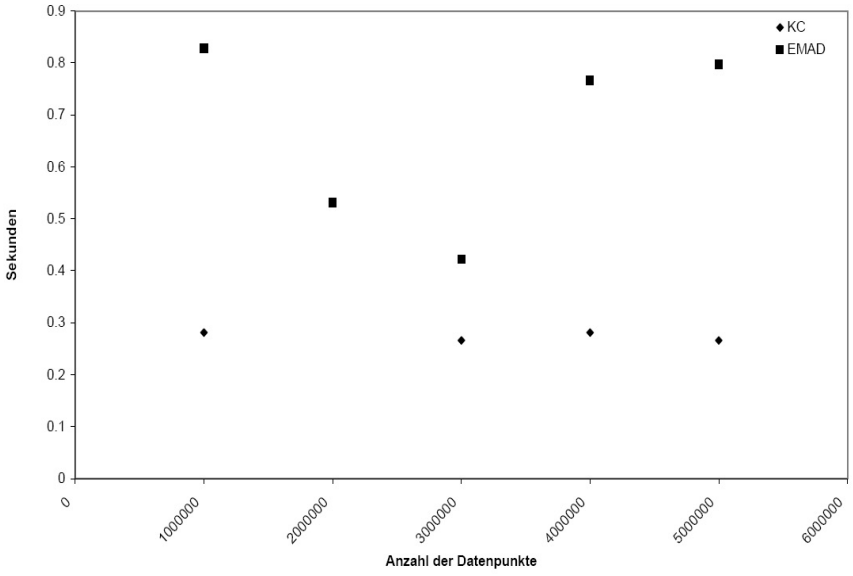

Abbildung 33: Gesamtlaufzeit beim Clustering mittels EMAD und KC

7.6.3 Laufzeit bei unterschiedlicher Clusteranzahl

Bei EMAD wird die Wahrscheinlichkeit der Clusterzugehörigkeit bei jedem Sub-Clusters für alle Cluster bestimmt, vergleiche Abschnitt 6.2, wodurch die Laufzeit bei einer steigenden Anzahl an zu bestimmenden Clustern zunimmt, da die Menge an Berechnungen für das gesamte Clustering ansteigt.

Es ist aus Abbildung 34 erkennbar, dass für jeden zusätzlich zu bestimmenden Cluster im Modell mit einer Steigerung der Laufzeit von ≈ 5 Millisekunden pro Iteration zu rechnen ist. Da die meisten Clustering-Modelle, wie die Testläufe gezeigt haben, erst bei über 20 Iterationen ermittelt werden konnten, erhöht jeder zusätzlich zu bestimmende Cluster die Laufzeit um mindestens eine Zehntelsekunde.

Die in Abbildung 34 dargestellten Laufzeiten pro Iteration beziehen sich auf Clustering-Modelle bestehend aus 3, 4 und 5 Clustern, wobei von den Datensätzen alle Dimensionen berücksichtigt werden. Als Abbruchbedingung wird bei EMAD die Gütedifferenz $\epsilon = 0,01$ verwendet.

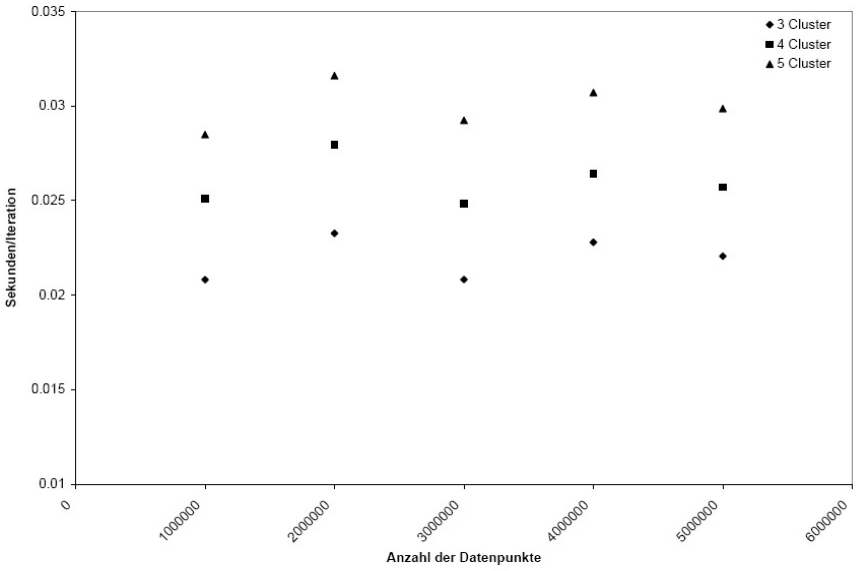

Abbildung 34: Laufzeit je Iteration bei einer unterschiedlichen Anzahl an Clustern

7.6.4 Laufzeit bei unterschiedlicher Gütedifferenz ϵ

In diesem Abschnitt soll die Gütedifferenz ϵ analysiert werden, das den Abbruch des Algorithmus EMAD auslöst. Es bestimmt die maximale Differenz zwischen den Güten von zwei Modellen. Eine ausführliche Beschreibung zum Abbruch des Algorithmus erfolgt in Abschnitt 6.2.

Um die Laufzeit des Algorithmus im Bezug auf unterschiedliche Gütedifferenzen ϵ zu analysieren, werden aus den Testläufen nur Clustering-Modelle mit 5 Clustern herangezogen, wobei die verwendeten Datenpunkte zehn Dimensionen aufweisen.

Wie aus Abbildung 35 ersichtlich, in der die Gesamtlaufzeit für das Clustering mittels EMAD dargestellt ist, beeinflusst die Gütedifferenz ϵ die Laufzeit, da bei einem niedrigen Wert die Anzahl der Iterationen ansteigt. Dies wird besonders bei dem Testlauf mit drei Millionen Datensätzen deutlich, das mit der Gütedifferenz $\epsilon = 0,001$ eine hohe Laufzeit aufweist.

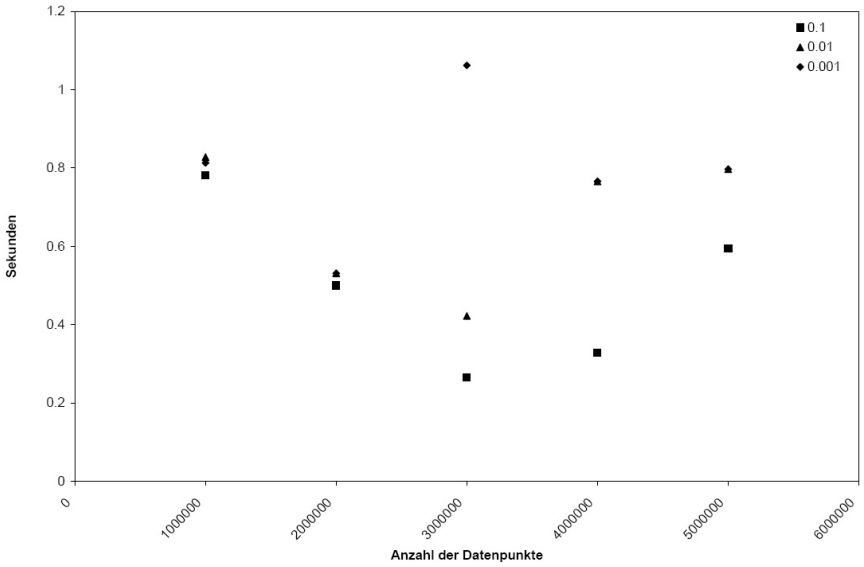

Abbildung 35: Laufzeit beim Clustering mit unterschiedlicher Gütedifferenz ϵ

7.6.5 Laufzeit bei einer unterschiedlichen Anzahl an Dimensionen

In diesem Abschnitt wird die Laufzeit von EMAD im Bezug auf die Anzahl der Dimensionen der Datenpunkte analysiert, da die Dimensionen die Menge an Rechenoperationen bestimmen.

Für den Vergleich der Laufzeit werden aus den Testläufen Clustering-Modelle mit 5 Clustern und Datenpunkten mit 3, 6 bzw. 9 Dimensionen herangezogen, wobei die Gütedifferenz ϵ, das den Abbruch des Algorithmus bestimmt, bei 0,01 liegt.

Wie aus Abbildung 36 ersichtlich, verlängert eine höhere Anzahl an Dimensionen die Laufzeit je Iteration von EMAD. Dabei kann von einem exponentiellen Anstieg ausgegangen werden, was auf die zu berechnenden Matrizen bei EMAD zurückzuführen ist, sodass schon ab achtzehn Dimensionen die Laufzeit je Iteration um mehr als eine Zehntelsekunde steigt.

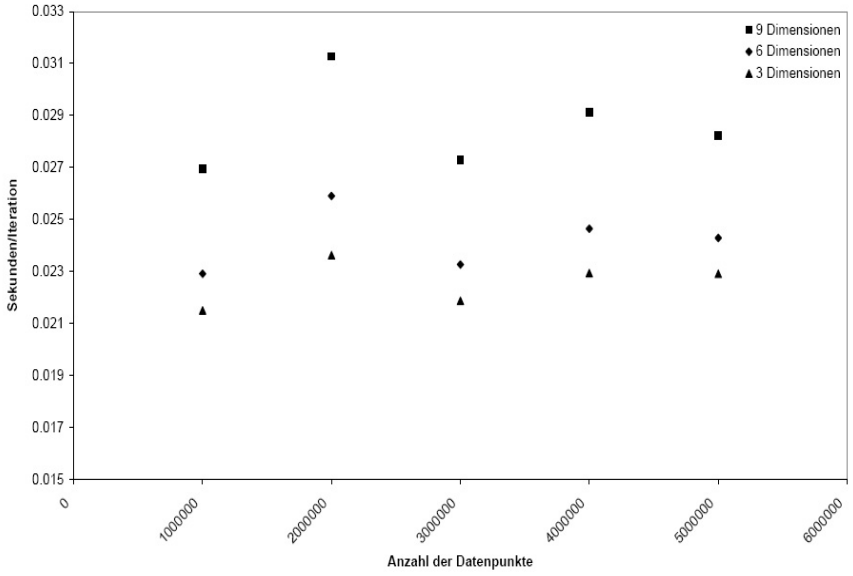

Abbildung 36: Laufzeit pro Iteration mit unterschiedlichen Dimensionen der Datenpunkte

7.6.6 Laufzeit bei unterschiedlichen Splittbedingungen

Die Splittbedingung, die auf Grund des CFE-Baums für das Clustering notwendig ist, kann durch die Extremwertbedingung oder die Wahrscheinlichkeitsbedingung

erfolgen. Da diese Bedingungen unterschiedlich ausgeführt werden, haben sie auch eine andere Auswirkung auf die Laufzeit, wie in diesem Abschnitt dargestellt wird.

Für die Extremwertbedingung werden für jeden Sub-Cluster während des Clusterings die Extremwerte bestimmt, wodurch jener Bereich ermittelt werden kann, innerhalb dem alle Tupel des Sub-Clusters liegen, vergleiche dazu Abschnitt 6.3.1. Zur Bestimmung der Extremwerte werden dabei die Vektoren, die das Minimum und das Maximum des Sub-Clusters enthalten, herangezogen und in jeder Dimension des Datenpunktes verglichen.

Bei der Wahrscheinlichkeitsbedingung wird zu Beginn des Clusterings die inverse Normalverteilung mit dem Fehlerwert *errval* berechnet. Während des Clusterings werden für die Wahrscheinlichkeitsbedingung die Sub-Cluster mit dieser inversen Normalverteilung verglichen. Weitere Ausführungen sind in Abschnitt 6.3.2 zu finden.

Da bei der Wahrscheinlichkeitsbedingung der Aufwand für die Berechnung der inversen Normalverteilung und der Vergleich beim Clustering höher ist, liegt auch die Laufzeit je Iteration über jener der Extremwertbedingung. Auch ein Steigen des Fehlerwerts *errval* kann die Laufzeit der Wahrscheinlichkeitsbedingung nicht stark genug verbessern.

Wie aus Abbildung 37 ersichtlich, ist die Laufzeit der Extremwertbedingung geringer als jene der Wahrscheinlichkeitsbedingung, selbst wenn diese mit dem höchsten Fehlerwert $errval = \frac{1}{32}$ der Testfälle verglichen wird. Für diese Abbildung werden Clustering-Modelle mit 5 Clustern, die Datensätze mit zehn Dimensionen verwenden, herangezogen. Die Laufzeit wird bei EMAD mit der Gütedifferenz $\epsilon = 0,01$ bestimmt.

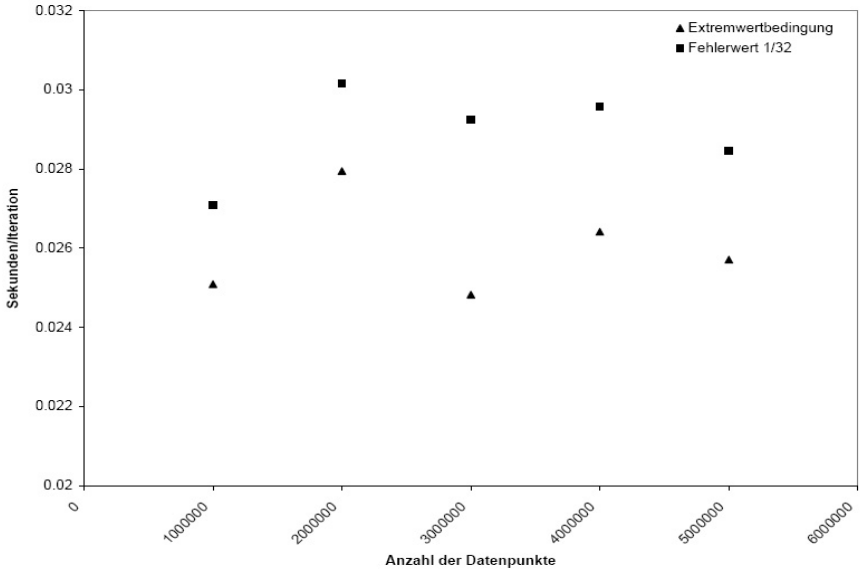

Abbildung 37: Laufzeit mit unterschiedlichen Splittbedingungen

7.6.7 Überblick der Testergebnisse

In den Abschnitten 7.6.3 bis 7.6.6 ist das Verhalten von EMAD bei der Bestimmung von Clustering-Modellen mit unterschiedlichen Parametern analysiert worden, wobei sich gezeigt hat, dass sowohl die Anzahl der Cluster, die Anzahl der Dimensionen, die Gütedifferenz ϵ und die Art der Splittbedingung Einfluss auf die Laufzeit je Iteration und damit auf die Gesamtlaufzeit des Algorithmus nehmen. Die Wahl der Parameter für das Clustering sollte deshalb nicht zufällig erfolgen, sondern sich auf das zu bestimmende Clustering beziehen.

Weiters konnte durch die Evaluation festgestellt werden, dass die Extremwertbedingung der Wahrscheinlichkeitsbedingung vorzuziehen ist, da diese eine kürzere Laufzeit aufweist und eine bessere Qualität bei den Clustering-Modellen erzielen kann.

Außerdem wurde in Abschnitt 7.5.1 dargestellt, dass eine Initialisierung von EMAD mittels KC nicht sinnvoll ist, da anstelle einer Verbesserung der Qualität durch Hintereinanderausführen von zwei Clustering-Methoden, eine Verschlechterung eintritt, was zum einen auf die Rechenungenauigkeit von EMAD und zum anderen auf die Initialisierung durch KC zurückzuführen ist.

8 Resümee

Abschließend werden die zentralen Punkte der vorliegenden Arbeit hervorgehoben und die Ergebnisse der Evaluation zusammengefasst, wobei auch ein Ausblick zur Weiterentwicklung des hier vorgestellten Verfahrens gegeben wird.

Es wurde der Ansatz des Vorausschauenden Data-Mining vorgestellt und gezeigt, dass bei großen Datenmengen dieser Ansatz wesentliche Vorteile gegenüber dem Prozess der Knowledge Discovery in Databases und dessen Teilbereich Data-Mining hat. Der Ansatz Vorausschauendes Data-Mining baut auf die einmalige Voranalyse der Daten auf, sodass mit Hilfe dieser mehrmals konkrete Analysen durchgeführt werden, die schneller zu einem Endergebnis führen.

Außerdem wurde der Algorithmus CHAD, Clustering Hierarchically Aggregated Data, vorgestellt, der den Ansatz des Vorausschauenden Data-Mining weitgehend umsetzt und sich in drei Phasen gliedert. Die erste Phase erstellt eine Aggregation der gesamten Daten mit Hilfe eines Clustering-Verfahrens. Die zweite Phase ermöglicht daraus die Extrahierung der aggregierten relevanten Daten für eine konkrete Anwendung, sodass in der dritten Phase die unterschiedlichen nachfolgenden Verfahren auf dieser aufbauen können.

Den zentralen Teil dieser Arbeit stellt der Algorithmus EMAD, Erwartungsmaximierung mit aggregierten Daten, dar. EMAD wird in der dritten Phase von CHAD verwendet und führt ein Clustering von bereits aggregierten und für eine Anwendung relevanten Daten durch. Der Algorithmus basiert auf der Erwartungsmaximierung, die für die Verwendung von aggregierten Daten angepasst wurde, damit auch bei großen Datenmengen eine interaktive Verwendung möglich wird.

Die gute Einsetzbarkeit von EMAD bei großen Datenmengen konnte durch die Evaluierung der Laufzeit gezeigt werden, da ein interaktives Arbeiten mit diesem Algorithmus umgesetzt werden kann. Die Verwendung von aufbereiteten Daten zur schnelleren Bestimmung der Ergebnisse, wie es im Ansatz des Vorausschauenden Data-Mining beschrieben wird, ist daher erfolgreich umgesetzt worden.

Die Evaluierung der Qualität von Clustering-Modellen, die mittels EMAD bestimmt wurden, zeigte jedoch noch Schwierigkeiten auf. So werden die speziell für EMAD notwendigen Aggregationen bei der derzeitigen Umsetzung in der zweiten

Phase von CHAD auf Grund bereits aggregierter Daten bestimmt, wodurch Rechenungenauigkeiten entstehen, die sich in weiterer Folge auf das Endergebnis von EMAD auswirken. Außerdem hat sich die verwendete Initialisierung als ungeeignet erwiesen, da ein lokales Minimum von KC bei EMAD auch zu einem lokalen Minimum führt.

Um die Qualität der Ergebnisse von EMAD zu verbessern ist die Verschiebung der notwendigen Aggregationen für EMAD von der zweiten Phase in die erste Phase von CHAD zu empfehlen, wodurch die Rechenungenauigkeit gemindert werden kann und sich die Qualität der Clustering-Modelle erhöht.

Zur Verbesserung der Initialisierung wird anstelle von KC der Algorithmus FREM vorgeschlagen, der wegen der oftmaligen Neuberechnung des Modells ein lokales Minimum vermeidet, und dadurch eine bessere Initialisierung für EMAD darstellt.

A Berechnungen der Kovarianzmatrizen

Für das Clustering mittels EMAD müssen diverse Matrizenrechnungen durchgeführt werden. Da es dafür bereits bestehende Algorithmen gibt und deren neue Implementierung nicht zentraler Bestandteil dieser Arbeit war, wurden diese verwendet.

Beim Clustering sind zum einen Operatoren, wie beispielsweise Addition, Subtraktion, Multiplikation und Division, mit Matrizen durchzuführen. Zum anderen ist die Berechnung der Determinante und der Inverse von einer Matrix notwendig. Die dafür verwendeten Algorithmen werden in den folgenden Abschnitten vorgestellt.

A.1 Matrix erstellen und Operatoren berechnen

Zur Erstellung und Manipulation mathematischer Matrizen wurde die Struktur der Klasse „matrix“ aus dem Buch [Eck89, Anhang B] verwendet. Die in Tabelle 7 aufgelisteten Prozeduren wurden von dieser Klasse übernommen.

Prozedur	Beschreibung
matrix()	Konstruktor zur Erstellung eines Matrix-Objektes;
∼matrix()	Destruktor zum Löschen eines Matrix-Objektes;
int rows() bzw. int cols()	Bestimmt die Anzahl der Reihen bzw. Spalten eines Matrix-Objektes;
clear()	Löscht die numerischen Werte in einem Matrix-Objekt;
matrix operator=()	Weist den Pointer zu;
matrix operator+()	Addition eines Matrix-Objektes mit einer Konstante;
matrix operator-()	Subtraktion einer Konstante von einem Matrix-Objekt;
matrix operator*()	Multiplikation eines Matrix-Objektes mit einer Konstante bzw. eines Matrix-Objektes;
double val()	Ermittelt den numerischen Wert eines Matrix-Objektes in einer bestimmten Reihe und Spalte;

Tabelle 7: Prozeduren der Klasse „matrix“ von [Eck89, Anhang B]

Zur optimalen Berechnung des Clustering-Modells durch EMAD wurde die Klasse „matrix“ noch mit den Prozeduren aus Tabelle 8 erweitert.

Prozedur	Beschreibung
vector operator*()	Multipliziert einen transponierten Vektor mit einem Matrix-Objekt;
matrix operator/()	Dividiert ein Matrix-Objekt durch eine Konstante;
double determinant()	Bestimmt die Determinante für das aktuelle Matrix-Objekt;
matrix inverse()	Bestimmt die Inverse für das aktuelle Matrix-Objekt;

Tabelle 8: Erweiterungen der Klasse „matrix“

A.2 Determinante und Inverse der Matrix berechnen

In diesem Abschnitt werden die Algorithmen, die für die Berechnung der Inverse und Determinante verwendet werden, erläutert. Für die Berechnungen ist eine Zerlegung der Matrix notwendig. Dafür wurden die Algorithmen von Press, Teukolsky und Vetterling [PTVF92] herangezogen. Sie basieren grundsätzlich auf dem Algorithmus von Crout, der eine quadratische Matrix A in das Produkt einer linken unteren Dreiecksmatrix L und einer rechten oberen Dreiecksmatrix U zerlegt (LU-Zerlegung).

Im Folgenden wird der Algorithmus LUDCMP kurz beschrieben, der die Zerlegung eine Matrix durchführt. Bei der Zerlegung werden hier keine Dreiecksmatrizen erstellt, sondern es wird die Ausgangsmatrix A, mit der Größe $N \times N$, durch reihenweises Verschieben zerlegt. Die Anzahl der Verschiebungen D sowie die Verschiebungen *Indx* werden für die Berechnung der Inversen gespeichert. Um Fehler durch singuläre Matrizen zu vermeiden, werden die Nullwerte in der Diagonalen der Matrix A durch einen kleinen positiven Wert, 1^{-20}, ersetzt. Der Algorithmus 14 nach [Zab06, Link: LU decomposition] soll die Funktionsweise dieser Zerlegung verdeutlichen. Dabei stellt die Matrix A eine globale Variable dar.

Für die Bestimmung der Determinante *Det* einer Matrix, siehe Algorithmus 11 nach [Zab06, Link: Determinant of a matrix], wird die mittels LUDCMP zerlegte Matrix A herangezogen. Die Determinante stellt das Produkt aus den Elementen der Diagonale der zerlegten Matrix A dar.

Bei der Berechnung der Inversen wird die Matrix A zuerst durch LUDCMP zerlegt. Anschließend werden mit dem Algorithmus LUBKSB die einzelnen Spalten B der inversen Matrix X berechnet. Dafür werden die gespeicherten Verschiebungen *Indx* herangezogen. Der Algorithmus 13 von [Zab06, Link: Solution of linear algebraic equations] soll diese Berechnung zeigen. Dabei sind die Matrix A sowie die Verschiebungen *Indx* und die Spalte der Inversen B globale Variablen.

Nachdem die Spalten der Inversen B berechnet worden sind, werden sie in der Matrix X zur inversen Matrix von A zusammengefügt. Der Algorithmus 12 nach [Zab06, Link: Inverse of a matrix] beinhaltet das vollständige Vorgehen für die Berechnung der Inversen.

Weiterführende Informationen zu den Algorithmen von Press, Teukolsky und Vetterling sind in deren Buch [PTVF92] zu finden.

Algorithm 11 DETERMINANTE (Größe N)

1: $Det \leftarrow$ CALL LUDCMP(N);
2: **for all** $J \in \{1, \ldots, N\}$ **do**
3: $\quad Det \leftarrow Det * A.J.J$;
4: **end for**
5: **return** Det

Algorithm 12 INVERSE (Größe N)

1: CALL LUDCMP(N);
2: **for all** $J \in \{1, \ldots, N\}$ **do**
3: $\quad B.I \leftarrow 0$; $B.J \leftarrow 1$;
4: $\quad$ CALL LUBKSB(N);
5: $\quad$ **for all** $I \in \{1, \ldots, N\}$ **do**
6: $\quad\quad X.I.J \leftarrow B.I$;
7: $\quad$ **end for**
8: **end for**

Algorithm 13 LUBKSB (Größe N)

```
 1: L ← 0;
 2: for all I ∈ {1, ..., N} do
 3:    P ← Indx.I; Sum ← B.P; B.P ← B.I;
 4:    if L <> 0 then
 5:       for all J ∈ {L, ..., I − 1}  do
 6:          (Sum ← Sum − A.I.J ∗ B.J);
 7:       end for
 8:    end if
 9:    if Sum <> 0 then
10:       L ← I;
11:    end if
12:    B.I ← Sum;
13: end for
14: for all I ∈ {N, ..., 1} do
15:    Sum ← B.I;
16:    for all J ∈ {I + 1, ..., N} do
17:       Sum ← Sum − A.I.J ∗ B.J;
18:    end for
19:    B.I ← Sum/A.I.I;
20: end for
21: return
```

Algorithm 14 LUDCMP (Größe N)

```
 1: D ← 1; Tiny ← 1^{-20};
 2: for all I ∈ {1,...,N} do
 3:    Max ← 0;
 4:    for all J ∈ {1,...,N} do
 5:       W ← ABS(A.I.J);
 6:       if W > Max then
 7:          Max ← W;
 8:       end if
 9:    end for
10:    Vv.I ← 1/Max;
11: end for
12: for all J ∈ {1,...,N} do
13:    for all I ∈ {1,...,J − 1} do
14:       Sum ← A.I.J;
15:       for all K ∈ {1,...,I − 1} do
16:          Sum ← Sum − A.I.K ∗ A.K.J;
17:       end for
18:       A.I.J ← Sum;
19:    end for
20:    Max ← 0;
21:    for all I ∈ {J,...,N} do
22:       Sum ← A.I.J;
23:       for all K ∈ {1,...,J − 1} do
24:          Sum ← Sum − A.I.K ∗ A.K.J;
25:       end for
26:       A.I.J ← Sum;W ← Vv.I ∗ ABS(Sum);
27:       if W >= Max then
28:          Max ← W;Imax ← I;
29:       end if
30:    end for
31:    if J <> Imax then
32:       for all K ∈ {1,...,N} do
33:          W ← A.Imax.K; A.Imax.K ← A.J.K; A.J.K ← W;
34:       end for
35:       D ← −D; Vv.Imax ← Vv.J;
36:    end if
37:    Indx.J ← Imax;
38:    if A.J.J == 0 then
39:       A.J.J ← Tiny;
40:    end if
41:    if J <> N then
42:       W ← 1/A.J.J;
43:       for all I ∈ {J + 1,...,N} do
44:          A.I.J ← A.I.J ∗ W;
45:       end for
46:    end if
47: end for
48: return D
```

Literatur

[Ack06] P. Acklam. An algorithm for computing the inverse normal cumulative distribution function. home.online.no/ ∼pjacklam/notes/invnorm/, Stand: 10.01.2006. nicht publiziert.

[Arm79] G. Arminger. *Faktorenanalyse: Statistik für Soziologen 3.* Teubner, Stuttgart, 1979.

[Bas94] H. Basler. *Grundbegriffe der Wahrscheinlichkeitsrechnung und Statistischen Methodenlehre.* Physica-Verlag, Heidelberg, 1994.

[BFR98a] P. Bradley, U. Fayyad, and C. Reina. Scaling Clustering Algorithms to Large Databases. In *Proc. 4th International Conf. on Knowledge Discovery and Data Mining (KDD98).* AAAI Press, 1998.

[BFR98b] P. Bradley, U. Fayyad, and C. Reina. Scaling EM (Expectation-Maximization) Clustering to Large Databases. *Microsoft Research Technical Report MSR-TR-98-35*, 1998.

[BFR00] P. Bradley, U. Fayyad, and C. Reina. Clustering Very Large Databases Using EM Mixture Models. *Pattern Recognition, 2000. Proceedings. 15th International Conference on Pattern Recognition (ICPR'00)*, 2:76–80, 2000.

[BH99] M. Berthold and D. Hand. *Intelligent Data Analysis: An Introduction.* Springer, Heidelberg, 1999.

[BH05] A. Büchter and H. Henn. *Elementare Stochastik - Eine Einführung in die Mathematik der Daten und des Zufalls.* Springer, Berlin, Heidelberg, New York, 2005.

[BKK01] M. Breunig, H. Kriegel, and P. Kröger. Data Bubbles: Quality Preserving Performance Boosting for Hierarchical Clustering. In *SIGMOD '01: Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 79–90, New York, NY, USA, 2001. ACM Press.

[BKKK04] C. Böhm, K. Kailing, P. Kröger, and H. Kriegel. Immer größere und komplexere Datenmengen: Herausforderungen für Clustering-Algorithmen. *Datenbank-Spektrum*, 9:11–17, 2004.

[BR93] J. Banfield and A. Raftery. Model-based gaussian and non-Gaussian Clustering. *Biometrics*, 49:803–821, 1993.

[CHY96] M. Chen, J. Han, and P. Yu. Data Mining: An Overview from a Database Perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):866–883, 1996.

[Cod06] W. J. Cody. SUBROUTINE CALERF. www.netlib.org/specfun/erf, Stand: 10.01.2006.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelyhood from Incomplete Data via the EM algorithm. *Journal of the Royal Statistical Society*, Series B, 39(1):1–31, 1977.

[Eck89] B. Eckel. *C++ Einführungskurs*. McGraw-Hill Book Company GmbH, Hamburg, 1989.

[ES00] M. Ester and J. Sander. *Knowledge Discovery in Databases*. Springer, Berlin Heidelberg New York, 2000.

[FN71] L. Fisher and J. Van Ness. Admissible clustering procedures. *Biometrika*, 58:91–104, 1971.

[For65] E. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21(3):768–69, 1965.

[FPSS96a] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, pages 37–54, 1996.

[FPSS96b] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Commun. ACM*, 39(11):27–34, 1996.

[FPSSU96] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. Advances in Knowledge Discovery and Data Mining. *MIT Press*, 1996.

[Für04] D. Fürst. *Effizientes K-Clustering: K-Centroids Clustering auf Basis von hierarchisch aggregierten Daten*. Diplomarbeit eingereicht an der Johannes Kepler Universität Linz, Institut für Wirtschaftsinformatik - Data & Knowledge Engineering, 2004.

[GG92] A. Gersho and R. Gray. *Vector quantization and signal compression*. Kluwer academic Publishers, Boston, 1992.

[Gol04] M. Goller. CHAD - Clustering Hierarchically Aggregated Data. Präsentation über das Konzept der Dissertation, 2004.

[Gol06] M. Goller. *Theory and Implementation of Anticipatory Data Mining*. PhD thesis, Institut für Wirtschaftsinformatik – Data & Knowledge Engineering, 2006.

[HK01] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publisher, USA, 2001.

[KK00] U. Küsters and C. Kalinowski. Traditionelle Verfahren der multivariaten Statistik. In *Handbuch Data Mining im Marketing*. Vieweg, Wiesbaden, 2000.

[KR89] L. Kaufman and P. Rousseeuw. *Finding Groups in Data*. John Wiley and Sons, New York, 1989.

[KR90] L. Kaufman and P. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley and Sons, New York, 1990.

[Mac67] J. MacQueen. Some Methods for Classification and Analysis of Multivariate. *5th Berkeley Symp. Math. Statist. Prob.*, 1:82–88, 1967.

[Mit97] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[OO02] C. Ordonez and E. Omiecinski. FREM: Fast and Robust EM Clustering for Large Data Sets. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 590–599, New York, NY, USA, 2002. ACM Press.

[PTVF92] H. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C : the art of scientific computing*. University Press, Cambridge, second edition, 1992.

[Wei06] E. Weisstein. Wolfram Math World, 2006. Stand 12.08.2006.

[WF00] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation*. Morgan Kaufmann, 2000.

[Zab06] V. Zabrodsky. Linear Algebraic Equations. us.geocities.com/zabrodskyvlada/aat/a_contents.html#contents, Stand: 14.08.2006.

[Zha96] T. Zhang. *Data Clustering für Very Large Datasets Plus Applicationons*. Dissertation, Univ. of Wisconsin-Madison, Computer Sciences Deptartment, 1996.

[ZRL96] T. Zhang, R. Ramakrishnan, and M. Livny. Interactive Classification of Very Large Datasets with BIRCH. Proc. of Workshop on Research Issues on Data Mining and Knowledge Discovery (in cooperation with ACM-SIGMOD'96), Montreal, Canada, 1996.

[ZRL97] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: A New Data Clustering Algorithm and Its Applications. *Data Mining and Knowledge Discovery*, 1(2), 1997.