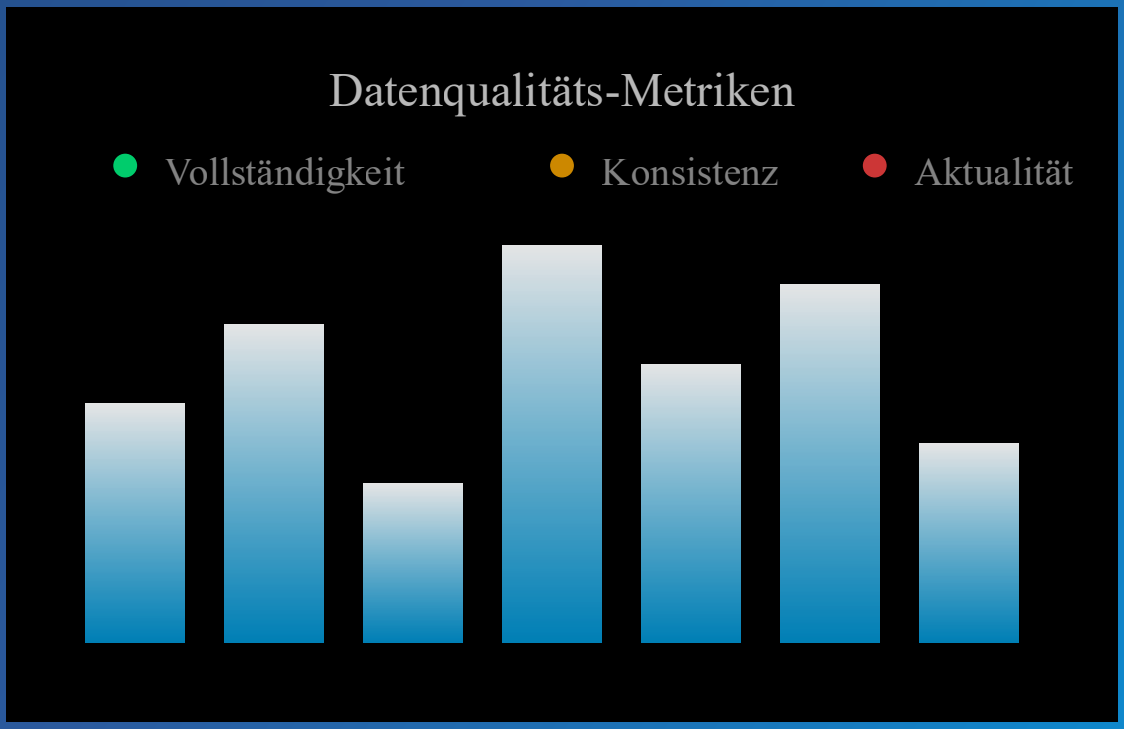
PRAXISHANDBUCH

HANDBUCH DATEN-QUALITÄT

Best Practices für erfolgreiche Datenqualitätsinitiativen



Andreas Igl | Josef Gruber

www.handbuch-datenqualitaet.de

Handbuch Datenqualität

Andreas Igl Josef Gruber

Oktober 2025

Haftungsausschluss
Der Inhalt dieses Buches wurde mit größter Sorgfalt erstellt. Dennoch kann keine Gewähr für die Richtigkeit, Vollständigkeit und Aktualität der Angaben übernommen werden. Die Nutzung der Inhalte erfolgt auf eigene Verantwortung.

Kolophon
Dieses Buch wurde mit Hilfe von KOMA-Script und LaTeX unter Verwendung der kaobook-Klasse gesetzt.

Verlag und Veröffentlichung
Erstveröffentlichung 2025 als Open-Source-Ausgabe mit ISBN 978-3-00-084832-2.
Print-on-Demand-Ausgabe über Amazon KDP.

Die wahre Herausforderung in der Datenwissenschaft liegt nicht in der Analyse, sondern darin, die Daten so aufzubereiten, dass die Analyse möglich wird.

– Inspiriert von Wickham und Patil, keine direkte Quelle

Vorwort

Die Welt der Technologie befindet sich in einem rasanten Wandel. Künstliche Intelligenz (KI), insbesondere Large Language Models (LLMs), verändert derzeit nahezu jeden Bereich unseres Lebens – von der Art und Weise, wie wir kommunizieren, bis hin zu den Methoden, mit denen wir komplexe Probleme lösen. Diese Entwicklungen schreiten unaufhaltsam voran, angetrieben durch immer leistungsfähigere Algorithmen, größere Rechenkapazitäten und innovative Ansätze. Doch inmitten dieses technologischen Fortschritts bleibt eine Konstante bestehen: die Daten. Sie sind das Fundament, auf dem jede Analyse, jede Vorhersage und jede Entscheidung ruht. Ohne qualitativ hochwertige Daten verlieren selbst die fortschrittlichsten Modelle ihre Wirksamkeit.

In diesem Buch stehen die Daten im Mittelpunkt. Wir widmen uns der Frage, wie ihre Qualität überprüft, gesichert und verbessert werden kann. Dabei betrachten wir nicht nur strukturierte Datenbanken, sondern auch Daten, die in alltäglichen Dokumenten verborgen liegen – sei es in Word-Dokumenten oder in den oft unterschätzten Excel-Tabellen von Mitarbeitenden. Diese Quellen sind in vielen Unternehmen allgegenwärtig, doch ihre Qualität wird selten systematisch hinterfragt.

Die vorliegende Arbeit gliedert sich in zwei große Teile: einen qualitativen Ansatz, der die Konzepte und Strukturen der Datenqualität beleuchtet, und einen quantitativen Ansatz, der sich mit der Messung und Analyse beschäftigt. Von den Grundlagen der Datenqualität über Daten-Governance bis hin zu maschinellem Lernen bieten wir einen umfassenden Überblick über Methoden und Herausforderungen. Dabei gehen wir einen neuen Weg: Anstelle klassischer Quellcodes setzen wir auf die Kraft der Sprache und geben Prompts für Sprachmodelle an. Diese ermöglichen es, die vorgestellten Konzepte flexibel und interaktiv umzusetzen – ein Ansatz, der die Möglichkeiten moderner KI nutzt und gleichzeitig die Bedeutung der Daten als Ausgangspunkt unterstreicht.

Dieses Buch richtet sich an alle, die Daten nicht nur als Ressource, sondern als Verantwortung begreifen. Es ist eine Einladung, die Qualität der eigenen Daten zu hinterfragen und die Werkzeuge an die Hand zu nehmen, um sie zu verbessern – heute und in der sich stetig wandelnden Zukunft.

Andreas Igl und Josef Gruber
Oktober 2025

Inhaltsverzeichnis

1 Einleitung

1.1 Die Bedeutung der Datenqualität

In einer zunehmend datengetriebenen Welt ist Datenqualität nicht nur ein technisches Detail, sondern ein entscheidender Erfolgsfaktor für Unternehmen, wissenschaftliche Forschung und gesellschaftliche Entwicklungen. Entscheidungen basieren auf Daten, Prozesse werden durch Daten optimiert, und Innovationen – insbesondere im Bereich der Künstlichen Intelligenz (KI) – stehen und fallen mit der Qualität der zugrunde liegenden Informationen.

Beispiele aus der Praxis verdeutlichen die Relevanz: Fehlerhafte Daten können teuer werden.

Ein Beispiel: Ein Einzelhändler, der auf fehlerhafte Lagerbestandsdaten vertraut, riskiert Überverkäufe oder unnötige Nachbestellungen; ein Gesundheitsdienstleister mit inkonsistenten Patientendaten gefährdet die Behandlungsqualität. Die Konsequenzen reichen von finanziellen Verlusten bis hin zu Vertrauenskrisen. Doch Datenqualität ist mehr als ein Mittel zur Vermeidung von Problemen – sie ist der wichtigste Baustein für moderne KI-Anwendungen und damit für Effizienz und Innovation.

„Garbage in, garbage out" – ein bekanntes Sprichwort in der Datenwelt.

Mit einer sauberen Datenstruktur und hoher Datenqualität können KI-Lösungen schnell und präzise implementiert werden, während schlechte Daten selbst die fortschrittlichsten Algorithmen ineffektiv machen. Angesichts der rasanten Entwicklungen im KI-Bereich wird deutlich: Ein Fokus auf Datenqualität ist wichtiger als auf unzuverlässigen Daten eine KI aufzusetzen. KI ist nur so gut wie die Daten, auf denen sie trainiert wird. Ein Modell, das auf inkonsistenten oder fehlerhaften Daten basiert, wird bestenfalls ineffizient, schlimmstenfalls irreführend sein.

Die Objekt-Analogie stammt aus der Programmierung und hilft, Datenqualität greifbar zu machen.

Eine weitere Perspektive unterstreicht diese Bedeutung: Daten lassen sich als Objekte betrachten, deren Attribute die eigentlichen Informationen darstellen und deren Methoden – in diesem Fall KI-Algorithmen – auf diesen Objekten operieren. Ein Objekt muss eine sinnvolle, nachvollziehbare Struktur haben und seine Attribute sollten fehlerfrei und konsistent sein. Nur dann kann eine Methode, sei es ein maschinelles Lernmodell oder eine analytische Anwendung, effizient und erfolgreich angewendet werden. Diese Analogie zeigt, dass Datenqualität nicht nur eine technische Anforderung ist, sondern ein grundlegendes Prinzip für die Nutzung von Daten als Ressource.

1.2 Ziel und Struktur des Buches

Zielgruppe: Datenwissenschaftler, IT-Manager, Analysten.

Dieses Buch widmet sich der umfassenden Betrachtung von Datenqualität aus zwei Perspektiven: einem qualitativen Ansatz, der die konzeptionellen und strukturellen Grundlagen beleuchtet, und einem quantitativen Ansatz, der Methoden zur Messung und Verbesserung der Datenqualität vorstellt. Ziel ist es, Leserinnen und Lesern ein tiefes Verständnis und praktische Werkzeuge an die Hand zu geben, um Datenqualität in ihren Organisationen nachhaltig zu sichern und zu optimieren – insbesondere als Basis für skalierbare KI-Anwendungen.

Im ersten Teil, dem qualitativen Ansatz, untersuchen wir die Rahmenbedingungen und Herausforderungen der Datenqualität: Welche Datenstrukturen fördern oder behindern Qualität? Wie beeinflussen Standards, Muss-Felder, Datendisziplin oder in Excel verteilte Daten die Konsistenz und Nutzbarkeit? Was ist der Unterschied von relationalen Datenbanken zu Non-SQL-Datenbanken wie `MongoDB` und Graphdatenbanken wie `Neo4j`. Wir betrachten zudem Daten-Governance, den Datenlebenszyklus und die Rolle von Datenschutz, um ein ganzheitliches Bild zu zeichnen.

Der quantitative Teil bietet konkrete Werkzeuge für die Praxis.

Der zweite Teil, der quantitative Ansatz, widmet sich der konkreten Analyse und Verbesserung von Datenqualität. Hier stehen Methoden wie Ausreißeranalysen (z. B. LOF, SOMs), die Erkennung unplausibler Daten und der Einsatz von maschinellem Lernen im Fokus. Wir zeigen, wie Qualität messbar wird und wie Unternehmen Daten als verlässliche Grundlage für KI und andere Anwendungen nutzen können.

Mit diesem dualen Ansatz möchten wir nicht nur die theoretischen Fundamente legen, sondern auch praktische Lösungen bieten, die den Leser befähigen, Datenqualität als strategischen Vorteil zu nutzen. In einer Zeit, in der KI die Art und Weise, wie wir Daten verwenden, revolutioniert, ist hochwertige Datenbasis der Schlüssel zu effizienten, skalierbaren und zukunftssicheren Anwendungen.

Angabe von LLM-Prompts zur Generierung der Codes anstelle von Quellcodes.

Anstelle von Programm-Codes geben wir die entsprechenden LLM-Prompts. Dies hat aus unserer Sicht den Vorteil, dass der Code Plattform-unabhängig erzeugt werden kann (z.B. für Python, R oder JavaScript). Zugleich wird im Prompt deutlicher, auf welche Faktoren (z.B. Verteilungsannahmen, Modell, Besonderheiten) das jeweilige Verfahren beruht.

Die Prompts und die jeweiligen Datensätze sind unter www.handbuch-datenqualitaet.de zu finden. Hier werden auch Errata und zusätzliche Informationen zum Buch bereitgestellt.

Teil I: Konzepte und Strukturen der Datenqualität

Einführung in die Datenqualität

2

Im Zeitalter der Digitalisierung sind Daten zu einem der wertvollsten Vermögenswerte für Unternehmen, wissenschaftliche Einrichtungen und die Gesellschaft als Ganzes geworden. Sie bilden die Grundlage für operative Prozesse, strategische Entscheidungen, wissenschaftliche Erkenntnisse und revolutionäre Technologien wie die künstliche Intelligenz (KI). Die Qualität dieser Daten ist jedoch kein Selbstläufer. Das bekannte Prinzip „Garbage In, Garbage Out" (GIGO) verdeutlicht auf prägnante Weise die Kernproblematik: Analysen, Modelle und Entscheidungen können nur so gut sein wie die Daten, auf denen sie basieren. Schlechte Datenqualität führt unweigerlich zu fehlerhaften Ergebnissen, Ineffizienzen, finanziellen Verlusten und einem schwindenden Vertrauen in datengetriebene Ansätze.

Der Begriff GIGO stammt aus der frühen Informatik der 1960er Jahre und wurde erstmals von George Fuechsel, einem IBM-Programmierer, verwendet, um Informatiker auf die Wichtigkeit korrekter Eingabedaten hinzuweisen.

Dieses Kapitel legt das Fundament für ein umfassendes Verständnis von Datenqualität. Es wird definiert, was unter dem Begriff Datenqualität zu verstehen ist, warum sie mehr als nur technische Korrektheit bedeutet und wie sie sich von reinen Daten zu wertvollem Wissen entwickelt. Ferner werden die verschiedenen Perspektiven auf Qualität beleuchtet und die zentralen Einflussfaktoren – Mensch, Prozess und Technologie – analysiert. Abschließend wird die exponentiell wachsende Bedeutung der Datenqualität im Kontext der künstlichen Intelligenz erörtert, die die Anforderungen an die Verlässlichkeit und Integrität von Daten auf ein neues Niveau hebt.

In der Antike warnte Aristoteles bereits vor unzuverlässigen Daten: „Der Anfang ist mehr als die Hälfte des Ganzen", was auf die Wichtigkeit einer soliden Grundlage hinweist.

2.1 Was ist Datenqualität?

Der Begriff „Datenqualität" wird oft intuitiv verstanden, doch eine präzise Definition ist für ein systematisches Management unerlässlich. Es handelt sich nicht um eine einzelne, absolute Eigenschaft, sondern um ein vielschichtiges Konstrukt, das in verschiedenen Kontexten unterschiedliche Bedeutungen annehmen kann. Um die Rolle der Datenqualität zu erfassen, ist es hilfreich, zunächst zu verstehen, wie aus rohen Daten wertvolle Erkenntnisse entstehen.

2.1.1 Von Daten zu Wissen: Die DIKW-Pyramide

Ein zentrales Modell zum Verständnis des Werts von Daten ist die DIKW-Pyramide, die die hierarchische Beziehung zwischen Daten (Data), Informationen (Information), Wissen (Knowledge) und Weisheit (Wisdom) beschreibt. Jede Stufe baut auf der vorhergehenden auf und stellt eine höhere Veredelungs- und Kontextebene dar.

Die Ursprünge der DIKW-Hierarchie sind umstritten. Oft wird sie dem Dichter T.S. Eliot und seinem Gedicht „The Rock" (1934) zugeschrieben, obwohl sie dort eher als Frage formuliert ist. Der Organisationstheoretiker Russell L. Ackoff gilt als einer der Ersten, der die Hierarchie 1989 explizit in einem Managementkontext beschrieb.

Auf der untersten Ebene stehen die **Daten**. Sie sind rohe, unstrukturierte Fakten, Symbole und Signale ohne inhärenten Kontext oder Bedeutung. Ein Beispiel wäre die Zeichenfolge „19041990". Für sich genommen ist dieser Wert nicht aussagekräftig. Datenqualität beginnt bereits hier: Ist die Zeichenfolge korrekt erfasst? Fehlen Ziffern? Enthält sie unzulässige Zeichen?

Ein frühes Beispiel für Datenqualitätsprobleme findet sich in der Geschichte der Schifffahrt: Ungenaue Seekarten führten zu unzähligen Schiffbrüchen bis zur Erfindung präziser Messinstrumente im 18. Jahrhundert.

Werden diese Daten in einen Kontext gesetzt, entsteht **Information**. Die Daten werden organisiert und interpretiert. Aus „19041990" wird in Verbindung mit dem Feldnamen „Geburtsdatum" die Information „Das Geburtsdatum ist der 19. April 1990". Die Qualität der Information hängt direkt von der Qualität der zugrunde liegenden Daten und der korrekten Zuordnung des Kontexts ab.

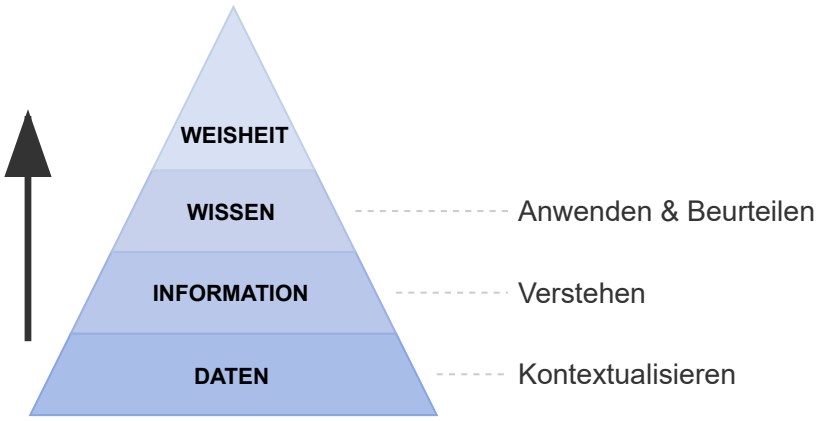


Abbildung 2.1: Die DIKW-Pyramide illustriert die Transformation von rohen Daten zu handlungsleitender Weisheit. Datenqualität ist das Fundament dieser Pyramide.

Wissen entsteht durch die Anwendung und Vernetzung von Informationen. Es repräsentiert das Verständnis von Mustern, Prinzipien und Zusammenhängen. Beispielsweise könnte die Analyse vieler Kundengeburtsdaten (Informationen) zu dem Wissen führen, dass Kunden einer bestimmten Altersgruppe besonders affin für ein neues Produkt sind. Dieses Wissen ermöglicht Vorhersagen und die Ableitung von Handlungsoptionen.

Die Spitze der Pyramide bildet die **Weisheit**. Sie ist die Fähigkeit, Wissen unter Berücksichtigung ethischer, sozialer und langfristiger Konsequenzen anzuwenden und fundierte Urteile zu fällen. Basierend auf dem Wissen über die Zielgruppe könnte eine weise Entscheidung darin bestehen, eine Marketingkampagne nicht nur auf maximalen kurzfristigen Umsatz auszurichten, sondern auch auf den Aufbau langfristiger Kundenbeziehungen und Markenintegrität.

Manche Modelle fügen eine Stufe zwischen Wissen und Weisheit ein: „Verständnis“ (Understanding), das die Fähigkeit zum Erklären („warum?“) betont, während Wissen sich auf das Handeln („wie?“) konzentriert.

Die DIKW-Pyramide macht deutlich: Hohe Datenqualität ist keine akademische Übung, sondern das unverzichtbare Fundament für jede höhere Stufe der Wertschöpfung. Mängel auf der Datenebene pflanzen sich unweigerlich nach oben fort und untergraben die Verlässlichkeit von Informationen, die Validität von Wissen und die Fundiertheit von weisen Entscheidungen.

Die Pyramide wird manchmal erweitert um eine Basisstufe „Signale“, die rohe sensorische Daten darstellen, bevor sie zu nutzbaren Daten werden. In der modernen Big Data-Ära wird die Pyramide manchmal um „Big Data“erweitert, um den Umgang mit massiven Datenmengen zu betonen.

2.1.2 Formale Definition: Datenqualität als “Fitness for Purpose“

Während die DIKW-Pyramide die Rolle der Datenqualität illustriert, bedarf es einer formaleren Definition für die praktische Anwendung. Eine der etabliertesten und praxistauglichsten Definitionen beschreibt Datenqualität als **„Fitness for Purpose“** – die Eignung für den Verwendungszweck. Dieser Ansatz betont die Kontext- und Anwendungsabhängigkeit von Qualität. Es gibt keinen absoluten, universellen Standard für „gute Daten“. Daten, die für einen Zweck von exzellenter Qualität sind, können für einen anderen Zweck völlig ungeeignet sein. Diese Perspektive markiert eine Abkehr von einer rein technischen, systemzentrierten Sichtweise, die Qualität oft mit dem Fehlen von technischen Fehlern (z.B. ‘NULL‘-Werten oder Formatverletzungen) gleichsetzt.

Das Konzept der „Fitness for Use“ wurde maßgeblich vom Qualitätsmanagement-Pionier Joseph M. Juran geprägt. Er argumentierte, dass Qualität nicht absolut ist, sondern immer aus der Perspektive des Kunden bzw. Anwenders definiert werden muss.

Datenqualität ist der Grad, in dem ein Satz von Datencharakteristika die expliziten und impliziten Anforderungen für einen bestimmten Verwendungszweck erfüllt. Qualität ist somit relativ zum Kontext und wird durch die Perspektive des Datennutzers bestimmt.

ISO 9000 definiert Qualität ähnlich als den Grad, zu dem ein Satz inhärenter Merkmale Anforderungen erfüllt, was Parallelen zur Fitness for Purpose zieht.

Die Relevanz dieses Verständnisses ist in Wirtschaft und Wissenschaft enorm. Unternehmen verlassen sich auf Daten

für Kundenbeziehungsmanagement, Lieferkettenoptimierung, Finanz-Reporting und strategische Planung. Mangelhafte Datenqualität kann hier zu falschen Marktanalysen, ineffizienten Prozessen, verpassten Verkaufschancen und Compliance-Verstößen führen. In der Wissenschaft, insbesondere im Bereich der künstlichen Intelligenz, ist die Qualität der Trainings-, Test- und Validierungsdaten entscheidend für die Leistungsfähigkeit, Fairness und Sicherheit von Modellen.

Eine berühmte Anekdote zur Datenqualität ist der Mars Climate Orbiter der NASA, der 1999 verloren ging. Ursache war ein Softwarefehler, der aus der Verwechslung von metrischen (Newton-Sekunden) und imperialen (Pfund-Sekunden) Einheiten resultierte. Die Daten waren präzise, aber nicht genau im richtigen Einheitensystem ([1]).

Beispiel: Qualität einer Adressliste

Betrachtet wird eine Kundendatendatei 'C:\daten.csv' mit Adressinformationen.

- **Zweck 1: Marketing-Postversand.** Für den Versand eines Werbeflyers wird eine Adresse wie „Maximillianstr. 10“ statt „Maximilianstraße 10“ wahrscheinlich trotzdem erfolgreich zugestellt. Die Daten sind für diesen Zweck also „fit“, obwohl sie einen leichten Genauigkeitsfehler aufweisen. Die Datenqualität ist ausreichend.
- **Zweck 2: Geokodierung für Routenplanung.** Ein Logistikunternehmen möchte die Adressen in exakte Längen- und Breitengrade umwandeln, um die Routen seiner Lieferfahrzeuge zu optimieren. Hier kann die leichte Abweichung „Maximillianstr.“ dazu führen, dass der Geokodierungsdienst die Adresse nicht findet. Für diesen Zweck sind die Daten nicht „fit“. Die Datenqualität ist unzureichend.
- **Zweck 3: Dublettenabgleich.** Ein System soll doppelte Kundeneinträge identifizieren. Eine exakte Übereinstimmung würde „Maximilianstraße 10“ und „Maximillianstr. 10“ als unterschiedliche Adressen werten. Ohne intelligente Bereinigungsalgorithmen sind die Daten für diesen Zweck nur bedingt „fit“.

Dieses Beispiel zeigt, dass dieselben Daten je nach Anwendungsfall eine hohe oder niedrige Qualität aufweisen können.

Schätzungen zufolge verursachen schlechte Datenqualität bei Unternehmen jährlich Kosten von 15% bis 25% ihres Umsatzes. Diese Kosten entstehen durch fehlerhafte Entscheidungen, Prozessineffizienzen und Nachbesserungsaufwände ([2]).

2.2 Objektive vs. subjektive Qualitätswahrnehmung

Die Bewertung der Datenqualität nach dem „Fitness for Purpose“-Prinzip erfordert die Berücksichtigung von zwei unterschiedlichen, aber gleichermaßen wichtigen Perspektiven: der objektiven und der subjektiven Qualitätswahrnehmung.

Die **objektive Datenqualität** bezieht sich auf messbare, quantifizierbare Kriterien, die oft system- oder datenzentriert sind.

Diese Kriterien können durch Algorithmen und definierte Regeln überprüft werden, ohne dass eine menschliche Interpretation erforderlich ist. Typische objektive Metriken, die in späteren Kapiteln wie Kapitel 3 detailliert werden, umfassen:

- **Vollständigkeit (Completeness):** Der Anteil nicht-leerer Werte. Eine einfache Formel zur Messung ist

$$S_{\text{vollständig}} = 1 - \frac{N_{\text{fehlend}}}{N_{\text{gesamt}}}.$$

- **Gültigkeit (Validity):** Der Anteil der Werte, der vordefinierten Formaten oder Wertebereichen entspricht (z.B. eine E-Mail-Adresse, die das '@'-Zeichen enthält).
- **Eindeutigkeit (Uniqueness):** Das Fehlen von Duplikaten, die dasselbe reale Objekt repräsentieren.

Diese Metriken liefern eine harte, nachvollziehbare Grundlage für die Qualitätsbewertung und sind essenziell für die technische Überwachung von Datenbeständen.

Demgegenüber steht die **subjektive Datenqualität**. Sie beschreibt die Wahrnehmung und Einschätzung der Qualität durch den Datennutzer. Diese Perspektive ist inhärent kontextbezogen und an die individuellen Erfahrungen, Erwartungen und Aufgaben des Nutzers gebunden. Wichtige Aspekte der subjektiven Qualität sind:

In der Praxis klaffen objektive und subjektive Wahrnehmung oft auseinander. Eine Abteilung mag ihre Daten als „zu 99,9% vollständig" (objektiv) bezeichnen, während der Anwender bemängelt, dass ausgerechnet das für ihn entscheidende Feld häufig leer ist (subjektiv unzureichend).

- **Glaubwürdigkeit (Believability):** Das Vertrauen des Nutzers in die Daten und deren Quelle.
- **Nützlichkeit (Utility):** Die Eignung der Daten zur Erfüllung einer spezifischen Aufgabe des Nutzers.
- **Verständlichkeit (Interpretability):** Die Fähigkeit des Nutzers, die Daten und ihre Bedeutung ohne großen Aufwand zu verstehen.

Diese Kriterien sind schwerer zu quantifizieren und werden oft durch Umfragen, Interviews oder Feedbackschleifen erfasst.

Psychologische Studien zeigen, dass subjektives Vertrauen in Daten durch Transparenz über die Herkunft gesteigert werden kann.

Ein ganzheitliches Datenqualitätsmanagement muss beide Perspektiven berücksichtigen. Rein objektiv hochwertige Daten, denen die Anwender nicht vertrauen, werden nicht genutzt und stiften somit keinen Wert. Umgekehrt können Daten, die von Nutzern als glaubwürdig empfunden werden, aber objektiv fehlerhaft sind, zu katastrophalen Fehlentscheidungen führen. Das Ziel muss es sein, eine hohe

In der Medizin kann subjektive Wahrnehmung lebensrettend sein: Ärzte ignorieren manchmal objektiv "perfekte" Daten, wenn sie intuitiv Ungereimtheiten spüren.

objektive Qualität zu erreichen und diese so an die Nutzer zu kommunizieren, dass auch eine hohe subjektive Qualität wahrgenommen wird.

To Do Assessment der Qualitätsperspektiven

In einem Datenqualitätsprojekt sollten beide Dimensionen erfasst werden:

1. **Objektive Analyse:** Führe ein Datenprofiling (siehe Kapitel u.a. 9) für die kritischen Datenobjekte durch. Messe Kennzahlen wie Vollständigkeit, Eindeutigkeit und Gültigkeit. Dokumentiere die Ergebnisse in einem Datenqualitäts-Dashboard.
2. **Subjektive Analyse:** Führe kurze Interviews oder eine standardisierte Umfrage mit den Hauptnutzern der Daten durch. Frage nach deren Vertrauen in die Daten, den häufigsten Problemen im Arbeitsalltag und der wahrgenommenen Nützlichkeit für ihre Aufgaben.
3. **Synthese:** Vergleiche die Ergebnisse. Wo decken sich die Wahrnehmungen? Wo gibt es Diskrepanzen? Nutze die Erkenntnisse, um Prioritäten für Verbesserungsmaßnahmen zu setzen.

2.3 Die zentralen Einflussfaktoren: Mensch, Prozess, Technologie

Datenqualität entsteht nicht im luftleeren Raum. Sie ist das Ergebnis eines komplexen Zusammenspiels dreier zentraler Säulen: Mensch, Prozess und Technologie. Probleme in der Datenqualität lassen sich fast immer auf Defizite in einem oder mehreren dieser Bereiche zurückführen. Ein erfolgreiches Management muss daher alle drei Faktoren adressieren.

Ein klassischer Datenqualitätsfehler durch den Faktor *Mensch* ist die falsche Dateneingabe – etwa Tippfehler, Zahlendreher oder versehentlich ausgelassene Werte. Solche Fehler entstehen häufig unter *Zeitdruck* oder bei fehlenden Plausibilitätsprüfungen im Eingabesystem.

Der Mensch (People) ist oft die entscheidende, aber auch variabelste Komponente. Die Fähigkeiten, das Bewusstsein und die Sorgfalt der Mitarbeiter, die Daten erstellen, bearbeiten und nutzen, sind von fundamentaler Bedeutung.

- *Positive Einflüsse:* Hohe Datenkompetenz („Data Literacy“), ein starkes Verantwortungsbewusstsein (z.B. durch definierte „Data Stewards“), Motivation und zielgerichtete Schulungen.
- *Negative Einflüsse:* Mangelndes Verständnis für die Auswirkungen von Fehlern, unzureichende Schulung, Zeitdruck, der zu fehlerhaften Eingaben führt, oder fehlende Anreize für eine hohe Datenpflege.

Der **Prozess (Process)** umfasst alle definierten Arbeitsabläufe, Regeln und Standards, die den Umgang mit Daten steuern. Gut definierte Prozesse schaffen den Rahmen für konsistente und qualitativ hochwertige Datenergebnisse.

Oft wird versucht, Datenqualitätsprobleme allein durch den Kauf eines neuen Tools (Technologie) zu lösen. Ohne die Anpassung der Prozesse und die Schulung der Menschen ist ein solches Vorhaben jedoch fast immer zum Scheitern verurteilt.

- *Positive Einflüsse:* Klare Data-Governance-Richtlinien (siehe Kapitel 4), automatisierte Validierungsregeln bei der Dateneingabe, etablierte Freigabeprozesse und regelmäßige Datenqualitäts-Audits.
- *Negative Einflüsse:* Fehlende oder unklare Prozessdefinitionen, Medienbrüche (z.B. manuelle Übertragung von Daten aus E-Mails), fehlende Kontrollmechanismen und unklare Verantwortlichkeiten.

Die **Technologie (Technology)** stellt die Werkzeuge und die Infrastruktur für die Datenverarbeitung und -haltung bereit. Sie kann die Einhaltung von Qualitätsstandards erzwingen und unterstützen, aber auch die Quelle von Problemen sein.

Das Toyota-Produktionssystem zeigt, wie Prozesse durch kontinuierliche Verbesserung (Kaizen) Qualität steigern können – ein Prinzip, das auf Datenprozesse übertragbar ist.

- *Positive Einflüsse:* Moderne Datenbanksysteme mit starken Integritätsprüfungen (Constraints), Master-Data-Management-Systeme zur Sicherung einer „Single Source of Truth“, spezialisierte Datenqualitäts-Tools und eine flexible Datenarchitektur (siehe Kapitel 6).
- *Negative Einflüsse:* Veraltete Legacy-Systeme mit starren Strukturen, unzureichend integrierte Anwendungslandschaften (Silos), fehlende Automatisierung und mangelhafte System-Performance.

Diese drei Säulen sind untrennbar miteinander verbunden. Die beste Technologie ist nutzlos, wenn die Menschen nicht geschult sind (Mensch-Technologie-Interaktion) oder die Prozesse ihre Nutzung nicht vorsehen (Prozess-Technologie-Interaktion). Umgekehrt können die besten Prozesse an mangelhafter technologischer Unterstützung oder fehlender Akzeptanz bei den Mitarbeitern scheitern. Nachhaltige Verbesserungen der Datenqualität erfordern daher einen ganzheitlichen Ansatz, der alle drei Bereiche in den Blick nimmt.

Die Bedeutung der drei Säulen kann sich je nach Branche unterscheiden. In stark regulierten Bereichen wie der Pharmazie oder dem Finanzwesen spielen Prozesse eine überragende Rolle, um Compliance sicherzustellen.

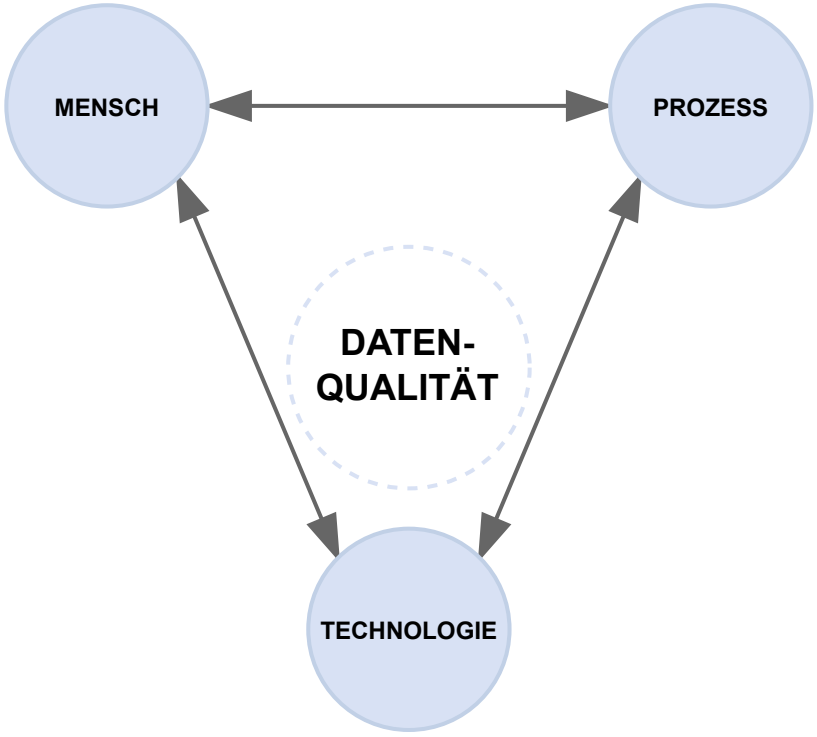


Abbildung 2.2: Das M-P-T-Modell: Datenqualität als Ergebnis des ausgewogenen Zusammenspiels von Mensch, Prozess und Technologie.

2.4 Steigende Bedeutung von Datenqualität im KI-Zeitalter

Während Datenqualität schon immer wichtig war, hat ihre Bedeutung mit dem Aufstieg der künstlichen Intelligenz (KI) und des maschinellen Lernens (ML) eine neue, kritische Dimension erreicht. KI-Modelle sind keine magischen Black Boxes; sie lernen Muster, Zusammenhänge und Verhaltensweisen direkt aus den Daten, mit denen sie trainiert werden. Die Qualität dieser Trainingsdaten ist der mit Abstand wichtigste Faktor für die Leistungsfähigkeit und Zuverlässigkeit eines KI-Systems.

Eine Studie von Google ([3]) zeigte, dass die Verfügbarkeit von mehr (hochwertigen) Daten für das Training eines Modells oft einen größeren positiven Effekt auf die Genauigkeit hat als die Wahl eines komplexeren Algorithmus. Daten sind der Treibstoff der KI.

Das Prinzip „Garbage In, Garbage Out“ potenziert sich im KI-Kontext zu „Garbage In, Garbage Out at Scale“. Ein auf schlechten Daten trainiertes Modell wird nicht nur eine falsche Vorhersage treffen, sondern systematisch und automatisiert tausende oder millionenfache falsche Entscheidungen treffen, oft mit weitreichenden finanziellen oder ethischen Konsequenzen.

Ein zentrales Problemfeld ist **Bias und Fairness**. Wenn die Trainingsdaten historische Vorurteile oder gesellschaftliche Ungleichheiten widerspiegeln, wird das KI-Modell diese Verzerrungen (Bias) lernen und möglicherweise sogar verstärken. Beispielsweise könnte ein Kreditvergabe-Modell, das

mit historisch diskriminierenden Daten trainiert wurde, bestimmte Bevölkerungsgruppen systematisch benachteiligen. Die Sicherstellung von Repräsentativität und die Vermeidung von Bias in den Daten wird somit zu einer zentralen ethischen und qualitativen Anforderung (siehe Kapitel 7).

Beispiel: Bias in Einstellungsalgorithmen

Ein Technologieunternehmen entwickelt ein KI-Tool, das Lebensläufe vorsortieren soll. Das Modell wird mit den Daten der erfolgreichen Bewerber der letzten zehn Jahre trainiert. In dieser Zeit waren jedoch überwiegend Männer in technischen Positionen beschäftigt. Das Modell lernt fälschlicherweise, dass Merkmale, die häufiger bei männlichen Bewerbern vorkommen (z.B. bestimmte Formulierungen oder besuchte Universitäten), Prädiktoren für Erfolg sind. Als Ergebnis stuft es die Lebensläufe qualifizierter weiblicher Bewerber systematisch herab. Das Problem liegt nicht im Algorithmus selbst, sondern in der mangelnden Qualität (Repräsentativität) der Trainingsdaten.

Der COMPAS-Algorithmus ist ein proprietärer Risiko-Bewertungsalgorithmus, der in den USA eingesetzt wird, um die Wahrscheinlichkeit eines Rückfalls bei Straftätern vorherzusagen. Er wurde stark kritisiert, da er Bias gegen afroamerikanische Häftlinge zeigte, basierend auf verzerrten Trainingsdaten ([4]).

Gleichzeitig bietet die KI auch neue Lösungsansätze für das Datenqualitätsmanagement. KI-gestützte Verfahren können zur **automatisierten Datenbereinigung** eingesetzt werden. Anstatt auf starre, manuell definierte Regeln angewiesen zu sein, können ML-Modelle lernen, Anomalien, Inkonsistenzen und wahrscheinliche Fehler in großen Datenmengen selbstständig zu erkennen und Korrekturvorschläge zu generieren.

Tools wie Google Data Commons nutzen KI, um öffentliche Datensätze zu bereinigen und zu verknüpfen.

Darüber hinaus sind **Metadaten** und **Transparenz** entscheidende Enabler für vertrauenswürdige KI. Um die Entscheidungen eines Modells nachvollziehen und validieren zu können (Explainable AI, XAI), ist es unerlässlich zu wissen, woher die Daten stammen (Datenherkunft, „Data Lineage“), wie sie definiert sind und welche Transformationen sie durchlaufen haben. Ein gutes Metadaten-Management (siehe Kapitel 5) ist daher die Voraussetzung für eine verantwortungsvolle KI-Entwicklung und -Nutzung.

Die EU-KI-Verordnung fordert Transparenz über Trainingsdaten, um Bias zu vermeiden und Rechenschaft zu gewährleisten.

2.5 Zusammenfassung

Dieses Kapitel hat die fundamentalen Konzepte der Datenqualität eingeführt und ihre zentrale Bedeutung in der heutigen datengetriebenen Welt verdeutlicht. Die Qualität von Daten ist eine entscheidende Voraussetzung für die Umwandlung roher Fakten in wertstiftende Informationen, handlungsleitendes Wissen und fundierte Weisheit, wie es die **DIKW-Pyramide** illustriert.

Die zentrale Erkenntnis ist, dass Datenqualität nicht als absolute, technische Größe verstanden werden darf, sondern als relative **„Fitness for Purpose"** – die Eignung für einen spezifischen Verwendungszweck. Diese Sichtweise erfordert die Berücksichtigung von **objektiv messbaren Kriterien** (z.B. Vollständigkeit, Gültigkeit) und der **subjektiven Wahrnehmung der Nutzer** (z.B. Vertrauen, Nützlichkeit). Ein ganzheitlicher Ansatz muss beide Perspektiven in Einklang bringen.

Nachhaltige Datenqualität kann nur durch das ausgewogene Zusammenspiel der drei Einflussfaktoren **Mensch, Prozess und Technologie** erreicht werden. Technologische Lösungen allein reichen nicht aus; sie müssen durch klare Prozesse und die Kompetenz und das Bewusstsein der Mitarbeiter ergänzt werden.

Oft wird versucht, auf schlechte Daten und Prozesse moderne Algorithmen für "Quick-Wins" anzuwenden. Hier ist ein Scheitern vorprogrammiert. Ohne genaue Daten- und Prozessbeschreibungen versagen auch die besten KI-Modelle.

Im Zeitalter der **künstlichen Intelligenz** hat die Bedeutung von Datenqualität eine neue Eskalationsstufe erreicht. Sie ist das Fundament für leistungsfähige, faire und vertrauenswürdige KI-Systeme. Probleme wie Daten-Bias können zu schwerwiegenden gesellschaftlichen und wirtschaftlichen Folgen führen, während KI-Techniken umgekehrt auch neue Möglichkeiten zur automatisierten Erkennung und Behebung von Qualitätsproblemen bieten.

3 Dimensionen der Datenqualität

Die Qualität von Daten ist kein monolithisches, einzelnes Konzept, sondern ein mehrdimensionales Konstrukt. Um Datenqualität systematisch zu bewerten, zu steuern und zu verbessern, hat sich in Wissenschaft und Praxis die Verwendung eines Frameworks von Qualitätsdimensionen etabliert. Diese Dimensionen dienen als standardisierte Kriterien, um spezifische Aspekte der Datenqualität zu analysieren und messbar zu machen. Sie bieten eine gemeinsame Sprache für Fachexperten, IT-Spezialisten und Entscheidungsträger. In diesem Kapitel werden die zentralen und gängigsten Dimensionen der Datenqualität detailliert vorgestellt, ihre Messbarkeit erörtert und ihre Bedeutung im Kontext des gesamten Datenlebenszyklus beleuchtet. Die Auswahl und Priorisierung der relevanten Dimensionen ist stets kontextabhängig und richtet sich nach dem spezifischen Verwendungszweck der Daten – ein Prinzip, das als „Fitness for Purpose" bekannt ist und im Kapitel 2 eingeführt wurde. Dieses Framework ermöglicht eine strukturierte Herangehensweise an Datenqualitätsprobleme.

Die Idee der Qualitätsdimensionen stammt ursprünglich aus der Fertigungsqualität, wo W. Edwards Deming in den 1950er Jahren multidimensionale Ansätze einführte, die später auf Daten übertragen wurden.

3.1 Kern-Dimensionen

Die Kern-Dimensionen bilden das Fundament der meisten Datenqualitätsinitiativen. Sie sind oft intuitiv verständlich und ihre Messung ist in der Regel mit etablierten Methoden gut umsetzbar. Ihre Beherrschung ist eine Voraussetzung für praktisch jede datengetriebene Anwendung. Diese Dimensionen werden häufig in DQ-Tools integriert.

In vielen Organisationen werden Kern-Dimensionen in Dashboards visualisiert, um Echtzeit-Überwachung zu ermöglichen.

3.1.1 Vollständigkeit (Completeness)

Die Vollständigkeit ist eine der fundamentalsten Dimensionen der Datenqualität. Sie befasst sich mit dem Vorhandensein oder Fehlen von Datenwerten. Unvollständige Daten können Analysen verzerren, Prozesse unterbrechen und die Aussagekraft von Modellen erheblich schwächen.

Die **Vollständigkeit (Completeness)** beschreibt das Ausmaß, in dem erwartete Daten in einem Datensatz vorhanden sind. Sie misst den Anteil der nicht-leeren Werte im Verhältnis zur Gesamtzahl der er-

warteten Werte. Fehlende Werte werden oft als `NULL`, `NA` oder leere Zeichenketten repräsentiert.

Das Konzept der „Closed World Assumption“ geht davon aus, dass alles, was nicht in der Datenbank steht, nicht existiert oder falsch ist. Im Gegensatz dazu lässt die „Open World Assumption“ die Möglichkeit offen, dass Informationen außerhalb der Datenbank existieren, aber unbekannt sind. Dies hat massive Implikationen für die Interpretation fehlender Werte.

Die Messung der Vollständigkeit kann auf verschiedenen Ebenen erfolgen:

1. **Attribut-Ebene:** Misst die Vollständigkeit einer einzelnen Spalte (Attribut).
2. **Datensatz-Ebene (Tupel-Ebene):** Misst, wie viele Attribute innerhalb eines einzelnen Datensatzes (Zeile) gefüllt sind.
3. **Populations-Ebene:** Misst, ob alle relevanten Objekte der realen Welt (z.B. alle Kunden) im Datensatz repräsentiert sind. Dies ist oft am schwierigsten zu ermitteln.

Mathematisch wird der Vollständigkeitsgrad S_{compl} für ein Attribut oft wie folgt berechnet. Sei N die Gesamtzahl der Datensätze und N_{missing} die Anzahl der fehlenden Werte für das betreffende Attribut. Dann gilt:

$$S_{\text{compl}} = \frac{N - N_{\text{missing}}}{N} = 1 - \frac{N_{\text{missing}}}{N}$$

Ein Score von 1 steht für vollständige Daten, ein Score von 0 für vollständig leere Daten.

Statistiker unterscheiden drei Typen fehlender Daten: MCAR (Missing Completely At Random), MAR (Missing At Random) und MNAR (Missing Not At Random). Die Art der fehlenden Daten bestimmt, welche Imputationsstrategien zulässig sind, ohne die Ergebnisse zu verzerren.

Beispiel: Vollständigkeit in Kundendaten

Eine Kundendatenbank mit 10.000 Einträgen enthält die Felder `Name`, `Adresse` und `Telefonnummer`. Eine Analyse ergibt:

- Das Feld `Name` ist zu 100
- Das Feld `Adresse` hat 200 fehlende Einträge ($S_{compl} = 1 - 200/10000 = 0,98$).
- Das Feld `Telefonnummer`, ein optionales Feld bei der Erfassung, hat 3.500 fehlende Einträge ($S_{compl} = 1 - 3500/10000 = 0,65$).

Während die Adressdaten eine hohe Vollständigkeit aufweisen, ist die Vollständigkeit der Telefonnummern für eine geplante Telefonmarketing-Aktion möglicherweise unzureichend.

Vollständigkeit wurde in frühen Datenbanken wie dBASE durch erforderliche Felder erzwungen, was ein Meilenstein in der DQ-Geschichte war.

To Do Vollständigkeitsanalyse

Führen Sie eine Vollständigkeitsanalyse für alle Tabellen in Ihrer Produktionsdatenbank durch. Identifizieren Sie alle Attribute mit einem Vollständigkeitsgrad unter 95%. Untersuchen Sie für die Top-10-Fälle die Ursachen der fehlenden Werte. Handelt es sich um optionale Felder, technische Probleme oder Prozesslücken? Dokumentieren Sie die Ergebnisse und leiten Sie Maßnahmen zur

Verbesserung der Datenerfassung ab.

3.1.2 Genauigkeit (Accuracy)

Die Genauigkeit ist eine kritische Dimension, da sie die Korrektheit der Daten im Abgleich mit der Realität beschreibt. Daten können vollständig und konsistent sein, aber dennoch schlichtweg falsch.

Genauigkeit unterscheidet sich von Präzision: Genauigkeit bedeutet Nähe zum wahren Wert, Präzision die Konsistenz wiederholter Messungen.

Die **Genauigkeit (Accuracy)** misst die Übereinstimmung von Datenwerten mit einem als korrekt anerkannten Referenzwert oder einer Referenzquelle. Sie beschreibt, wie nah ein Datenwert an der wahren Begebenheit ist.

Die größte Herausforderung bei der Messung der Genauigkeit ist die Verfügbarkeit einer verlässlichen Referenzquelle, oft als „Golden Record“ oder „Ground Truth“ bezeichnet. Diese Referenz kann eine externe maßgebliche Quelle (z.,B. ein staatliches Register) oder ein intern kuratierter und verifizierter Datensatz sein. Die Metrik zur Messung der Genauigkeit S_{acc} hängt vom Datentyp ab. Für kategoriale Daten ist es oft der Anteil der korrekten Werte:

In der Kartographie führte Ungenauigkeit in Karten zu historischen Katastrophen, wie der Verlust von Schiffen durch falsche Seekarten im 18. Jahrhundert.

$$S_{\text{acc}} = \frac{\text{Anzahl korrekter Werte}}{\text{Gesamtzahl der Werte}} = 1 - \text{Fehlerrate}$$

Für numerische Daten können Abstandsmaße wie der mittlere absolute Fehler (MAE) verwendet werden.

Beispiel: Überprüfung von Adressdaten

Ein E-Commerce-Unternehmen möchte die Genauigkeit seiner Kundenadressen sicherstellen, um Lieferprobleme zu minimieren. Es gleicht einen Auszug von 1.000 Adressen aus seiner Datenbank `C:/Daten/daten.csv` mit den Daten eines externen Postdienstleisters (dem Golden Record) ab. Die Analyse ergibt:

- 920 Adressen stimmen exakt überein.
- Bei 50 Adressen gibt es kleine Abweichungen (z.,B. „Strasse“ statt „Str.“), die aber als korrekt gelten.
- Bei 30 Adressen sind die Postleitzahlen oder Straßennamen veraltet oder falsch.

Die Genauigkeit beträgt somit $(920 + 50)/1000 = 0,97$ oder 97

Die Genauigkeit kann sich über die Zeit ändern. Eine Adresse, die heute genau ist, kann nach einem Umzug des Kunden in einem Monat ungenau sein. Dies verdeutlicht die enge Beziehung zwischen Genauigkeit und Aktualität.

3.1.3 Konsistenz (Consistency)

Konsistenz und Genauigkeit sind nicht dasselbe. Eine Datenbank kann konsistent, aber ungenau sein. Beispiel: Wenn das Geburtsdatum aller Mitarbeiter systematisch um ein Jahr falsch erfasst wurde, sind die Daten zwar ungenau, aber möglicherweise konsistent mit anderen Feldern (z. B. dem Eintrittsalter).

Konsistenz bezieht sich auf die Widerspruchsfreiheit von Daten. Inkonsistenzen sind oft ein Indikator für systemische Probleme in der Datenhaltung oder in den Geschäftsprozessen.

Die **Konsistenz (Consistency)** beschreibt die logische Widerspruchsfreiheit von Daten. Daten sind konsistent, wenn sie nicht gegen definierte Geschäftsregeln oder Integritätsbedingungen verstoßen. Dies gilt sowohl innerhalb eines Datensatzes, zwischen verschiedenen Datensätzen als auch über Systemgrenzen hinweg.

Inkonsistenzen können sich auf verschiedene Weisen manifestieren:

- **Strukturelle Inkonsistenz:** Ein Feld, das Postleitzahlen enthalten soll, enthält Werte in unterschiedlichen Formaten (z.,B. „12345“ und „D-12345“).
- **Logische Inkonsistenz:** Ein Datensatz enthält widersprüchliche Informationen. Zum Beispiel ist das Sterbedatum eines Patienten vor seinem Geburtsdatum eingetragen. Ein anderer Fall wäre, wenn ein Kunde in einem System als „aktiv“ und in einem anderen als „gekündigt“ geführt wird.

ACID-Eigenschaften in Datenbanken (Atomicity, Consistency, Isolation, Durability) gewährleisten Konsistenz auf Transaktionsebene.

Die Messung der Konsistenz S_{cons} erfolgt durch die Definition von Geschäftsregeln und die Überprüfung, wie viele Datensätze diese Regeln verletzen:

$$S_{\text{cons}} = 1 - \frac{\text{Anzahl inkonsistenter Datensätze}}{\text{Gesamtzahl der Datensätze}}$$

Beispiel: Logische Widersprüche in Auftragsdaten

In einer Auftragsdatenbank wird folgende Geschäftsregel definiert: „Wenn der Auftragsstatus ’Versendet’ ist, muss das Versanddatum gefüllt sein und darf nicht vor dem Bestelldatum liegen.“ Eine Analyse von 5.000 versendeten Aufträgen ergibt:

- Bei 150 Aufträgen ist das Versanddatum leer.
- Bei 25 Aufträgen liegt das Versanddatum vor dem Bestelldatum.

Insgesamt sind 175 Datensätze inkonsistent. Der Konsistenz-Score bezogen auf diese Regel ist $S_{cons} = 1 - 175/5000 = 0,965$. Diese Inkonsistenzen deuten auf Fehler im Auftragsabwicklungsprozess hin.

Referenzielle Integrität, eine Funktion von relationalen Datenbanken, ist eine technische Form der Konsistenzsicherung. Sie stellt sicher, dass ein Fremdschlüsselwert immer auf einen existierenden Primärschlüssel in einer anderen Tabelle verweist.

In der Blockchain-Technologie wird Konsistenz durch dezentrale Konsensmechanismen wie Proof-of-Work erzwungen.

3.1.4 Aktualität (Timeliness)

In der heutigen schnelllebigen Welt ist der Wert von Daten oft direkt an ihre Aktualität gekoppelt. Veraltete Daten können zu falschen Entscheidungen, verpassten Gelegenheiten und ineffizienten Prozessen führen.

Die **Aktualität (Timeliness)** beschreibt, ob Daten für den beabsichtigten Verwendungszweck ausreichend zeitnah verfügbar sind. Sie umfasst zwei Aspekte: die **Currency**, also wie alt die Daten sind, und die **Volatility**, also wie schnell die Daten veralten.

Der Begriff „Data Half-Life“ (Daten-Halbwertszeit) beschreibt, wie lange es dauert, bis 50% eines Datensatzes veraltet oder ungenau sind. Die Halbwertszeit von Adressdaten wird oft auf nur 2-3 Jahre geschätzt.

Die Anforderungen an die Aktualität variieren stark je nach Anwendungsfall. Börsenkurse müssen in Echtzeit verfügbar sein, während Bevölkerungsstatistiken auch dann noch wertvoll sind, wenn sie mehrere Monate alt sind. Die Messung der Aktualität S_{time} kann über eine Zeitdifferenz erfolgen. Sei T_{event} der Zeitpunkt eines realen Ereignisses, $T_{\text{available}}$ der Zeitpunkt, zu dem die Daten über das Ereignis im System verfügbar sind, und T_{required} der Zeitpunkt, zu dem die Daten für eine Entscheidung benötigt werden. Ein Maß für die Verzögerung (Delay) ist $\Delta T = T_{\text{available}} - T_{\text{event}}$. Ein Score könnte dann über eine Abklingfunktion definiert werden, die den Wert der Information über die Zeit mindert. Zum Beispiel:

$$S_{time} = \max\left(0; 1 - \frac{\text{Alter der Daten}}{\text{Erwartete Lebensdauer}}\right)$$

Beim *Flash Crash* am 6. Mai 2010 kam es zu erheblichen Verzögerungen im *Consolidated Quotation System (CQS)*. Laut dem gemeinsamen Bericht von SEC und CFTC zeigten NYSE-Quotes für über 1,600 Symbole zeitweise Verzögerungen von über 10 Sekunden, was zu einem Aktienrückgang von 9% innerhalb von Minuten führte ([5]).

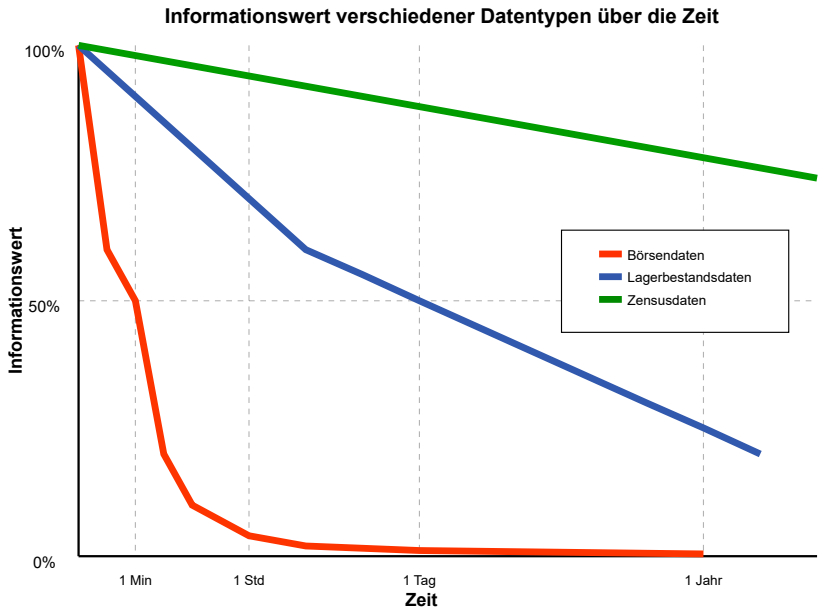


Abbildung 3.1: Veranschaulichung der Aktualität: Der Wert von Informationen nimmt über die Zeit je nach Datentyp unterschiedlich schnell ab.

Beispiel: Aktualität von Lagerbestandsdaten

Ein Online-Händler betreibt einen Webshop, der Kunden den aktuellen Lagerbestand anzeigt. Die Bestandsdaten werden nur einmal täglich um Mitternacht aus dem Warenwirtschaftssystem (WWS) in die Webshop-Datenbank synchronisiert. Ein Produkt wird um 10:00 Uhr morgens im Laden verkauft, aber im Webshop als „verfügbar“ angezeigt. Ein Kunde bestellt es um 11:00 Uhr online. Erst am nächsten Tag wird die Bestellung storniert, was zu Kundenunzufriedenheit führt. Die Aktualität der Daten war für diesen Prozess unzureichend. Eine Echtzeit-Synchronisation wäre erforderlich.

3.2 Strukturelle Dimensionen

Strukturelle Dimensionen sind in semantischen Web-Standards wie RDF entscheidend für interoperable Daten.

Strukturelle Dimensionen beziehen sich auf die Form und Organisation der Daten. Sie sind oft die Voraussetzung dafür, dass Daten überhaupt maschinell verarbeitet und korrekt interpretiert werden können.

3.2.1 Eindeutigkeit (Uniqueness)

Historisch gesehen entstanden viele Duplikate durch die Migration und Zusammenführung von Systemen (z.,B. bei Unternehmensfusionen), bei denen derselbe Kunde in beiden Altsystemen mit leicht unterschiedlichen Daten existierte.

Die Eindeutigkeit stellt sicher, dass jedes reale Objekt in einem Datensatz nur einmal repräsentiert wird. Doppelte Einträge, sogenannte Duplikate, sind eine häufige Ursache für Probleme.

Die **Eindeutigkeit (Uniqueness)** beschreibt die Abwesenheit von Duplikaten in einem Datensatz. Ein Datensatz ist eindeutig, wenn jedes Objekt der realen Welt (z.,B. ein Kunde, ein Produkt) durch genau einen einzigen Datensatz repräsentiert wird.

Levenshtein-Distanz: misst die minimale Anzahl an Einfüge-, Lösch- oder Ersetzoperationen, um einen String in einen anderen zu überführen.

Jaro-Winkler: bewertet die Zeichenübereinstimmung und ihre Reihenfolge; höhere Werte deuten auf größere Ähnlichkeit hin – besonders geeignet für Personennamen.

Duplikate können zu vielfältigen Problemen führen, wie zum Beispiel der mehrfachen Zusendung von Werbematerial an denselben Kunden, verfälschten Analysen (z.,B. Zählung von Kunden) oder inkonsistenten Daten, wenn nur einer der doppelten Einträge aktualisiert wird. Die Herausforderung bei der Duplikaterkennung liegt darin, dass Duplikate selten identisch sind. Sie enthalten oft Tippfehler, Abkürzungen oder unterschiedliche Schreibweisen (z.,B. „Max Mustermann“ und „M. Mustermann“). Man unterscheidet daher:

- **Deterministisches Matching:** Suche nach exakten Übereinstimmungen.
- **Probabilistisches (Fuzzy) Matching:** Suche nach ähnlichen Datensätzen mittels Ähnlichkeitsmaßen (z. B. Levenshtein-Distanz, Jaro-Winkler).

Der Eindeutigkeits-Score S_{uniq} kann als das Verhältnis von eindeutigen Entitäten zur Gesamtzahl der Datensätze definiert werden:

$$S_{\text{uniq}} = \frac{\text{Anzahl der einzigartigen realen Entitäten}}{\text{Gesamtzahl der Datensätze}}$$

Ein Wert von 1 bedeutet, dass keine Duplikate vorhanden sind.

Tools wie OpenRefine nutzen Clustering-Algorithmen für effiziente Duplikaterkennung in großen Datensätzen.

3.2.2 Validität (Validity)

Die Validität, oft auch als Gültigkeit bezeichnet, prüft, ob Daten formalen Vorgaben entsprechen. Sie ist eine grundlegende, syntaktische Prüfung.

Die **Validität (Validity)** beschreibt die Konformität von Datenwerten mit den für sie definierten syntaktischen Regeln. Dazu gehören Format-, Typ- und Wertebereichsvorgaben.

Validität prüft die Form, nicht den Inhalt. Eine Telefonnummer „0123-456789“ kann syntaktisch valide sein (korrektes Format), aber dennoch ungenau, weil sie nicht der Person zugeordnet ist. Ein Wert wie „99“ für das Alter ist valide, weil er im Wertebereich liegt, aber ungenau, wenn die Person erst 30 ist.

Beispiele für Validitätsregeln sind:

- **Datentyp:** Ein Feld für das Alter muss eine Ganzzahl enthalten.
- **Format:** Ein Datum muss im Format JJJJ-MM-TT vorliegen. Eine E-Mail-Adresse muss einem regulären Ausdruck entsprechen.
- **Wertebereich:** Das Alter muss zwischen 0 und 120 liegen. Ein Rabatt darf nicht > 100% sein.
- **Zulässige Werte:** Ein Feld für den Wochentag darf nur Werte aus der Menge Montag, Dienstag, ... enthalten.

Die Messung der Validität S_{valid} ist oft unkompliziert. Es wird der Anteil der Werte gemessen, die den definierten Regeln entsprechen:

$$S_{\text{valid}} = \frac{\text{Anzahl der validen Werte}}{\text{Gesamtzahl der Werte (ohne fehlende)}}$$

Schema-Validierung in XML oder JSON stellt Validität auf Dateiebene sicher und wurde in den 2000er Jahren standardisiert.

Beispiel: Validierung von IBANs

Eine Finanzanwendung verarbeitet IBANs. Jede IBAN muss zwei Kriterien erfüllen: 1. Sie muss dem länderspezifischen Format entsprechen (z. B. 22 Zeichen in Deutschland). 2. Die interne Prüfsumme (die Ziffern 3 und 4) muss korrekt sein. Bei einer Prüfung von 50.000 IBANs wird festgestellt, dass 120 ein falsches

Format haben und weitere 50 eine ungültige Prüfsumme. Somit sind $50000 - 170 = 49830$ IBANs valide, was einem Score von $S_{valid} = 49830/50000 = 0,9966$ entspricht.

3.3 Weitere relevante Dimensionen

Weitere Dimensionen wie Zugänglichkeit oder Interpretierbarkeit werden in erweiterten Frameworks wie ISO 8000 berücksichtigt.

Neben den Kern- und strukturellen Dimensionen existiert eine Reihe weiterer, oft subjektiverer oder prozessorientierter Dimensionen, die für das Vertrauen in und die Nutzbarkeit von Daten entscheidend sind.

3.3.1 Glaubwürdigkeit (Believability)

Im Zeitalter von Fake News hat Glaubwürdigkeit an Bedeutung gewonnen.

Glaubwürdigkeit ist eine subjektive Dimension, die das Vertrauen des Anwenders in die Daten widerspiegelt. Selbst objektiv korrekte Daten können nutzlos sein, wenn ihnen nicht geglaubt wird.

Die **Glaubwürdigkeit (Believability)** ist das Maß, in dem Daten von den Anwendern als wahr, real und vertrauenswürdig angesehen werden. Sie wird stark von der Reputation der Datenquelle und den bisherigen Erfahrungen des Nutzers beeinflusst.

Der „Halo-Effekt" kann die Glaubwürdigkeit beeinflussen: Wenn eine Datenquelle in einem Bereich als sehr zuverlässig gilt, neigen Nutzer dazu, auch ihren Daten in anderen Bereichen zu vertrauen, selbst wenn dort die Qualität schlechter ist.

Faktoren, die die Glaubwürdigkeit beeinflussen, sind unter anderem die wahrgenommene Vertrauenswürdigkeit der Datenquelle, das Vorhandensein von Metadaten, eine transparente Darstellung der Datenherkunft und die Abwesenheit offensichtlicher Fehler. Die Messung ist schwierig und erfolgt typischerweise durch Nutzerbefragungen, Umfragen oder Bewertungssysteme (z.,B. ein 5-Sterne-Rating für Datensätze).

In der Praxis wird Glaubwürdigkeit oft durch Peer-Reviews oder Zertifizierungen gesteigert.

3.3.2 Nachvollziehbarkeit (Traceability)

Die Nachvollziehbarkeit ist entscheidend für die Transparenz, Prüfbarkeit und Fehleranalyse von Daten.

Die **Nachvollziehbarkeit (Traceability)**, auch als **Data Lineage** bezeichnet, ist die Fähigkeit, den gesamten Lebensweg von Daten nachzuverfolgen – von ihrer ursprünglichen Quelle über alle Transformationen und Verarbeitungsschritte bis zu ihrer aktuellen Position und Verwendung.

Eine gute Nachvollziehbarkeit ermöglicht es, bei zweifelhaften Analyseergebnissen die Ursache schnell zu finden, die Auswirkungen von Datenänderungen abzuschätzen (Impact Analysis) und regulatorische Anforderungen zu erfüllen. Die Messung ist oft qualitativ und bewertet, ob eine Data Lineage dokumentiert, vollständig und zugänglich ist.

Git für Datenversionierung ermöglicht Nachvollziehbarkeit in ML-Projekten.

Beispiel: Regulatorische Anforderungen an Banken

Regulierungen wie der BCBS 239 (Basel Committee on Banking Supervision's Principles for effective risk data aggregation and risk reporting) im Finanzwesen oder die GMP-Richtlinien (Good Manufacturing Practice) in der Pharmaindustrie fordern explizit eine lückenlose Nachvollziehbarkeit von relevanten Daten, einschließlich end-to-end Data Lineage für Risikodatenaggregation und -berichterstattung in Banken, um Genauigkeit, Vollständigkeit und Transparenz zu gewährleisten.

Ein zentraler Baustein der Data Lineage ist der **ETL-Prozess** (Extract, Transform, Load). Dieser dreistufige Prozess bildet das Rückgrat der meisten Datenintegrationslösungen: In der Extract-Phase werden Daten aus verschiedenen Quellsystemen wie CRM- oder ERP-Systemen extrahiert. Die Transform-Phase bereinigt, standardisiert und konsolidiert diese Daten durch Filterung, Datentyp-Konvertierung, Aggregation und Geschäftsregeln. Schließlich werden die aufbereiteten Daten in der Load-Phase in das Zielsystem wie ein Data Warehouse geladen. Die Dokumentation dieser ETL-Schritte ist essentiell für die Nachvollziehbarkeit, da hier die meisten Datentransformationen stattfinden und potenzielle Fehlerquellen entstehen können.

ERP-System steht für „Enterprise Resource Planning" – das ist eine Software, die alle wichtigen Geschäftsprozesse eines Unternehmens in einem System zusammenfasst und verwaltet. Ein ERP-System integriert verschiedene Bereiche wie Finanzen und Buchhaltung, Personalwesen etc.

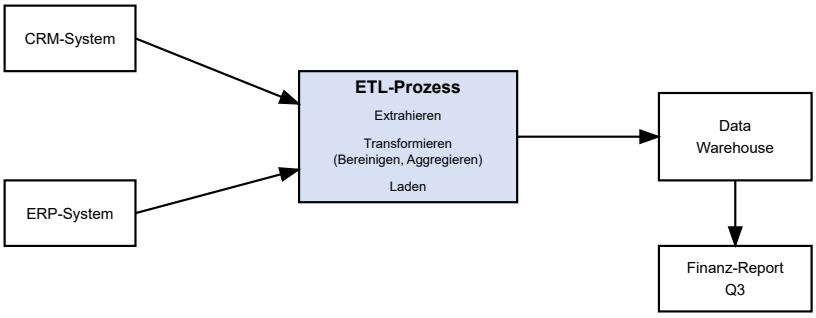


Abbildung 3.2: Vereinfachte Darstellung der Data Lineage für einen Finanz-Report, die den Datenfluss von den Quellsystemen bis zum Endprodukt nachvollziehbar macht.

3.4 Integration der Dimensionen in den Datenlebenszyklus

Der Datenlebenszyklus wird in modernen Frameworks wie CRISP-DM für Data Mining erweitert.

Datenqualität ist kein einmaliges Projekt, sondern ein kontinuierlicher Prozess, der in allen Phasen des Datenlebenszyklus verankert sein muss. Die Relevanz der einzelnen Dimensionen variiert entlang dieses Zyklus. Der Datenlebenszyklus umfasst typischerweise die Phasen der Datenerzeugung, -speicherung, -nutzung, -archivierung und -löschung.

- **Bei der Datenerzeugung:** Hier sind Validität, Vollständigkeit und Genauigkeit von größter Bedeutung. Eingabekontrollen, Pflichtfelder und Plausibilitätsprüfungen („Quality at the Source“) verhindern, dass fehlerhafte Daten überhaupt erst ins System gelangen.

- **Bei der Datenspeicherung und -integration:** Konsistenz und Eindeutigkeit rücken in den Fokus. Bei der Zusammenführung von Daten aus verschiedenen Quellen müssen Duplikate erkannt und logische Widersprüche aufgelöst werden.

- **Bei der Datennutzung:** Hier werden alle Dimensionen relevant. Insbesondere Aktualität und Glaubwürdigkeit beeinflussen die Akzeptanz und den Wert der Daten für den Anwender. Die Nachvollziehbarkeit wird entscheidend, um die Ergebnisse zu verstehen und zu validieren.

DataOps vereint DevOps-Prinzipien mit Datenmanagement für agile Qualitätskontrolle.

Moderne Ansätze wie DataOps integrieren automatisierte Qualitäts-Checks als festen Bestandteil von Datenpipelines. Jeder Verarbeitungsschritt wird durch Tests begleitet, die die Einhaltung der Qualitätsdimensionen sicherstellen. Dies schafft eine iterative Rückkopplungsschleife: Die Nutzung der Daten deckt Qualitätsprobleme auf, die dann in den vorgelagerten Prozessen systematisch behoben werden, um eine kontinuierliche Qualitätsverbesserung zu erreichen. Dieses Vorgehen wird im Kapitel **??** näher beleuchtet.

3.5 Zusammenfassung

Die systematische Bewertung von Datenqualität erfordert ein strukturiertes Vorgehen, das auf einem etablierten Set von Dimensionen basiert.

Dieses Kapitel hat die wichtigsten Dimensionen vorgestellt, die sich in Kern-, strukturelle und weitere relevante Kategorien einteilen lassen. Die Kern-Dimensionen – **Vollständigkeit**, **Genauigkeit**, **Konsistenz** und **Aktualität** – bilden das Fundament und adressieren die grundlegendsten Aspekte der Datenqualität. Die strukturellen Dimensionen – **Eindeutigkeit** und **Validität** – stellen sicher, dass Daten formal korrekt und ohne Redundanzen sind, was eine Voraussetzung für ihre maschinelle Verarbeitung ist. Darüber hinaus spielen subjektivere Dimensionen wie **Glaubwürdigkeit** und prozessorientierte wie die **Nachvollziehbarkeit** eine entscheidende Rolle für das Vertrauen in und die Transparenz von Daten.

[6] zeigen in einer empirischen Studie, dass unvollständige, fehlerhafte oder inkonsistente Trainings- und Testdaten die Leistung gängiger Machine-Learning-Algorithmen deutlich verschlechtern.

Es wurde deutlich, dass die Messung dieser Dimensionen von einfachen Zählungen (z. B. für Validität) bis hin zu komplexen, heuristischen Verfahren (z. B. für Eindeutigkeit) und subjektiven Bewertungen (z. B. für Glaubwürdigkeit) reicht. Die Auswahl, Gewichtung und Messung der relevanten Dimensionen muss stets im Kontext des spezifischen Anwendungsfalls („Fitness for Purpose“) erfolgen. Schließlich ist die Verankerung von Qualitätsprüfungen über den gesamten Datenlebenszyklus hinweg, von der Entstehung bis zur Nutzung, essenziell für ein nachhaltiges Datenqualitätsmanagement.

Data Governance als Fundament

4

Daten sind zu einem der wertvollsten Güter für Organisationen geworden. Doch ohne eine strukturierte Verwaltung und Steuerung bleibt ihr Potenzial ungenutzt oder, schlimmer noch, sie werden zur Quelle von Risiken und Fehlentscheidungen. Hier setzt die Data Governance an. Sie bildet das organisatorische, prozessuale und technologische Fundament, um sicherzustellen, dass Daten als strategisches Asset behandelt werden. Eine effektive Data Governance ist keine einmalige Initiative, sondern ein kontinuierliches Programm, das weit über die IT-Abteilung hinausgeht und das gesamte Unternehmen durchdringt. Sie schafft die notwendigen Rahmenbedingungen für ein erfolgreiches Datenqualitätsmanagement und ist die Voraussetzung für fortgeschrittene Analysen und den Einsatz von Künstlicher Intelligenz (KI).

In der digitalen Transformation spielen Daten eine zentrale Rolle, und Data Governance hilft, den Übergang von datenbasierten zu datengetriebenen Organisationen zu ermöglichen.

4.1 Ziele und Prinzipien der Data Governance

Die primären Ziele der Data Governance sind darauf ausgerichtet, den Wert von Daten für die Organisation zu maximieren und gleichzeitig die damit verbundenen Risiken zu minimieren. Dies wird durch die Umsetzung grundlegender Prinzipien erreicht, die Transparenz, Verantwortlichkeit und einen standardisierten Umgang mit Daten etablieren.

Data Governance ist die Ausübung von Autorität und Kontrolle (Planung, Überwachung und Durchsetzung) über die Verwaltung von Datenbeständen. Sie umfasst die Personen, Prozesse und Technologien, die erforderlich sind, um Daten als Unternehmensvermögen zu managen und zu schützen.

Die Ursprünge der Data Governance lassen sich bis in die 1990er Jahre zurückverfolgen, als Unternehmen mit der zunehmenden Datenflut in Data Warehouses begannen, die Notwendigkeit einer formalen Datenverwaltung zu erkennen. Regulatorische Anforderungen wie der Sarbanes-Oxley Act (2002) beschleunigten diese Entwicklung erheblich.

Ein zentrales Ziel ist die Sicherstellung der **Verfügbarkeit**, **Nutzbarkeit**, **Integrität** und **Sicherheit** der Daten im gesamten Unternehmen. Verfügbarkeit bedeutet, dass die richtigen Daten zur richtigen Zeit für die richtigen Personen zugänglich sind. Nutzbarkeit stellt sicher, dass die Daten verständlich, gut dokumentiert und für Analysezwecke geeignet sind. Integrität garantiert die Korrektheit und Konsistenz der Daten, während Sicherheit den Schutz vor unbefugtem Zugriff, Missbrauch und Verlust gewährleistet. Diese vier Säulen sind

untrennbar miteinander verbunden und bilden die Basis für das Vertrauen in die Daten.

Ein häufiges Missverständnis ist, Data Governance mit Datenmanagement gleichzusetzen. Data Governance ist die übergeordnete Steuerungsfunktion, die die Regeln festlegt, während Datenmanagement die operative Umsetzung dieser Regeln (z.B. Datenbankadministration, Backup-Prozesse) darstellt.

Ein weiteres fundamentales Prinzip ist die **Etablierung klarer Entscheidungsrechte und Verantwortlichkeiten**. Ohne definierte Zuständigkeiten kommt es zu unklaren Prozessen, widersprüchlichen Definitionen und einer diffusen Verantwortung für die Datenqualität. Data Governance legt fest, wer welche Entscheidungen über Daten treffen darf, wer für die Qualität bestimmter Datenbereiche zuständig ist und wer bei Problemen oder Konflikten konsultiert werden muss. Dies schafft eine Kultur der „Data Ownership“, in der Daten nicht als herrenloses Gut, sondern als wertvolle Ressource mit klaren Eigentümern behandelt werden.

Data Ownership fördert nicht nur Verantwortung, sondern auch Innovation, da Eigentümer motiviert sind, den Wert ihrer Daten zu maximieren.

Transparenz ist ein weiteres Leitprinzip. Es muss nachvollziehbar sein, woher Daten stammen (Data Lineage), wie sie transformiert wurden und welche Qualitätsstandards für sie gelten. Diese Transparenz ist entscheidend, um Vertrauen bei den Datennutzern aufzubauen und die Einhaltung interner sowie externer Vorschriften (Compliance) sicherzustellen.

4.2 Rollen und Verantwortlichkeiten

Rollen in Data Governance sollten regelmäßig überprüft und angepasst werden, um mit organisatorischen Veränderungen Schritt zu halten.

Eine . erfolgreiche Data Governance steht und fällt mit der klaren Definition und Besetzung spezifischer Rollen Diese Rollen stellen sicher, dass sowohl strategische als auch operative Aspekte der Datenverwaltung abgedeckt sind. Die bloße Existenz von Datenwerkzeugen genügt nicht; es sind die Menschen in ihren definierten Rollen, die die Governance mit Leben füllen.

4.2.1 Data Owner und Data Steward

Ein **Data Owner** ist typischerweise eine Führungskraft aus einem Fachbereich (z.B. Leiter Marketing, Leiter Finanzen), der die letztendliche Verantwortung für einen bestimmten Datenbereich (z.B. Kundendaten, Finanzdaten) trägt.

Die beiden zentralen Rollen in der operativen Umsetzung der Data Governance sind der Data Owner und der Data Steward. Ihre Abgrenzung ist entscheidend für die reibungslose Funktion des Systems. Der **Data Owner** ist eine strategische Rolle. Er oder sie ist rechenschaftspflichtig für einen bestimmten Datendomänen-Asset, wie beispielsweise „Kundendaten“oder „Produktdaten“. Der Data Owner trifft Entscheidungen über die Klassifizierung von Daten, genehmigt Zugriffsrechte und ist für die Einhaltung von Richtlinien und gesetzlichen Vorgaben verantwortlich. Er delegiert jedoch die tägliche, operative Verwaltung der Daten an den Data Steward. Die Hauptaufgaben umfassen:

- Festlegung der Datenqualitätsanforderungen aus Geschäftssicht.
- Genehmigung von Daten-Definitionen und Standards.
- Verantwortung für die Datensicherheit und den Datenschutz in seiner Domäne.
- Eskalationsinstanz bei Datenkonflikten.

Der **Data Steward** ist die operative Rolle. Er ist der anerkannte Experte für eine bestimmte Datendomäne und für deren tägliche Verwaltung und Pflege verantwortlich. Der Data Steward sorgt dafür, dass die vom Data Owner festgelegten Regeln umgesetzt werden, überwacht die Datenqualität, identifiziert und analysiert Datenprobleme und arbeitet an deren Lösung. Er ist der zentrale Ansprechpartner für Datennutzer bei Fragen zu „seinen"Daten. Seine Aufgaben beinhalten:

Ein **Data Steward** ist oft ein Fachexperte, der das Tagesgeschäft und die Bedeutung der Daten in seiner Domäne im Detail kennt. Er ist die erste Anlaufstelle für Fragen zur Datenqualität und -bedeutung.

- Definition und Pflege von Metadaten und Geschäftsdefinitionen.
- Durchführung von Datenqualitätsanalysen und Monitoring.
- Koordination von Datenbereinigungsaktivitäten.
- Dokumentation von Datenflüssen und Transformationsregeln.

Data Stewards nutzen oft Tools wie Data Catalogs, um Metadaten zentral zu verwalten und zugänglich zu machen.

Beispiel: Abgrenzung im Kundendatenmanagement

In einem Unternehmen ist der Leiter der Marketingabteilung der **Data Owner** für die Kundendaten. Er entscheidet, welche Kundensegmente für Kampagnen genutzt werden dürfen und genehmigt das Budget für ein neues Datenqualitätstool zur Adressvalidierung. Ein erfahrener Marketinganalyst wird zum **Data Steward** für Kundendaten ernannt. Er definiert im Business Glossary, was genau unter einem „aktiven Kunden"zu verstehen ist, überwacht täglich die Duplikatsrate in der Kundendatenbank (z.B. aus der Datei 'C:/Daten/daten.csv') und arbeitet mit der IT zusammen, um Regeln zur automatischen Bereinigung von Anreden (z.B. „Herr"vs. „Hr.") zu implementieren.

4.2.2 Das Data Governance Office (DGO)

Während Data Owner und Stewards dezentral in den Fachbereichen angesiedelt sind, bedarf es einer zentralen Instanz, die die Data-Governance-Aktivitäten unternehmensweit koordiniert, steuert und standardisiert. Diese Rolle übernimmt das **Data Governance Office (DGO)**, manchmal auch als

Die Berichtslinie des DGO ist entscheidend für seinen Erfolg. Idealerweise berichtet das DGO an eine hohe Managementebene, wie den Chief Data Officer (CDO) oder sogar den CEO, um die notwendige Autorität und unternehmensweite Akzeptanz zu gewährleisten.

Das DGO organisiert oft monatliche Meetings, um Best Practices auszutauschen und Konflikte frühzeitig zu lösen.

Data Governance Council bezeichnet. Das DGO agiert als zentrale Koordinations- und Steuerungsinstanz. Es ist nicht für die tägliche Datenpflege zuständig, sondern schafft die Rahmenbedingungen, Methoden und Standards, innerhalb derer die Data Stewards und Owner agieren. Zu den Kernaufgaben des DGO gehören die Entwicklung und Pflege des Data Governance Frameworks, die Bereitstellung von Schulungen und Richtlinien für die dezentralen Rollen sowie die Überwachung des Fortschritts der Governance-Initiative. Es fungiert als Schiedsstelle bei domänenübergreifenden Datenkonflikten und stellt sicher, dass die Governance-Ziele mit der übergeordneten Unternehmensstrategie im Einklang stehen. Das DGO treibt die Etablierung einer gemeinsamen Datensprache voran und fördert den Austausch von Best Practices zwischen den verschiedenen Datendomänen.

4.3 Datenqualitätsstrategie

Das Prinzip „Garbage In, Garbage Out"(GIGO) ist im Zeitalter der KI relevanter denn je. KI-Modelle, insbesondere solche des maschinellen Lernens, sind extrem empfindlich gegenüber der Qualität ihrer Trainingsdaten. Verzerrte oder fehlerhafte Daten führen unweigerlich zu verzerrten oder unzuverlässigen Modellen.

Eine wirksame Data Governance ist die Voraussetzung für eine erfolgreiche Datenqualitätsstrategie. Ohne die in der Governance definierten Rollen, Regeln und Prozesse bleiben Maßnahmen zur Verbesserung der Datenqualität oft punktuell, unkoordiniert und wenig nachhaltig. Die Verankerung von Datenqualität als strategisches Unternehmensziel ist ein Kernanliegen der Governance. Dies bedeutet, dass Datenqualität nicht länger als rein technisches Problem der IT-Abteilung betrachtet wird, sondern als geschäftskritischer Faktor, der direkt den Erfolg von Geschäftsprozessen, die Kundenzufriedenheit und die Qualität von strategischen Entscheidungen beeinflusst. Die Ausrichtung auf den Geschäftswert von Daten ist dabei entscheidend. Anstatt zu versuchen, „alle Daten zu 100 % zu bereinigen", fokussiert sich eine gute Strategie auf jene Daten und Qualitätsdimensionen, die den größten Einfluss auf kritische Geschäftsprozesse und -ziele haben.

Die Verbindung von Datenstrategie und KI-Roadmap ist essenziell. Viele KI-Initiativen scheitern nicht an den Algorithmen, sondern an der unzureichenden Verfügbarkeit und Qualität der benötigten Trainingsdaten.

Die Datenqualitätsstrategie muss eng mit der KI-Roadmap des Unternehmens verknüpft sein. Wenn das Ziel die Entwicklung eines KI-gestützten Empfehlungssystems ist, müssen die Datenqualitätsbemühungen auf die Vollständigkeit und Genauigkeit des Kundenkaufverhaltens und der Produktattribute konzentriert werden. Data Governance stellt sicher, dass diese Priorisierung stattfindet und die notwendigen Ressourcen bereitgestellt werden.

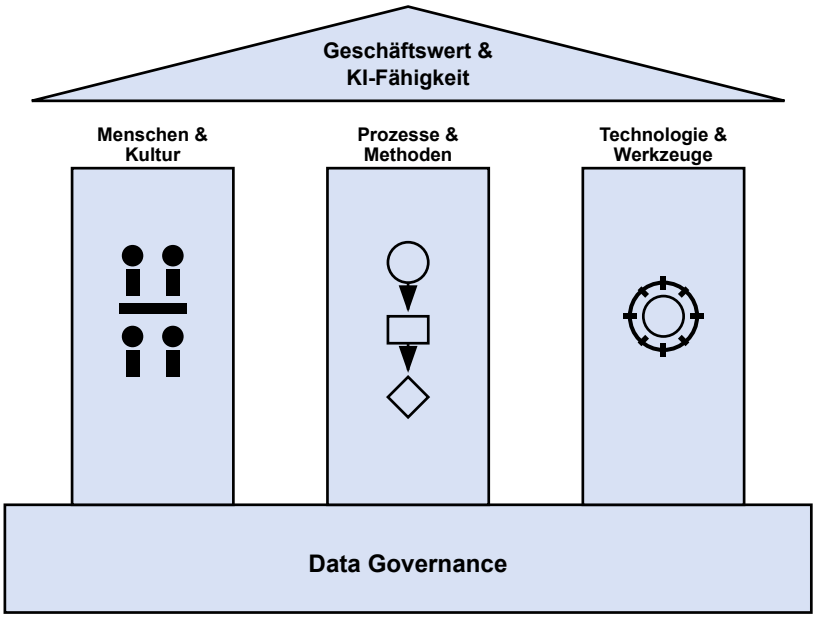


Abbildung 4.1: Die Säulen einer erfolgreichen Datenqualitätsstrategie, die auf dem Fundament der Data Governance ruht.

Ein weiterer Eckpfeiler ist die Etablierung von unternehmensweiten Standards und Richtlinien. Dies umfasst die Definition von Datenqualitätsmetriken, die Festlegung von Schwellenwerten für die Akzeptanz von Datenqualität und die Beschreibung von Prozessen zur Fehlerbehandlung. Schließlich fördert eine gelebte Data Governance eine datengetriebene Unternehmenskultur, in der jeder Mitarbeiter seine Verantwortung für die Datenqualität kennt und Qualität als gemeinsames Ziel versteht.

KI-Roadmaps sollten immer eine Datenbereitschafts-Bewertung enthalten, um Qualitätslücken früh zu identifizieren.

To Do Etablierung einer DQ-Kultur

Um eine datengetriebene Kultur zu fördern, sollten konkrete Schritte unternommen werden. Zunächst müssen die Data-Governance-Rollen klar kommuniziert und die Verantwortlichen im gesamten Unternehmen bekannt gemacht werden. Anschließend sollten regelmäßige, zielgruppenspezifische Schulungen zur Datenqualität für Mitarbeiter auf allen Ebenen angeboten werden. Ein wichtiger Schritt ist die Integration von Datenqualitäts-KPIs in die Zielvereinbarungen relevanter Mitarbeiter und Abteilungen. Schließlich sollte die Schaffung eines zentralen Anlaufpunkts, beispielsweise eines Wikis oder eines Bereichs im Intranet, für alle Richtlinien, Definitionen und Ansprechpartner sichergestellt werden.

Datenqualitätsmetriken können als KPIs in Dashboards visualisiert werden, um Transparenz zu schaffen. Zudem sollten Schulungen zur DQ-Kultur Gamification-Elemente enthalten, um das Engagement der Mitarbeiter zu steigern.

4.4 Frameworks und Reifegradmodelle

Reifegradmodelle wie CMMI können an Data Governance angepasst werden, um Fortschritte messbar zu machen.

Um Data Governance nicht von Grund auf neu erfinden zu müssen, können Organisationen auf etablierte Frameworks und Reifegradmodelle zurückgreifen. Diese bieten strukturierte Anleitungen, Best Practices und bewährte Vorgehensweisen für die Implementierung und Weiterentwicklung von Governance-Programmen. Reifegradmodelle helfen Organisationen, ihre aktuellen Fähigkeiten im Bereich Data Governance zu bewerten (Ist-Zustand), einen gewünschten Ziel-Zustand zu definieren und eine Roadmap für die schrittweise Verbesserung zu entwickeln.

4.4.1 Das DAMA-DMBOK Framework

DAMA International wurde 1980 gegründet und ist eine globale, gemeinnützige und anbieterunabhängige Vereinigung für Fachleute im Bereich Datenmanagement. Der DMBOK Guide wird oft als die „Bibel“des Datenmanagements bezeichnet.

Eines der bekanntesten und umfassendsten Frameworks für das Datenmanagement ist das DAMA-DMBOK (Data Management Body of Knowledge), herausgegeben von der DAMA International (Data Management Association). Das DMBOK-Framework strukturiert das Datenmanagement in verschiedene Wissensgebiete („Knowledge Areas“), die alle Aspekte des Umgangs mit Daten über deren gesamten Lebenszyklus abdecken. Data Governance wird dabei als zentrales, übergreifendes Wissensgebiet betrachtet, das alle anderen Bereiche steuert und koordiniert.

Das DMBOK umfasst über 700 Seiten und wird regelmäßig aktualisiert, um neue Technologien wie Big Data einzubeziehen. Die Wissensgebiete umfassen unter anderem Datenarchitektur, Datenmodellierung und -design, Datenspeicherung und -betrieb, Datensicherheit, Datenintegration, Stammdatenmanagement sowie Datenqualität. Das Framework beschreibt für jedes dieser Gebiete die Ziele, Prozesse, Rollen, Technologien und Metriken.

Durch die Anwendung des DMBOK-Frameworks können Unternehmen sicherstellen, dass sie alle relevanten Aspekte des Datenmanagements berücksichtigen und eine ganzheitliche Strategie entwickeln. Es bietet eine gemeinsame Sprache und ein gemeinsames Verständnis, was die Kommunikation zwischen Fachbereichen und IT erheblich erleichtert.

4.4.2 Industriestandards wie ISO 8000

Neben umfassenden Frameworks wie DMBOK gibt es auch internationale Standards, die sich auf spezifische Aspekte

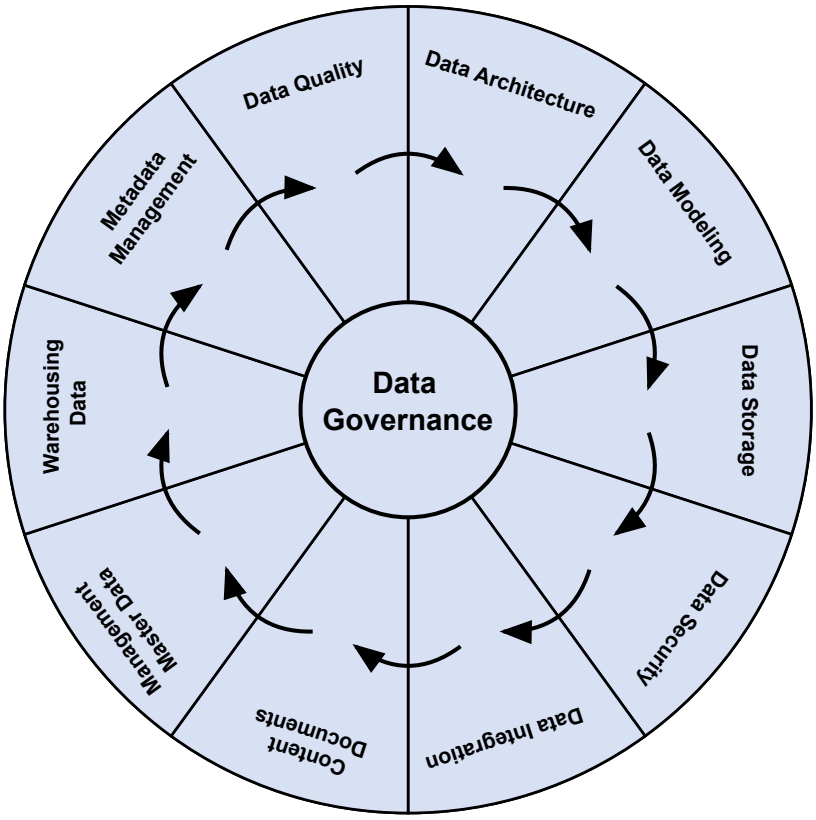


Abbildung 4.2: Das DAMA-DMBOK Framework, visualisiert als Rad, mit Data Governance als zentraler Nabe.

der Datenqualität konzentrieren. Die Normenreihe **ISO 8000** ist ein solcher Standard, der sich explizit mit Datenqualität befasst.

Die ISO 8000 besteht aus mehreren Teilen, die sich verschiedenen Facetten der Datenqualität widmen, von den grundlegenden Prinzipien (Teil 2) über das Datenqualitätsmanagement (Teil 6x) bis hin zum Austausch von Stammdaten (Teil 1xx). Ein zentraler Fokus der Norm liegt auf der Portabilität und Interoperabilität von Daten. Das Ziel ist es, sicherzustellen, dass Daten über System- und Unternehmensgrenzen hinweg ausgetauscht werden können, ohne an Qualität zu verlieren oder fehlinterpretiert zu werden.

Der Begriff **„charakteristische Daten"** in der ISO 8000 bezieht sich auf die Kombination aus einer Eigenschaft (z.B. „Farbe") und ihrem Wert (z.B. „rot"). Der Standard legt fest, wie solche Daten syntaktisch und semantisch eindeutig repräsentiert werden müssen, um maschinenlesbar und interpretierbar zu sein.

Die Norm legt besonderen Wert auf die explizite Dokumentation von Metadaten, die Spezifikation von Datenanforderungen und die Überprüfung der Konformität von Daten mit diesen Anforderungen. Für eine Organisation, die eine Data-Governance-Initiative startet, bietet die ISO 8000 einen konkreten, standardisierten Anforderungskatalog, der als Grundlage für interne Richtlinien und Qualitätsprüfungen dienen kann. Die Zertifizierung nach ISO 8000 kann zudem ein Wettbewerbsvorteil sein, da sie extern nachweist, dass das Unternehmen über robuste Prozesse zur Sicherung der Datenqualität verfügt.

ISO 8000 wird in der Fertigungsindustrie häufig verwendet, um Lieferketten zu standardisieren.

Zertifizierung nach ISO 8000 erfordert oft externe Audits, die die Einhaltung überprüfen.

4.5 Beispiele aus der Industrie

Die Notwendigkeit und Ausprägung von Data Governance variiert je nach Branche, getrieben durch regulatorische Anforderungen, Geschäftsmodelle und die Art der verarbeiteten Daten.

Der **BCBS 239** Standard wurde vom Basler Ausschuss für Bankenaufsicht nach der Finanzkrise 2008 veröffentlicht. Eines der Hauptprobleme in der Krise war, dass Banken nicht in der Lage waren, ihre Risikopositionen schnell und präzise zu aggregieren, was auf massive Mängel in der Dateninfrastruktur und -governance zurückzuführen war.

Im **Finanzwesen** ist Data Governance stark durch Compliance und regulatorische Vorgaben getrieben. Standards wie **BCBS 239** („Grundsätze für die effektive Aggregation von Risikodaten und die Risikoberichterstattung") fordern von global systemrelevanten Banken explizit die Einrichtung starker Data-Governance-Frameworks. Diese müssen die Genauigkeit, Integrität und Aktualität von Risikodaten sicherstellen. Auch der **EZB Guide on Effective Risk Data Aggregation and Risk Reporting** stellt ähnliche Anforderungen. Fehler in der Data Governance können hier zu massiven Strafen und einem Reputationsverlust führen. Im Finanzwesen können Governance-Fehler zu Millionenstrafen führen, wie bei mehreren Banken nach der Krise zu sehen war.

Beispiel: Umsetzung von BCBS 239

Eine Großbank implementiert BCBS 239, indem sie für alle kritischen Risikodaten (z.B. Kreditrisiko, Marktrisiko) klare Data Owner und Stewards benennt. Ein zentrales DGO wird etabliert, das die Einhaltung der Grundsätze überwacht. Es werden automatisierte Datenqualitätskontrollen eingeführt, die täglich die Vollständigkeit und Plausibilität der aus den Vorsystemen gelieferten Daten prüfen. Ein Data Lineage Tool wird implementiert, um den Datenfluss vom Quellsystem bis in den finalen Risikobericht lückenlos nachvollziehen zu können. Dies ermöglicht es der Bank, gegenüber den Aufsichtsbehörden die Herkunft und Qualität jeder einzelnen Kennzahl zu belegen.

Data Lineage in der Bankenbranche hilft nicht nur bei Compliance, sondern auch bei der Fehlersuche.

Im **Gesundheitswesen** stehen der Schutz sensibler Patientendaten (Datenschutz gemäß DSGVO) und die Patientensicherheit im Vordergrund. Eine präzise und konsistente Patientenhistorie ist lebenswichtig. Data Governance sorgt hier für die Einhaltung strenger Zugriffsregeln und stellt durch die Standardisierung von medizinischer Terminologie (z.B. SNOMED CT) sicher, dass Daten zwischen verschiedenen Krankenhäusern und Ärzten korrekt interpretiert werden können. In der **produzierenden Industrie** mit dem Aufkommen des Internets der Dinge (IoT) müssen riesige Mengen an Echtzeit-Sensordaten verarbeitet werden (**IoT-Daten**). Data Governance definiert hier, welche Daten für welche Analysen (z.B. Predictive Maintenance) relevant sind,

SNOMED CT ist ein internationales Vokabular mit über 300.000 Konzepten für medizinische Begriffe.

wie ihre Aktualität sichergestellt wird und wer die Verantwortung für die Qualität der von den Maschinen gelieferten Daten trägt. Unternehmen wie die **BMW Group** nutzen Data Governance, um Daten aus der gesamten Wertschöpfungskette – von der Produktion über die Logistik bis hin zum vernetzten Fahrzeug – zu integrieren. Dies ermöglicht komplexe Analysen zur Optimierung von Lieferketten und zur Entwicklung neuer datenbasierter Dienstleistungen.

Auch in der **Konsumgüterindustrie** ist Data Governance entscheidend. Unternehmen wie **Unilever** verwalten eine immense Vielfalt an Produktdaten, Lieferantendaten und Verbraucherdaten aus unterschiedlichen Quellen. Eine starke Governance ist hier notwendig, um eine „Single Source of Truth"für Produkte zu schaffen, die Einhaltung von Lebensmittelvorschriften weltweit zu gewährleisten und personalisierte Marketingkampagnen auf Basis verlässlicher Kundendaten durchzuführen. Eine Single Source of Truth minimiert Inkonsistenzen und verbessert die Entscheidungsfindung.

Policies sollten jährlich überprüft werden, um Aktualität zu gewährleisten.

Prompt Data Governance Policy

Erstelle das Grundgerüst für ein „Data Governance Policy"-Dokument in {Sprache}. Das Dokument soll für ein mittelständisches Produktionsunternehmen bestimmt sein. Es soll Abschnitte für die folgenden Themen enthalten: (1) Vision und Ziele der Data Governance, (2) Geltungsbereich, (3) Rollen und Verantwortlichkeiten (mit Kurzbeschreibung von Data Owner, Data Steward und DGO), (4) Grundprinzipien (z.B. Daten als Asset, Qualitätsverantwortung, Datensicherheit), (5) Prozess zur Handhabung von Datenqualitätsproblemen und (6) relevante Standards und Metriken.

4.6 Zusammenfassung

Dieses Kapitel hat die fundamentale Bedeutung der Data Governance als Grundpfeiler für ein erfolgreiches Datenqualitätsmanagement und die Nutzung von Daten als strategisches Unternehmensvermögen beleuchtet. Es wurde verdeutlicht, dass Data Governance kein rein technisches Thema ist, sondern eine disziplinübergreifende Aufgabe, die tief in der Organisation verankert sein muss.

Die zentralen Erkenntnisse sind:

- Data Governance verfolgt die übergeordneten Ziele, die Verfügbarkeit, Nutzbarkeit, Integrität und Sicherheit

von Daten zu gewährleisten und klare Entscheidungsrechte und Verantwortlichkeiten zu etablieren, wie in Abschnitt 4.1 auf Seite 29 dargelegt.

- Die Implementierung fester Rollen wie des strategischen **Data Owners** und des operativen **Data Stewards**, koordiniert durch ein zentrales **Data Governance Office (DGO)**, ist entscheidend für den Erfolg (siehe Abschnitt 4.2).

Erfolgreiche Data-Governance- und Datenqualitätsprogramme erfordern die aktive Unterstützung des Top-Managements. Ohne klare Verantwortlichkeiten, Priorisierung und Ressourcen auf Führungsebene bleiben viele Initiativen wirkungslos oder versanden im Tagesgeschäft.

- Eine durchdachte **Datenqualitätsstrategie** baut auf dem Fundament der Governance auf, richtet sich am Geschäftswert aus und ist insbesondere für den erfolgreichen Einsatz von KI-Technologien unerlässlich, wie in Abschnitt 4.3 ausgeführt.
- Etablierte **Frameworks** wie das **DAMA-DMBOK** und Standards wie die **ISO 8000** bieten wertvolle Orientierung und Best Practices für die systematische Einführung von Data Governance (Abschnitt 4.4).
- Die spezifische Ausgestaltung und die Treiber von Data Governance sind stark **branchenabhängig**, wie die Beispiele aus dem Finanzwesen, dem Gesundheitssektor und der Industrie in Abschnitt 4.5 zeigen.

Zusammenfassend lässt sich festhalten, dass Organisationen, die Data Governance implementieren, nicht nur Risiken minimieren und Compliance sicherstellen, sondern vor allem die Grundlage für Innovation, Effizienzsteigerung und eine nachhaltig datengetriebene Kultur schaffen.

5 Metadaten-Management

Metadaten-Management ist die systematische Verwaltung und Nutzung von "Daten über Daten". Es bildet das Rückgrat eines jeden erfolgreichen Data-Governance-Programms und ist eine unverzichtbare Disziplin, um den Wert von Datenbeständen in einem Unternehmen zu erschließen. Ohne ein effektives Metadaten-Management bleiben Daten oft unverständlich, unauffindbar und nicht vertrauenswürdig. Dieses Kapitel beleuchtet die grundlegenden Konzepte, Prozesse und Werkzeuge des Metadaten-Managements und zeigt dessen entscheidende Rolle für Datenqualität, Nachvollziehbarkeit und die Skalierung von KI-Anwendungen auf.

Metadaten haben ihre Wurzeln in der Bibliothekswissenschaft, wo sie bereits im 19. Jahrhundert zur Katalogisierung von Büchern verwendet wurden, lange bevor digitale Daten existierten.

5.1 Arten und Nutzen von Metadaten

Der Begriff Metadaten ist allgegenwärtig, wird jedoch oft unterschiedlich interpretiert. Um ein solides Fundament für das Management dieser wertvollen Ressource zu legen, ist eine klare Definition und Klassifizierung unerlässlich.

Metadaten sind strukturierte Informationen, die andere Daten beschreiben, erklären, lokalisieren oder auf andere Weise deren Verwaltung und Nutzung erleichtern. Sie liefern den notwendigen Kontext, um Rohdaten in aussagekräftige Informationen zu verwandeln.

In Social-Media-Plattformen wie Instagram dienen Metadaten wie Geotags und Hashtags als mächtiges Werkzeug zur Kategorisierung und Auffindbarkeit von Inhalten. Metadaten sind somit der Schlüssel zum Verständnis und zur Interpretation von Daten. Sie beantworten die fundamentalen Fragen: Was bedeuten diese Daten? Woher kommen sie? Wer ist für sie verantwortlich? Wie aktuell sind sie? Die Antworten auf diese Fragen steigern nicht nur das Verständnis, sondern auch das Vertrauen der Nutzer in die Daten, was eine Grundvoraussetzung für eine datengetriebene Kultur ist. Man unterscheidet üblicherweise drei Hauptkategorien von Metadaten, die jeweils unterschiedliche Zwecke erfüllen.

Der Begriff „Metadaten" wird oft Philip Bagley in seiner Arbeit „Extension of Programming Language Concepts" aus dem Jahr 1968 zugeschrieben. Er legte damit einen Grundstein für das moderne Datenmanagement.

Geschäftliche, technische und operationale Metadaten

Die Klassifizierung von Metadaten hilft dabei, Verantwortlichkeiten und Nutzungsszenarien klar zuzuordnen. Die drei wichtigsten Arten sind geschäftliche, technische und operationale Metadaten.

Laut der *Data Culture & Literacy Survey 2023* von Forrester ([7]) berichten Daten- und Analysemitarbeitende in Unternehmen mit hoher Datenkompetenz eine um bis zu 42 % höhere Produktivität.

Geschäftliche Metadaten (Business Metadata) beschreiben den fachlichen Kontext von Daten aus der Perspektive des Unternehmens. Sie sind entscheidend, um eine gemeinsame Sprache zwischen IT und Fachbereichen zu etablieren. Dazu gehören beispielsweise Definitionen von Geschäftsbegriffen aus einem „Business Glossary", die Beschreibung von Kennzahlen (KPIs), Informationen über den Data Owner oder Data Steward, Qualitätsregeln und die Klassifizierung von Daten hinsichtlich ihrer Vertraulichkeit (z. B. öffentlich, intern, geheim).

Technische Metadaten (Technical Metadata) beschreiben die Struktur und das Format der Daten aus einer IT-Perspektive. Sie sind für Entwickler, Datenbankadministratoren und Datenarchitekten von zentraler Bedeutung. Beispiele hierfür sind Datenbankschemata, Tabellen- und Spaltennamen, Datentypen (z. B. VARCHAR, INTEGER, TIMESTAMP), Längenbeschränkungen, Informationen über Primär- und Fremdschlüssel sowie Indexdefinitionen.

Operationale Metadaten (Operational Metadata) geben Auskunft über die Verarbeitung und Nutzung von Daten. Sie dokumentieren den Lebenszyklus der Daten und sind für die Überwachung von Datenprozessen und die Sicherstellung der Aktualität unerlässlich. Hierzu zählen Informationen über ETL-Prozesse (Extract, Transform, Load), Ausführungszeiten von Daten-Jobs, Ladefrequenzen, Zugriffsstatistiken und Informationen zur Datenherkunft (Data Lineage).

Beispiel: Metadaten einer Kundentabelle

Betrachten wir eine Tabelle namens `TBL_CUSTOMER` in einer Datenbank. Die zugehörigen Metadaten könnten wie folgt aussehen:

- **Geschäftlich:** Der Data Owner ist die Marketingabteilung. Die Tabelle enthält „alle aktiven Kunden", wobei ein aktiver Kunde als jemand definiert ist, der in den letzten 12 Monaten mindestens einen Kauf getätigt hat. Die Spalte 'Email' ist als personenbezogenes Datum (PII) klassifiziert.
- **Technisch:** Die Tabelle befindet sich in der Oracle-Datenbank `PROD_DB` im Schema `SALES`. Die Spalte 'CustomerID' ist vom Typ `NUMBER(10)` und der Primärschlüssel. Die Spalte 'RegistrationDate' ist vom Typ `DATE`.

- **Operational:** Die Tabelle wird jede Nacht um 02:00 Uhr durch den ETL-Job `J_Load_Customers` aus dem CRM-System aktualisiert. Der letzte erfolgreiche Lauf war heute um 02:15 Uhr und hat 150.000 Datensätze geladen.

In der Praxis sind die Grenzen zwischen den Metadaten-Arten oft fließend. Eine gut definierte Geschäftsregel (geschäftlich) kann beispielsweise direkt in einer SQL-Validierungsroutine (technisch) implementiert werden, deren Ausführungsprotokoll (operational) wiederum Metadaten erzeugt.

5.2 Data Lineage: Datenflüsse nachvollziehen

Data Lineage, oder Datenherkunftsnachverfolgung, ist eine der kritischsten Funktionen des Metadaten-Managements. Sie visualisiert den gesamten Lebenszyklus von Daten, von ihrer Quelle bis zu ihrer Verwendung in Berichten oder Analysen.

Data Lineage dokumentiert den Datenfluss und die Transformationen, die Daten auf ihrem Weg durch verschiedene Systeme und Prozesse durchlaufen. Sie zeigt auf, woher Daten stammen (Herkunft), was mit ihnen auf dem Weg geschieht (Transformationen) und wo sie verwendet werden (Ziel).

Die Visualisierung erfolgt typischerweise in Form eines gerichteten Graphen, in dem Knoten die Daten-Assets (z. B. Tabellen, Dateien, Berichte) und Kanten die Prozesse (z. B. ETL-Jobs, SQL-Abfragen) darstellen, die die Daten bewegen und verändern. Diese Nachvollziehbarkeit ist aus mehreren Gründen von entscheidender Bedeutung. Sie ist fundamental für die **Fehleranalyse** (Root Cause Analysis). Wenn in einem Finanzbericht ein falscher Wert entdeckt wird, ermöglicht die Data Lineage eine schnelle Rückverfolgung zur fehlerhaften Quelle oder Transformation, was die Zeit zur Fehlerbehebung drastisch reduziert. Ebenso wichtig ist die **Auswirkungsanalyse** (Impact Analysis). Vor einer Änderung an einem Datenfeld oder einer Transformationslogik kann analysiert werden, welche nachgelagerten Systeme, Berichte und Modelle davon betroffen sind. Dies verhindert ungeplante Ausfälle und sichert die Stabilität der Dateninfrastruktur.

Data Lineage ist essenziell für Explainable AI, da es die Herkunft von Trainingsdaten offenlegt und Bias-Quellen identifizieren hilft.

Darüber hinaus sind Datenqualitätsmetriken selbst eine wichtige Form von Metadaten. Die Speicherung von Qualitätskennzahlen wie Vollständigkeit, Aktualität oder Korrektheit

auf Spalten- oder Tabellenebene als Metadaten ermöglicht es, die Vertrauenswürdigkeit von Daten-Assets direkt zu bewerten. Ein Analyst kann so auf einen Blick sehen, dass beispielsweise die Spalte `Postleitzahl` eine Vollständigkeit von 95 % und eine Plausibilität von 99 % aufweist.

In Big-Data-Umgebungen wie Hadoop können Metadaten die Verarbeitung von Petabytes an Daten effizient steuern, indem sie den Zugriff auf verteilte Dateisysteme optimieren.

Letztendlich fungieren Metadaten und insbesondere Data Lineage als entscheidender „Enabler" für Datenverständnis, Wiederverwendbarkeit und Compliance. Sie ermöglichen es den Nutzern, Daten eigenständig zu finden, zu verstehen und ihnen zu vertrauen, was die Effizienz und Agilität im gesamten Unternehmen steigert und für regulatorische Anforderungen wie die DSGVO oder den AI Act oft zwingend erforderlich ist.

Die Komplexität eines Lineage-Graphen kann symbolisch ausgedrückt werden. Sei $G = (V, E)$ ein Lineage-Graph, wobei V die Daten-Assets und E die Prozesse sind. Die Komplexität kann als Funktion der Anzahl der Knoten $|V|$, Kanten $|E|$ und der Tiefe des Graphen betrachtet werden.

5.3 Governance und organisatorische Aspekte

Ein erfolgreiches Metadaten-Management ist keine rein technische Aufgabe, sondern erfordert eine solide Verankerung in der Organisation durch klare Data-Governance-Strukturen. Ohne definierte Rollen, Verantwortlichkeiten und Prozesse bleiben Metadaten oft unvollständig, veraltet und inkonsistent. Die zentrale Frage lautet: Wer pflegt und nutzt Metadaten? Die Verantwortung für die Metadatenpflege muss klar zugewiesen werden. Typischerweise liegt die Verantwortung für geschäftliche Metadaten bei den **Data Stewards** aus den Fachbereichen. Sie definieren, was Daten bedeuten und welche Qualitätsanforderungen gelten. Technische und operationale Metadaten werden hingegen oft von IT-Teams, z. B. Datenbankadministratoren oder ETL-Entwicklern, gepflegt, wobei eine zunehmende Automatisierung der Erfassung angestrebt wird. Die Etablierung dieser Verantwortlichkeiten muss in einem übergreifenden **Data Governance Framework** erfolgen, wie es in Kapitel 4 beschrieben wird. Dieses Framework legt die Spielregeln für den Umgang mit Daten und Metadaten fest. Ein wesentlicher Bestandteil davon sind **Metadatenrichtlinien**. Diese Richtlinien definieren, welche Metadaten für welche Daten-Assets verpflichtend zu dokumentieren sind, in welcher Qualität dies geschehen muss

Die Einbindung in KI-Projekte ist entscheidend. Metadaten über Trainingsdatensätze, wie Herkunft, Bias-Metriken und Feature-Definitionen, sind für die Entwicklung robuster und erklärbarer KI-Modelle unabdingbar und werden durch den AI Act zunehmend zur Pflicht.

(z. B. muss jede Tabelle einen dokumentierten Eigentümer haben) und mit welcher Frequenz die Metadaten überprüft und aktualisiert werden müssen.

To Do Erstellung einer Metadatenrichtlinie

1. **Stakeholder identifizieren:** Stellen Sie ein Gremium aus Vertretern der Fachbereiche, der IT und der Data Governance zusammen.
2. **Kritische Datenobjekte definieren:** Identifizieren Sie die wichtigsten Daten-Assets (z. B. Kundenstamm, Produktdaten, Finanzkennzahlen), für die eine Metadatendokumentation priorisiert werden soll.
3. **Minimale Metadatenfelder festlegen:** Definieren Sie einen minimalen Satz an Metadaten, der für jedes kritische Datenobjekt erfasst werden muss. Dies könnte umfassen: Geschäftliche Definition, Data Owner, Datenklassifizierung und Aktualisierungsfrequenz.
4. **Verantwortlichkeiten zuweisen:** Weisen Sie klare Rollen (z. B. Data Stewards) für die Erfassung und Pflege dieser Metadaten zu.
5. **Prozess zur Qualitätssicherung etablieren:** Definieren Sie einen Prozess, wie die Qualität und Vollständigkeit der erfassten Metadaten regelmäßig überprüft wird.

Metadaten-Silos entstehen oft durch dezentrale Systeme. Eine zentrale Governance kann diese durch Standardisierung auflösen. Für GDPR-Compliance sind zudem Metadaten über personenbezogene Daten (PII) entscheidend, da sie die Nachverfolgung von Datenschutzverletzungen ermöglichen.

5.4 Werkzeuge: Data Catalogs und Business Glossaries

Um Metadaten effektiv zu verwalten und für eine breite Nutzerschaft zugänglich zu machen, sind spezialisierte Werkzeuge unerlässlich. Die beiden wichtigsten Kategorien sind Data Catalogs und Business Glossaries, die oft integriert sind, aber unterschiedliche Schwerpunkte haben.

Ein **Data Catalog** fungiert als zentrales Inventar aller Datenbestände eines Unternehmens. Man kann ihn sich wie eine Suchmaschine oder eine Bibliothek für Daten vorstellen. Er sammelt (oft automatisiert) technische und operationale Metadaten aus verschiedenen Quellsystemen (Datenbanken, Data Lakes, BI-Tools) und bereitet sie in einer benutzerfreundlichen Oberfläche auf. Nutzer können im Katalog nach Daten

Bekannte Tools wie Collibra oder Alation kombinieren Data Catalogs mit Glossaries und bieten KI-gestützte Features für automatisierte Metadaten-Erfassung.

suchen, deren Herkunft (Lineage) einsehen, Qualitätsmetriken prüfen und sehen, wer die Daten nutzt und wer für sie verantwortlich ist. Moderne Data Catalogs nutzen KI, um automatisch Daten zu klassifizieren, PII zu erkennen oder Verknüpfungen zu Geschäftsbegriffen vorzuschlagen.

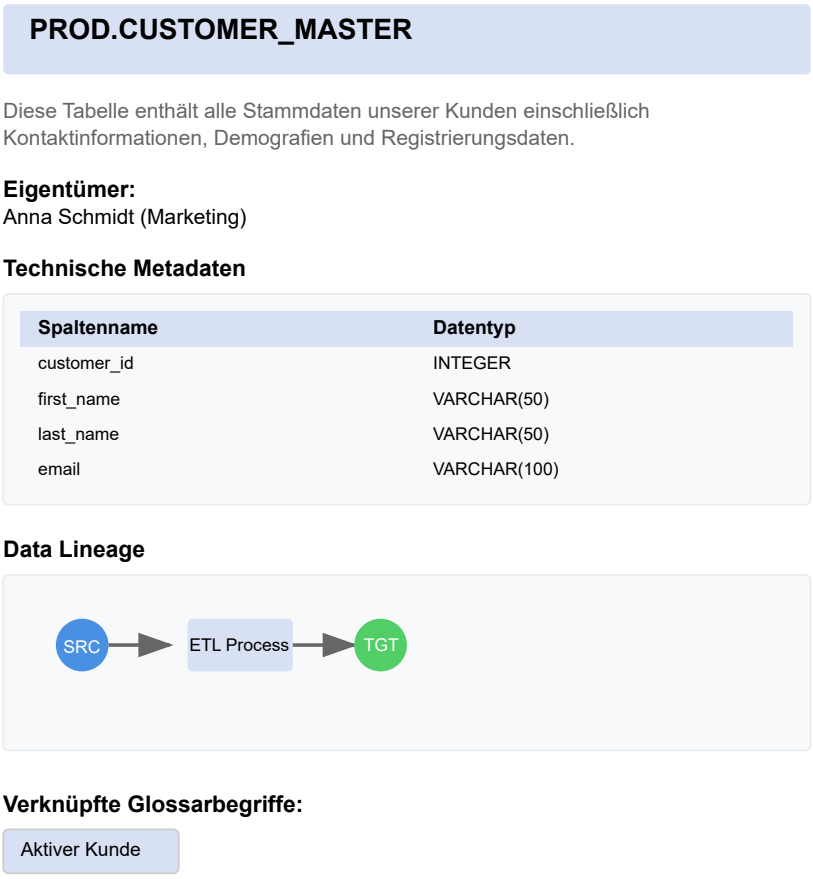


Abbildung 5.1: Konzeptioneller Aufbau einer Benutzeroberfläche für einen Data Catalog. Die Plattform integriert Suche, Filterung und eine detaillierte Ansicht von Metadaten, um die Auffindbarkeit und das Verständnis von Daten zu maximieren.

Ein **Business Glossary** ist hingegen ein zentrales Wörterbuch, das die Geschäftsdefinitionen und -regeln eines Unternehmens standardisiert. Es ist die „Single Source of Truth“ für die Fachterminologie. Während der Data Catalog die Frage „Wo finde ich die Kundentabelle?“ beantwortet, beantwortet das Business Glossary die Frage „Was verstehen wir unter einem ’aktiven Kunden’?“. Es schafft ein gemeinsames Vokabular und verhindert Missverständnisse, die entstehen, wenn verschiedene Abteilungen denselben Begriff unterschiedlich interpretieren. In einer integrierten Lösung sind die Begriffe aus dem Glossar direkt mit den entsprechenden technischen Assets im Katalog verknüpft.

Die Einführung eines Data Catalogs sollte als agiles Projekt verstanden werden. Anstatt zu versuchen, von Anfang an alle Datenquellen zu integrieren, ist es oft erfolgreicher, mit einem oder zwei hoch-prioritären Anwendungsfällen oder Fachdomänen zu starten und den Umfang iterativ zu

erweitern. Eine IDC-Studie schätzt, dass bis 2025 über 80 % der Unternehmen Data Catalogs einsetzen werden, um ihre Datenassets zu managen.

5.5 Zusammenfassung

Dieses Kapitel hat die fundamentale Bedeutung von Metadaten als eine zentrale Säule der Data Governance und als Voraussetzung für hohe Datenqualität und vertrauenswürdige Analytik beleuchtet. Metadaten sind weit mehr als nur technische Dokumentation; sie sind ein strategisches Gut, das den Kontext liefert, um Daten in wertvolle Informationen zu überführen.

Die zentralen Erkenntnisse lassen sich wie folgt zusammenfassen: Metadaten werden in geschäftliche, technische und operationale Kategorien unterteilt, die zusammen ein 360-Grad-Bild eines Daten-Assets zeichnen.

Data Lineage wurde als entscheidendes Instrument identifiziert, das die Nachvollziehbarkeit von Datenflüssen ermöglicht und damit die Fehler- und Auswirkungsanalyse maßgeblich unterstützt.

Ein effektives Metadaten-Management kann nicht ohne eine klare **Governance** existieren, die Rollen wie den Data Steward verankert und durch Richtlinien klare Vorgaben schafft.

Schließlich wurden mit dem **Data Catalog** und dem **Business Glossary** die zentralen Werkzeuge vorgestellt, welche die Erfassung, Verwaltung und Nutzung von Metadaten in der Praxis ermöglichen und so eine datengetriebene Kultur fördern.

Die Implementierung eines umfassenden Metadaten-Managements ist eine Investition, die das Vertrauen in Daten stärkt, die Effizienz steigert und die Grundlage für fortgeschrittene Analysen und KI-Anwendungen schafft.

Auswirkungen der Datenarchitektur

6

Die Architektur, auf der Daten gespeichert, verarbeitet und bereitgestellt werden, ist das Fundament für alle datengetriebenen Aktivitäten in einem Unternehmen. Sie bestimmt nicht nur die Leistungsfähigkeit und Skalierbarkeit von Analysesystemen, sondern prägt maßgeblich die erreichbare Datenqualität. Entscheidungen über Datenbanktechnologien, Plattformmodelle und Integrationsmuster haben direkte und oft langanhaltende Konsequenzen für Dimensionen wie Konsistenz, Validität und Aktualität. Eine unpassende Architektur kann die Implementierung von Datenqualitätsregeln erschw itching die erreichbare Datenqualität.

Eine unpassende Architektur kann die Implementierung von Datenqualitätsregeln erschweren oder sogar unmöglich machen, während eine durchdachte Architektur Datenqualität proaktiv fördert und als integralen Bestandteil des Systems verankert. Dieses Kapitel beleuchtet die kritischen Zusammenhänge zwischen Architekturentscheidungen und Datenqualität, beginnend bei der fundamentalen Wahl zwischen relationalen und NoSQL-Datenbanken bis hin zu modernen Paradigmen wie Data Mesh und der Nutzung von Graphdatenbanken. Data Mesh als Konzept gewann 2020 an Popularität durch Zhamak Dehghanis Artikel auf martinFowler.com.

Die Entwicklung relationaler Datenbanken begann in den 1960er Jahren, aber erst in den 1980er Jahren wurden kommerzielle Systeme wie IBM DB2 und Oracle populär.

6.1 Relationale vs. NoSQL-Datenbanken

Die Wahl des Datenbanktyps ist eine der grundlegendsten Weichenstellungen in der Datenarchitektur. Die beiden dominierenden Paradigmen, relationale und NoSQL-Systeme, verfolgen fundamental unterschiedliche Ansätze hinsichtlich der Datenstrukturierung und Konsistenzsicherung, was direkte Auswirkungen auf die Datenqualität hat.

6.1.1 Relationale Datenbanken (RDBMS)

Relationale Datenbankmanagementsysteme (RDBMS) sind seit den 1970er Jahren der etablierte Standard für die Speicherung strukturierter Daten. Ihr Fundament ist das relationale

Modell, das von Edgar F. Codd entwickelt wurde und Daten in Tabellen organisiert, die aus Zeilen (Tupel) und Spalten (Attribute) bestehen. Jede Tabelle besitzt ein festes Schema, das die Datentypen, Beziehungen und Integritätsbedingungen vordefiniert.

Edgar F. Codd, ein britischer Informatiker bei IBM, veröffentlichte 1970 sein bahnbrechendes Papier „A Relational Model of Data for Large Shared Data Banks". Damit legte er den theoretischen Grundstein für praktisch alle modernen relationalen Datenbanksysteme.

Das entscheidende Merkmal von RDBMS im Hinblick auf die Datenqualität ist das **Schema-on-Write-Prinzip**. Daten müssen den im Schema definierten Regeln entsprechen, *bevor* sie in die Datenbank geschrieben werden dürfen. Dies erzwingt eine proaktive Qualitätssicherung. Beispielsweise wird ein Eintrag abgewiesen, wenn versucht wird, eine Zeichenkette in ein numerisches Feld zu schreiben oder einen Datensatz ohne einen gültigen Fremdschlüssel zu einer anderen Tabelle einzufügen. Diese inhärenten Validierungsmechanismen, wie Primärschlüssel, Fremdschlüssel, 'NOT NULL'-Constraints und 'CHECK'-Constraints, sind mächtige Werkzeuge zur Sicherung der referenziellen Integrität und der Validität der Daten direkt auf der Speicherebene.

Relationale Datenbanken wie MySQL werden in über 70% der Unternehmen für transaktionale Systeme eingesetzt, laut einer Umfrage von Stack Overflow 2023.

Ein weiteres Kernkonzept von RDBMS ist die Gewährleistung von **ACID**-konformen Transaktionen, die für die Verlässlichkeit von geschäftskritischen Prozessen unerlässlich sind.

Die **ACID**-Eigenschaften beschreiben ein Transaktionsparadigma, das die Datenintegrität in Datenbanksystemen sicherstellt:

- **Atomarität (Atomicity):** Eine Transaktion wird entweder vollständig oder gar nicht ausgeführt. Schlägt ein Teil fehl, wird die gesamte Operation zurückgerollt, wodurch keine inkonsistenten Zwischenzustände in der Datenbank verbleiben.
- **Konsistenz (Consistency):** Eine Transaktion überführt die Datenbank von einem gültigen Zustand in einen anderen. Alle definierten Integritätsregeln (z.B. Constraints) müssen nach der Transaktion erfüllt sein.
- **Isolation (Isolation):** Parallel ausgeführte Transaktionen beeinflussen sich nicht gegenseitig. Jede Transaktion läuft so ab, als ob sie die einzige im System wäre.
- **Dauerhaftigkeit (Durability):** Die Ergebnisse einer erfolgreich abgeschlossenen Transaktion bleiben dauerhaft erhalten, selbst bei einem Systemausfall.

Die Stärke von SQL (Structured Query Language) liegt in seiner deklarativen Natur. Der Benutzer beschreibt, *welche* Daten er benötigt, nicht *wie* er sie abrufen möchte. Das RDBMS-Optimierungsmodul kümmert sich um den effizienten Ausführungsplan.

Die Stärken von RDBMS liegen somit in der hohen Datenintegrität durch starre Schemata und ACID-Garantien, der Standardisierung durch die Abfragesprache SQL und der technologischen Reife. Diese Systeme sind die bevorzugte Wahl für Anwendungsfälle, bei denen Konsistenz und Verlässlichkeit oberste Priorität haben, wie beispielsweise in ERP-Systemen, Buchhaltungssoftware oder Kernbanken-

systemen. Die Herausforderungen liegen jedoch in der horizontalen Skalierbarkeit – das Verteilen einer relationalen Datenbank auf viele Maschinen ist komplex – und in der geringen Flexibilität. Änderungen am Schema, wie das Hinzufügen einer Spalte, können, abhängig vom System und der Tabellengröße, aufwendige Migrationen erfordern.

6.1.2 NoSQL-Datenbanken

PostgreSQL, ein Open-Source RDBMS, unterstützt Erweiterungen wie PostGIS für geospatiale Daten, was seine Vielseitigkeit erhöht.

Der Begriff NoSQL (oft als „Not only SQL“interpretiert) entstand als Sammelbezeichnung für eine Vielzahl von Datenbanktechnologien, die von dem starren Tabellenmodell der RDBMS abweichen. Sie wurden mit dem Aufkommen von Big Data und hochskalierbaren Webanwendungen populär und sind für Flexibilität, Leistung und horizontale Skalierbarkeit optimiert. Anstelle eines einheitlichen Schemas unterstützen sie vielfältige Datenmodelle, darunter Dokumentenspeicher (z.B. MongoDB), Key-Value-Speicher (z.B. Redis), Spaltenfamilien-Datenbanken (z.B. Cassandra) und Graphdatenbanken (z.B. Neo4j). Das zentrale Prinzip dieser Datenbanken ist oft **Schema-on-Read**. Daten können in unterschiedlichen, auch unvollständigen Strukturen gespeichert werden. Die Verantwortung für die Interpretation und Validierung der Daten liegt bei der Anwendung, die die Daten liest. Dies ermöglicht eine enorme Flexibilität, da neue Datenformate ohne vorherige Schema-Anpassung aufgenommen werden können. Gleichzeitig verlagert es aber die Last der Qualitätssicherung von der Datenbank in den Anwendungscode. Ohne disziplinierte Governance kann dies zu einem Sammelsurium inkonsistenter Datenformate führen, was die spätere Analyse erheblich erschwert.

Cassandra wurde von Facebook entwickelt, um skalierbare Speicherung für Milliarden von Nutzernachrichten zu ermöglichen.

Anstelle von ACID-Garantien folgen viele verteilte NoSQL-Systeme dem **BASE**-Prinzip, das für hochverfügbare und ausfalltolerante Systeme besser geeignet ist.

MongoDB, ein populäres NoSQL-System, speichert Daten als BSON-Dokumente, die JSON-ähnlich sind, aber binär für Effizienz.

Das **BASE**-Prinzip beschreibt ein schwächeres Konsistenzmodell, das in vielen verteilten Systemen priorisiert wird:

- **Basically Available (Grundsätzlich verfügbar):**

  Das System garantiert die Verfügbarkeit, auch wenn Teile davon ausfallen. Anfragen werden stets beantwortet, eventuell mit veralteten Daten oder einem Fehler.

- **Soft State (Weicher Zustand):** Der Zustand des Systems kann sich auch ohne neue Eingaben ändern, da er durch die eventuelle Konsistenz beeinflusst wird.

- **Eventual Consistency (Eventuelle Konsistenz):** Wenn keine neuen Aktualisierungen mehr erfolgen, wird das System nach einer gewissen Zeit in einen konsistenten Zustand übergehen. Alle Replikate der Daten werden irgendwann den gleichen Wert aufweisen.

Der Begriff „NoSQL“wurde 1998 von Carlo Strozzi für seine leichtgewichtige, relationale Open-Source-Datenbank ohne SQL-Schnittstelle geprägt. 2009 wurde der Begriff von Johan Oskarsson von Last.fm für ein Event über verteilte, nicht-relationale Datenbanken wiederbelebt und populär gemacht.

Das CAP-Theorem von Eric Brewer besagt, dass ein verteiltes System nicht gleichzeitig Konsistenz, Verfügbarkeit und Partitionstoleranz garantieren kann; NoSQL wählt oft Verfügbarkeit über Konsistenz.

Die Vorteile von NoSQL-Datenbanken liegen in ihrer hervorragenden horizontalen Skalierbarkeit und ihrer Fähigkeit, große Mengen polystrukturierter Daten (Daten mit variierenden Schemata) effizient zu verarbeiten. Dies macht sie ideal für Big-Data-Anwendungen, soziale Netzwerke, IoT-Plattformen und Content-Management-Systeme.

6.1.3 Vergleichskriterien und Anwendungsfälle

Die Entscheidung zwischen RDBMS und NoSQL hängt von den spezifischen Anforderungen des Anwendungsfalls ab. Die Abwägung betrifft primär das Datenmodell, das Konsistenzmodell, die Abfragekomplexität und die Skalierungsanforderungen. RDBMS erzwingen ein strukturiertes Datenmodell (Tabellen) und bieten starke Konsistenz (ACID), was sie für transaktionale Systeme prädestiniert. Die Abfragesprache SQL ist standardisiert und sehr mächtig für komplexe, relationale Abfragen und Aggregationen. Die Skalierung erfolgt traditionell vertikal („scaling up“– mehr Leistung auf einer Maschine).

NoSQL-Systeme hingegen bieten flexible Datenmodelle (Dokumente, Graphen etc.) und favorisieren Verfügbarkeit über strikte Konsistenz (BASE). Die Abfragesprachen sind vielfältig und oft auf das jeweilige Datenmodell optimiert. Ihre Stärke liegt in der horizontalen Skalierbarkeit („scaling out“– Verteilung der Last auf viele Maschinen).

Beispiel: Wahl des Datenbanksystems

Ein **E-Commerce-Unternehmen** benötigt ein System zur Verwaltung von Kundenbestellungen. Die Kerndaten – Kunden, Aufträge, Artikel, Zahlungen – sind stark strukturiert und erfordern höchste Konsistenz. Eine nicht ausgeführte Zahlung oder ein falsch berechneter Lagerbestand kann direkte finanzielle Auswirkungen haben. Hier ist ein **RDBMS** (z.B. PostgreSQL, Oracle) aufgrund sei-

ner ACID-Garantien und starken Integritätsprüfungen die richtige Wahl für das Kern-Transaktionssystem.

Dasselbe Unternehmen möchte jedoch auch das Klickverhalten der Benutzer auf der Webseite analysieren, um Produktempfehlungen zu verbessern. Diese Klickstrom-Daten fallen in riesigen Mengen an, sind semi-strukturiert (jeder Klick kann unterschiedliche Metadaten enthalten) und erfordern keine sofortige Konsistenz. Hier eignet sich ein **NoSQL-System**, z.B. ein Dokumentenspeicher oder ein Spaltenfamilienspeicher, um die Daten schnell aufzunehmen und skalierbar zu verarbeiten. Die Datenqualität wird hier in nachgelagerten Prozessen sichergestellt.

In hybriden Systemen werden RDBMS für Transaktionen und NoSQL für Analysen kombiniert, um das Beste aus beiden Welten zu nutzen.

6.1.4 Schema-on-Write vs. Schema-on-Read

Der konzeptionelle Unterschied zwischen Schema-on-Write und Schema-on-Read ist für die Datenqualitätsstrategie von zentraler Bedeutung. Er definiert, an welchem Punkt im Datenlebenszyklus Qualitätsregeln durchgesetzt werden.

In der Praxis verschwimmen die Grenzen. Moderne RDBMS können JSON-Dokumente speichern und NoSQL-Datenbanken führen zunehmend optional einsetzbare Schema-Validierungsfunktionen ein, um ein Mindestmaß an Struktur zu erzwingen.

Schema-on-Write, der Ansatz von RDBMS, ist eine Form der **proaktiven Qualitätssicherung**. Die Daten werden bei ihrer Entstehung an der Quelle validiert. Dies verhindert von vornherein, dass syntaktisch falsche oder inkonsistente Daten in das System gelangen. Der Vorteil ist eine hohe, garantierte Grundqualität der Daten im System. Der Nachteil ist die Rigidität: Das Schema muss im Voraus bekannt sein und Änderungen sind aufwendig. Dies kann die Aufnahme neuer, unvorhergesehener Datenformate verlangsamen.

Schema-on-Read ermöglicht agile Entwicklung, da Änderungen am Datenmodell die Speicherung nicht beeinträchtigen.

Schema-on-Read , typisch für NoSQL-Datenbanken und Data Lakes, ist dagegen eine Form der **reaktiven Qualitätssicherung**. Daten werden zunächst in ihrer Rohform gespeichert, was eine schnelle und flexible Datenerfassung ermöglicht. Die Strukturierung, Bereinigung und Validierung erfolgen erst zum Zeitpunkt der Abfrage oder Analyse. Diese Flexibilität ist ein enormer Vorteil bei der explorativen Analyse und bei der Arbeit mit heterogenen Datenquellen. Das Risiko besteht jedoch darin, dass ohne eine strikte Governance im Analyseprozess unterschiedliche Nutzer die gleichen Rohdaten unterschiedlich interpretieren und bereinigen, was zu inkonsistenten Ergebnissen führt. Die Datenqualität wird zu einer Verantwortung der Anwendung und des Nutzers, nicht des Speichersystems.

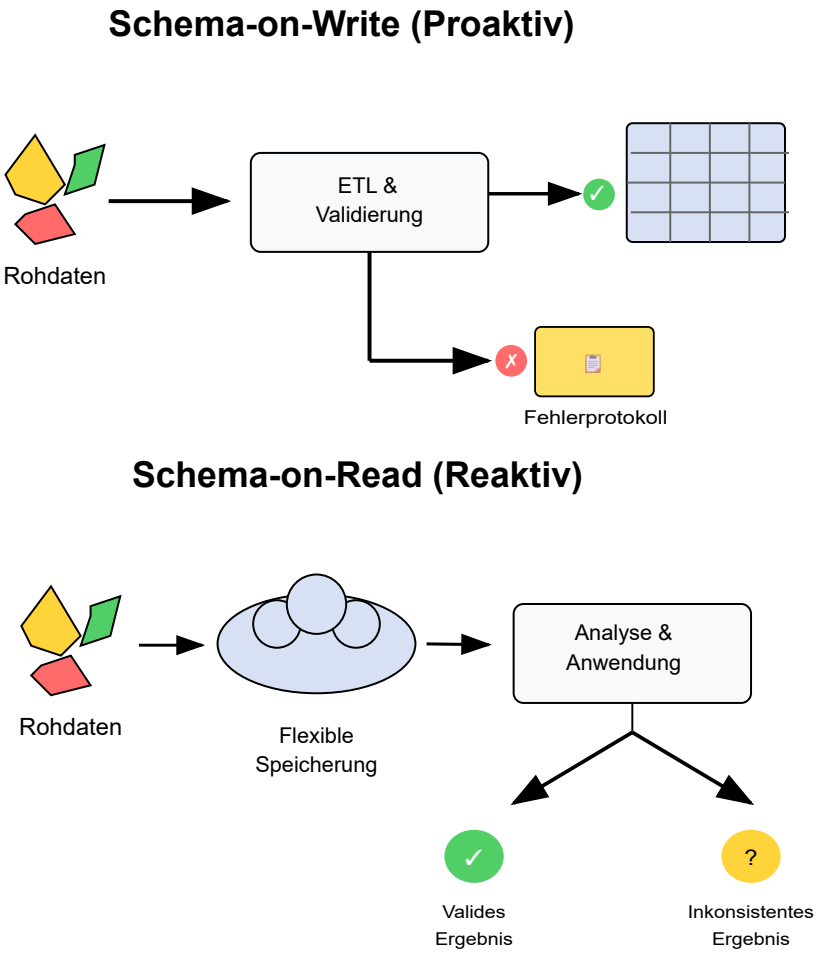


Abbildung 6.1: Vergleich der Datenqualitätssicherung bei Schema-on-Write (oben) und Schema-on-Read (unten). Schema-on-Write validiert Daten vor dem Schreiben und garantiert so eine hohe Basisqualität. Schema-on-Read speichert Rohdaten flexibel, die Qualitätssicherung erfolgt reaktiv durch die lesende Anwendung.

6.2 Moderne Datenplattformen

Über die Wahl einzelner Datenbanken hinaus haben sich in den letzten Jahren umfassende Architekturmuster für die unternehmensweite Datenhaltung etabliert. Konzepte wie Data Warehouses, Data Lakes und der neuere Data-Mesh-Ansatz haben jeweils spezifische Auswirkungen auf die Organisation und Sicherstellung der Datenqualität.

6.2.1 Data Lakes, Warehouses und Lakehouses

Data Warehouses wie Snowflake erlauben Cloud-basierte Skalierung, was Kosten senkt und Flexibilität erhöht.

Ein **Data Warehouse (DWH)** ist ein zentrales Repository für strukturierte, historisierte und aggregierte Daten, die aus verschiedenen operativen Systemen stammen. Die Daten werden durch einen ETL-Prozess (Extract, Transform, Load) transformiert und in ein wohldefiniertes, für Business-Intelligence-Zwecke optimiertes Schema geladen. Datenqualität ist hier ein zentrales Merkmal: Die Transformationsphase dient explizit der Bereinigung, Standardisierung und Integration der Daten.

Ein **Data Lake** verfolgt einen anderen Ansatz. Er ist ein zentraler Speicherort, der riesige Mengen an Daten in ihrem Rohformat aufnehmen kann, seien sie strukturiert, semi-strukturiert oder unstrukturiert. Die Idee ist, alle potenziell nützlichen Daten zu sammeln, ohne im Voraus über deren Verwendung oder Schema entscheiden zu müssen. Dies bietet maximale Flex202bildung für Data-Science- und Machine-Learning-Anwendungen.

Die Qualitätsherausforderungen in einem Data Lake sind immens. Ohne strikte Governance kann er schnell zu einem „Data Swamp"(Datensumpf) verkommen. Um dies zu verhindern, werden Data Lakes oft in Zonen unterteilt (z.B. Bronze/Raw, Silver/Cleansed, Gold/Curated). In der Bronze-Zone landen die Rohdaten unverändert. Im Übergang zur Silver-Zone finden erste Bereinigungen, Validierungen und Standardisierungen statt. Die Gold-Zone enthält schließlich aggregierte, qualitativ hochwertige Datenprodukte, die für spezifische Analysen oder Geschäftsbereiche aufbereitet sind. Die Verantwortung für die Qualitätssicherung liegt bei den Prozessen, die die Daten zwischen diesen Zonen bewegen.

Der Begriff Data Swamp beschreibt einen Data Lake, der aufgrund mangelnder Governance, fehlender Metadaten und unklarer Datenherkunft unbrauchbar geworden ist. Analysten finden entweder nicht, was sie suchen, oder sie misstrauen den Daten, die sie finden.

Apache Hadoop war ein Pionier für Data Lakes, indem es verteilte Speicherung mit HDFS ermöglichte.

To Do Vermeidung eines Data Swamps

1. **Metadaten-Management etablieren:** Jede in den Data Lake aufgenommene Datei muss mit Metadaten versehen werden, die ihre Herkunft, ihren Inhalt und ihre Qualität beschreiben. Ein Data Catalog ist hierfür essenziell (siehe Kapitel 5).
2. **Data Lineage verfolgen:** Es muss jederzeit nachvollziehbar sein, woher die Daten stammen und welche Transformationen sie durchlaufen haben.
3. **Zonen-Konzept durchsetzen:** Etabliere klare Regeln und Qualitätsgates für den Übergang von Daten zwischen den Zonen (z.B. von Raw zu Cleansed).
4. **Verantwortlichkeiten definieren:** Kläre, wer für die Qualität der Daten in den jeweiligen Zonen und für die Transformationsprozesse verantwortlich ist (Data Owner, Data Stewards).

Das **Data Lakehouse**-Konzept versucht, die Vorteile von Data Warehouses und Data Lakes zu kombinieren. Es implementiert Data-Warehouse-ähnliche Strukturen und Managementfunktionen (wie Transaktionen, Versionierung und Governance) direkt auf dem kostengünstigen Speicher eines Data Lakes. Technologien wie Delta Lake, Apache Iceberg und Apache Hudi sind Enabler dieses Ansatzes und bringen ACID-ähnliche Garantien in die Big-Data-Welt, was die Datenqualität und -zuverlässigkeit im Lake erheblich verbessert.

Delta Lake von Databricks fügt ACID-Transaktionen zu Spark-basierten Data Lakes hinzu.

6.2.2 Data Mesh als dezentraler Ansatz

Data Mesh wurde von Zhamak Dehghani während ihrer Zeit bei ThoughtWorks konzipiert. Der Ansatz zielt darauf ab, die Skalierbarkeit von Datenarchitekturen in großen, komplexen Organisationen zu verbessern.

Data Mesh ist ein relativ neues Paradigma, das die Probleme zentralisierter Datenplattformen (wie Engpässe und mangelnde fachliche Nähe des zentralen Datenteams) adressiert. Es ist ein soziotechnischer Ansatz, der auf einer dezentralen Architektur und domänengetriebener Verantwortung beruht. Data Mesh basiert auf vier Kernprinzipien:

1. **Domänenorientierte Datenverantwortung (Domain Ownership):** Die Verantwortung für Daten wird von einem zentralen Team in die Fachdomänen verlagert, die die Daten erzeugen oder am besten verstehen (z.B. Vertrieb, Produktion, Marketing).
2. **Daten als Produkt (Data as a Product):** Jede Domäne behandelt ihre Daten wie ein Produkt, das sie anderen Domänen zur Verfügung stellt. Dieses Datenprodukt hat einen verantwortlichen Product Owner, ist auffindbar, adressierbar, verständlich, vertrauenswürdig, interoperabel und sicher. Die Datenqualität wird zu einem expliziten Merkmal des Produkts, das über Service Level Objectives (SLOs) definiert und gemessen wird.
3. **Self-Service-Datenplattform (Self-Serve Data Platform):** Eine zentrale Plattform stellt den Domänenteams die Werkzeuge und Dienste zur Verfügung, die sie benötigen, um ihre Datenprodukte autonom zu erstellen, zu betreiben und zu verwalten. Dies reduziert die Reibung und Komplexität.
4. **Föderierte, computergestützte Governance (Federated Computational Governance):** Ein Gremium aus Vertretern der Domänen und der zentralen Plattform definiert globale Standards, Regeln und Richtlinien (z.B. für Sicherheit, Interoperabilität, Qualität). Diese Regeln werden so weit wie möglich automatisiert und in die Plattform eingebettet.

In Data Mesh werden Datenprodukte mit SLOs für Qualität wie Vollständigkeit und Aktualität versehen, ähnlich wie bei Software-Produkten.

Für die Datenqualität ist das Prinzip „Data as a Product“als revolutionär zu betrachten. Es verschiebt die Qualitätssicherung direkt an die Quelle, zu den Experten, die den fachlichen Kontext der Daten am besten kennen. Anstatt dass ein zentrales Team versucht, die Qualitätsregeln für alle Daten im Unternehmen zu verstehen und umzusetzen, ist nun das Domänenteam selbst dafür verantwortlich, ein hochwertiges, gut dokumentiertes und verlässliches Datenprodukt zu liefern. Dies fördert nicht nur die Qualität, sondern auch die Skalierbarkeit des gesamten Datenqualitätsmanagements.

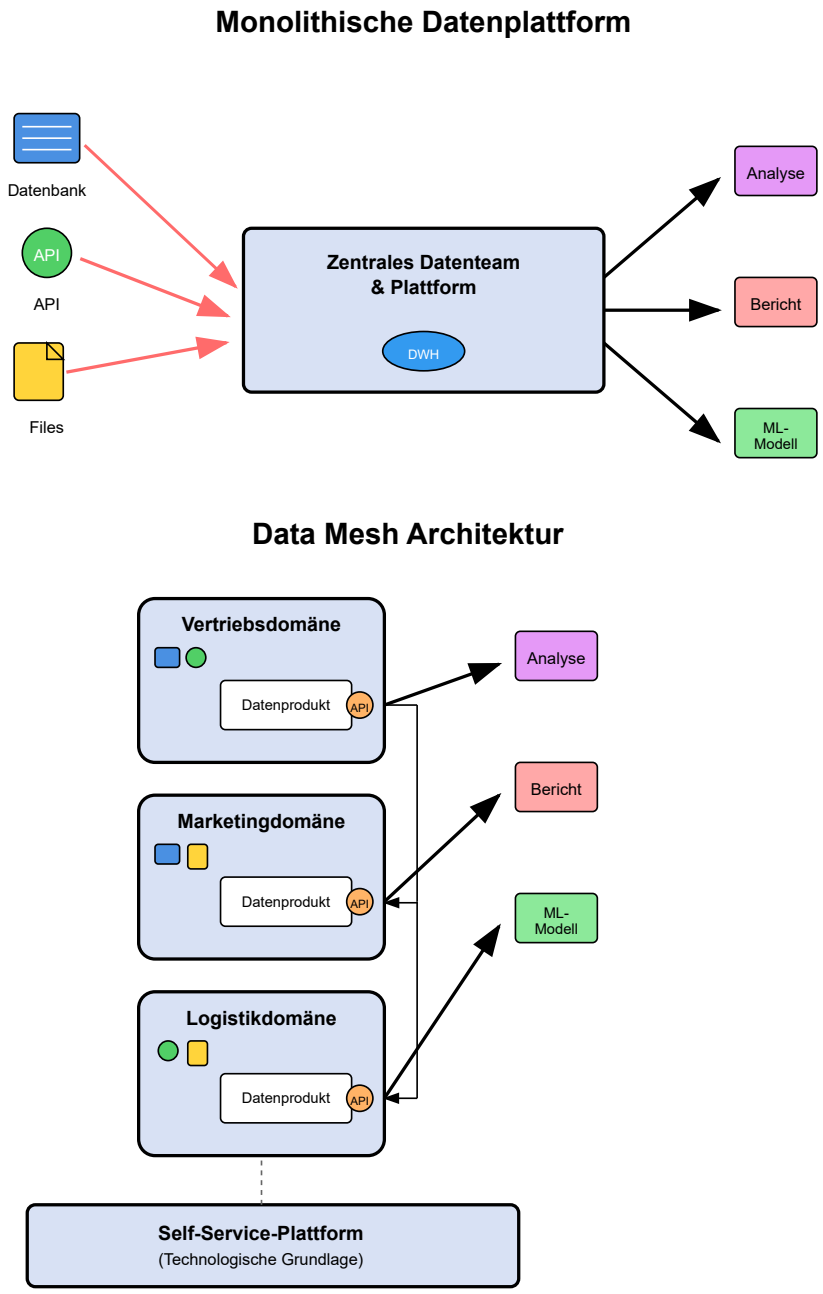


Abbildung 6.2: Vergleich einer zentralisierten, monolithischen Datenarchitektur (oben) mit der dezentralen, domänenorientierten Data-Mesh-Architektur (unten). Data Mesh verlagert die Verantwortung für Datenqualität in die Fachdomänen („Data as a Product“) und vermeidet so Engpässe.

6.3 Graphdatenbanken – Strukturierte Beziehungen und KI-Potenzial

Graphdatenbanken sind eine spezialisierte Art von NoSQL-Datenbanken, die für die Speicherung und Abfrage von Daten optimiert sind, deren Wert primär in den Beziehungen zwischen den Entitäten liegt. Anstatt Daten in Tabellen oder Dokumenten zu speichern, nutzen sie eine Struktur aus Knoten und Kanten, die es ermöglicht, komplexe Netzwerke und Zusammenhänge intuitiv und performant abzubilden.

Die Graphentheorie, die mathematische Grundlage von

Graphdatenbanken, geht auf Leonhard Euler und sein berühmtes Problem der Sieben Brücken von Königsberg im Jahr 1736 zurück. Er bewies, dass es unmöglich ist, einen Rundgang zu finden, bei dem jede Brücke genau einmal überquert wird.

6.3.1 Grundlagen

Das am weitesten verbreitete Modell ist das **Property-Graph-Modell**.

Ein **Property Graph** besteht aus folgenden Elementen:

- **Knoten (Nodes):** Repräsentieren Entitäten oder Objekte (z.B. eine Person, ein Unternehmen, ein Produkt). Knoten können Labels haben, die ihren Typ definieren (z.B. ':Person', ':Firma').
- **Kanten (Edges/Relationships):** Repräsentieren die Verbindungen oder Beziehungen zwischen Knoten. Kanten haben immer eine Richtung, einen Typ (z.B. ':KAUFT', ':ARBEITET_FÜR') und verbinden genau einen Start- mit einem Endknoten.
- **Eigenschaften (Properties):** Schlüssel-Wert-Paare, die sowohl an Knoten als auch an Kanten angehängt werden können, um diese mit zusätzlichen Attributen zu beschreiben (z.B. an einem ':Person'-Knoten die Eigenschaft '{name: "Anna", alter: 34}'; an einer ':KAUFT'-Kante die Eigenschaft '{datum: "2023-10-26"}').

Neo4j wird in der Panama Papers Untersuchung verwendet, um Verbindungen zwischen Offshore-Firmen aufzudecken.

Prominente Beispiele für Graphdatenbanken sind Neo4j, Amazon Neptune, TigerGraph und ArangoDB. Sie bieten spezielle Abfragesprachen wie Cypher (für Neo4j) oder Gremlin (ein Apache TinkerPop Standard), die für die Navigation durch das Netzwerk (Traversierung) optimiert sind.

6.3.2 Vorteile für Datenqualität und Analyse

Graphdatenbanken können Abfragen über Millionen von Knoten in Millisekunden ausführen, im Vergleich zu Stunden in relationalen Systemen für ähnliche Komplexität.

Die Stärke von Graphdatenbanken liegt in der expliziten Modellierung und Speicherung von Beziehungen. Während in einem RDBMS eine Beziehung durch einen 'JOIN' zwischen Tabellen zur Laufzeit berechnet werden muss, ist sie in einer Graphdatenbank eine physisch gespeicherte Entität. Dies führt zu einer extrem hohen Performance bei komplexen Abfragen, die viele Beziehungen durchlaufen (sogenannte Multi-Hop-Abfragen).

Aus Sicht der Datenqualität ermöglicht dieses Modell, die Konsistenz von Beziehungen direkt im Datenmodell abzubilden und zu validieren. Es wird einfacher, Fragen zu beantworten wie: „Gibt es Kunden, die ein Produkt bestellt,

aber nie bezahlt haben?“oder „Gibt es Zulieferer, die nicht mit einem gültigen Rahmenvertrag verbunden sind?“.

Solche Abfragen, die in SQL oft komplexe und potenziell langsame ‘JOIN‘-Operationen über mehrere Tabellen erfordern, sind in einer Graphabfragesprache oft einfacher und intuitiver auszudrücken. Ein Pfad wie

```
(k:Kunde)-[:HAT_BESTELLT]->(b:Bestellung)-[:ENTHÄLT]->(p:Produkt)
```

ist leicht verständlich und kann effizient abgefragt werden. Die Flexibilität des Modells erlaubt es zudem, neue Arten von Knoten und Beziehungen hinzuzufügen, ohne bestehende Strukturen zu beeinträchtigen, was die evolutionäre Weiterentwicklung der Datenmodelle vereinfacht.

6.3.3 Typische Anwendungsfälle und Verbindung zu KI

Graphdatenbanken eignen sich hervorragend für Anwendungsfälle, bei denen der Kontext und die Verbindungen zwischen Datenpunkten entscheidend sind.

Beispiel: Betrugserkennung mit Graphdatenbanken

Eine Versicherung möchte organisierten Versicherungsbetrug aufdecken. Betrüger nutzen oft komplexe Netzwerke aus Scheinadressen, geteilten Telefonnummern, Bankkonten und involvierten Personen, um ihre Spuren zu verwischen. In einem RDBMS ist die Aufdeckung solcher Ringe extrem schwierig. In einer Graphdatenbank werden Personen, Adressen, Konten und Schadensfälle als Knoten modelliert. Kanten stellen Beziehungen wie ‘(Person)-[:WOHNT_AN]->(Adresse)‘, ‘(Person)-[:NUTZT]->(Bankkonto)‘ und ‘(Schadensfall)-[:BETRIFFT]->(Person)‘ dar. Eine einfache Graphabfrage kann dann Muster identifizieren wie: „Finde alle Personen, die mit demselben Schadensfall verbunden sind und dieselbe Adresse oder dasselbe Bankkonto teilen, obwohl sie verschiedene Namen haben.“Solche Muster deuten stark auf Betrug hin und sind in einem Graphen schnell auffindbar.

Google’s Knowledge Graph nutzt Graphstrukturen, um Suchergebnisse mit semantischen Verbindungen zu bereichern.

Weitere typische Anwendungsfälle sind Empfehlungsdienste („Kunden, die dieses Produkt kauften, kauften auch jenes“), Wissensgraphen (Knowledge Graphs) zur Abbildung komplexen Domänenwissens und die Analyse von Lieferketten (Supply Chain Monitoring).

In der Pharmaindustrie werden Graphen verwendet, um Wechselwirkungen zwischen Medikamenten und Genen zu modellieren.

Die Verbindung zur Künstlichen Intelligenz (KI) ist besonders stark. Graphen sind ein ideales Format, um Daten mit

reichem Kontext für Machine-Learning-Modelle bereitzustellen. Techniken wie **Graph Embeddings** wandeln Knoten und ihre Nachbarschaft in numerische Vektoren um, die als Features für traditionelle ML-Modelle dienen können. Noch fortschrittlicher sind **Graph Neural Networks (GNNs)**, die direkt auf der Graphstruktur operieren und lernen, Muster in den Verbindungen zu erkennen. Dies wird beispielsweise zur Vorhersage von Interaktionen in sozialen Netzwerken oder zur Entdeckung neuer Medikamentenwirkungen in biologischen Netzwerken genutzt. Ferner unterstützen Graphen die Erstellung von semantischen Modellen und Ontologien, die eine Grundlage für erklärbare KI (Explainable AI) schaffen, da die Entscheidungswege eines Modells entlang der Kanten des Graphen nachvollzogen werden können.

GNNs haben in der Pandemie geholfen, Ausbreitungsmuster von Viren in sozialen Graphen zu prognostizieren.

Prompt Cypher-Abfrage zur Betrugserkennung

Erstelle eine Abfrage in der Graphabfragesprache Cypher für eine Neo4j-Datenbank. Die Datenbank enthält Knoten mit den Labels ':Person', ':Konto' und ':Telefon'. Personen können über die Beziehung ':NUTZT' mit Konten und Telefonnummern verbunden sein. Die Abfrage soll ein potenzielles Betrugsnetzwerk identifizieren, das durch einen „Betrugsring"definiert ist: Finde Gruppen von mindestens drei verschiedenen Personen, die alle direkt mit demselben Bankkonto oder derselben Telefonnummer verbunden sind. Gib die identifizierten Personen sowie das gemeinsam genutzte Konto oder Telefon zurück.

6.4 Zusammenfassung

Die in diesem Kapitel vorgestellten Konzepte verdeutlichen, dass die Datenarchitektur kein reines IT-Thema ist, sondern eine strategische Entscheidung mit tiefgreifenden Auswirkungen auf die Datenqualität. Bereits die fundamentale Wahl zwischen einem relationalen und einem NoSQL-Datenbanksystem definiert, ob Qualitätssicherung proaktiv durch ein starres **Schema-on-Write**-Modell oder reaktiv durch ein flexibles **Schema-on-Read**-Modell erfolgen muss. Während RDBMS durch ACID-Garantien und strikte Schemata eine hohe Basisqualität erzwingen, bieten NoSQL-Systeme die für Big-Data-Anwendungen notwendige Skalierbarkeit und Flexibilität, verlagern die Qualitätsverantwortung jedoch auf die Anwendungsebene.

Moderne Datenplattformen wie **Data Lakes** erweitern diese Herausforderung. Ohne rigorose Governance und klare

Prozesse, beispielsweise durch die Etablierung von Qualitätszonen und umfassendes Metadaten-Management, laufen sie Gefahr, zu unbrauchbaren „Data Swamps“zu werden. Das **Data Lakehouse** versucht, diese Lücke zu schließen, indem es Warehouse-Funktionalitäten wie ACID-Transaktionen in den Lake bringt. Einen Paradigmenwechsel stellt **Data Mesh** dar: Durch die Dezentralisierung der Datenverantwortung und das Prinzip „Data as a Product“wird Datenqualität zu einer Kernaufgabe der Fachexperten in den Domänen, was die Skalierbarkeit und Effektivität des Qualitätsmanagements in großen Organisationen erheblich steigern kann. Schließlich bieten **Graphdatenbanken** einzigartige Möglichkeiten, die Qualität und Konsistenz von Beziehungsdaten zu sichern und komplexe Netzwerke performant zu analysieren. Ihre Fähigkeit, Kontext explizit zu modellieren, macht sie zu einem wertvollen Werkzeug, insbesondere im Zusammenspiel mit KI-Methoden wie GNNs.

Rechtliche und ethische Rahmenbedingungen

7

Die Verwaltung und Nutzung von Daten findet nicht in einem Vakuum statt. Sie ist eingebettet in ein komplexes Gefüge aus gesetzlichen Vorschriften und ethischen Normen, die den Rahmen für den verantwortungsvollen Umgang mit Informationen vorgeben. In einer Zeit, in der Daten als „das neue Öl" bezeichnet werden und Algorithmen weitreichende Entscheidungen treffen, wächst die Bedeutung dieser Rahmenbedingungen stetig. Die Missachtung rechtlicher Vorgaben kann zu empfindlichen Strafen und Reputationsverlust führen, während die Vernachlässigung ethischer Prinzipien das Vertrauen der Gesellschaft untergräbt und zu systematischen Benachteiligungen führen kann.

Die Idee, Daten mit Öl zu vergleichen, wird oft dem britischen Mathematiker Clive Humby (2006) zugeschrieben. Er betonte, dass Daten, wie Öl, erst durch Veredelung (Analyse) wertvoll werden.

Dieses Kapitel beleuchtet die zentralen rechtlichen und ethischen Dimensionen, die für das Datenqualitätsmanagement von entscheidender Bedeutung sind. Es wird gezeigt, dass Datenqualität nicht nur ein technisches oder betriebswirtschaftliches Ziel ist, sondern auch eine rechtliche und moralische Verpflichtung darstellt. Wir beginnen mit der Datenschutzgrundverordnung (DSGVO), die explizite Anforderungen an die Datenrichtigkeit stellt. Anschließend werden die Auswirkungen des aufkommenden EU AI Acts auf die Qualität von Trainingsdaten für künstliche Intelligenz untersucht. Ferner wird der Zielkonflikt zwischen Datenschutz durch Anonymisierung und der Aufrechterhaltung der Datenqualität analysiert. Abschließend widmet sich das Kapitel den ethischen Aspekten von Fairness und Bias und zeigt auf, wie ein proaktiver, werteorientierter Ansatz die Grundlage für vertrauenswürdige und gerechte Datennutzung schafft.

7.1 Datenschutzgrundverordnung (DSGVO)

Die Datenschutzgrundverordnung der Europäischen Union, die seit dem 25. Mai 2018 in allen Mitgliedstaaten unmittelbar gilt, hat den Umgang mit personenbezogenen Daten grundlegend verändert. Sie etabliert ein umfassendes Regelwerk zum Schutz natürlicher Personen bei der Verarbeitung ihrer Daten und zum freien Datenverkehr. Obwohl die DSGVO primär auf den Schutz der Privatsphäre abzielt, hat sie direkte und weitreichende Implikationen für die Datenqualität.

Die DSGVO löste die Datenschutzrichtlinie 95/46/EG aus dem Jahr 1995 ab. Ein wesentlicher Unterschied ist der Verordnungscharakter, der eine EU-weit einheitliche Geltung ohne nationale Umsetzungsakte sicherstellt.

Einer der Grundsätze der Datenverarbeitung, verankert in Art. 5 DSGVO, ist der Grundsatz der „Richtigkeit". Demnach müssen personenbezogene Daten sachlich richtig und erforderlichenfalls auf dem neuesten Stand sein. Es sind alle angemessenen Maßnahmen zu treffen, damit unrichtige Daten unverzüglich gelöscht oder berichtigt werden. Diese Forderung macht Datenqualität von einer betriebswirtschaftlichen Notwendigkeit zu einer gesetzlichen Pflicht. Verstöße gegen die DSGVO können mit Bußgeldern von bis zu 20 Mio. Euro oder 4% des weltweiten Jahresumsatzes geahndet werden, je nachdem, welcher Betrag höher ist. Dies unterstreicht die finanzielle Relevanz von Compliance.

7.1.1 Das Recht auf Berichtigung (Art. 16 DSGVO)

Eine der stärksten rechtlichen Verankerungen der Datenqualität findet sich im Recht auf Berichtigung, das in Artikel 16 der DSGVO festgeschrieben ist. Dieser Artikel verleiht den betroffenen Personen ein direkt durchsetzbares Recht, die Korrektur von fehlerhaften Daten zu verlangen.

Das **Recht auf Berichtigung** (Art. 16 DSGVO) gibt der betroffenen Person das Recht, von dem Verantwortlichen unverzüglich die Berichtigung sie betreffender unrichtiger personenbezogener Daten zu verlangen. Ergänzend hat die betroffene Person das Recht, unter Berücksichtigung der Zwecke der Verarbeitung, die Vervollständigung unvollständiger Daten zu verlangen.

Der „Verantwortliche" ist die natürliche oder juristische Person, Behörde oder Einrichtung, die allein oder gemeinsam mit anderen über die Zwecke und Mittel der Verarbeitung von personenbezogenen Daten entscheidet. Er trägt die primäre Verantwortung für die Einhaltung der DSGVO.

Diese Bestimmung etabliert eine direkte gesetzliche Anforderung an die Datenrichtigkeit (Accuracy) und Vollständigkeit (Completeness), zwei zentrale Dimensionen der Datenqualität, wie in Kapitel 3 beschrieben. Unternehmen und andere Organisationen, die personenbezogene Daten verarbeiten, sind somit nicht nur aus eigenem Interesse, sondern per Gesetz dazu verpflichtet, Mechanismen zur Gewährleistung und Wiederherstellung der Datenrichtigkeit zu implementieren. Dies impliziert, dass reaktive Datenqualitätsprozesse vorhanden sein müssen, die es ermöglichen, auf Berichtigungsanfragen von betroffenen Personen zeitnah und effektiv zu reagieren. Die bloße Speicherung von Daten ohne regel-

Art. 25 DSGVO fordert zudem den Datenschutz durch Technikgestaltung und durch datenschutzfreundliche Voreinstellungen ('Privacy by Design and by Default'), was proaktive Datenqualitätsmaßnahmen impliziert.

mäßige Überprüfung oder die Möglichkeit zur Korrektur ist nicht mehr gesetzeskonform.

Beispiel: Korrektur einer Kundenadresse

Eine Kundin stellt im Online-Portal ihres Energieversorgers fest, dass ihre Adresse falsch geschrieben ist (z. B. „Hauptstrase 123" anstelle von „Hauptstraße 123"). Gemäß Art. 16 DSGVO kontaktiert sie den Kundenservice und verlangt die sofortige Korrektur. Der Energieversorger ist gesetzlich verpflichtet, dieser Aufforderung unverzüglich nachzukommen. Er muss nicht nur den Fehler im Kundendatensystem korrigieren, sondern auch sicherstellen, dass diese Korrektur an alle anderen Systeme weitergegeben wird, die diese Daten nutzen (z. B. das Abrechnungssystem oder das System für den postalischen Versand), um die Konsistenz zu wahren.

Das Recht auf Berichtigung erzwingt somit eine organisatorische und prozessuale Auseinandersetzung mit der Datenqualität. Es genügt nicht, fehlerhafte Daten zu ignorieren. Stattdessen müssen klare Prozesse für die Entgegennahme, Prüfung und Umsetzung von Korrekturanfragen etabliert werden. Eng verwandt mit dem Recht auf Berichtigung ist das Recht auf Löschung ('Recht auf Vergessenwerden') aus Art. 17 DSGVO, das die Entfernung irrelevanter oder unrechtmäßig verarbeiteter Daten fordert.

7.2 Auswirkungen des AI Act auf die Datenqualität

Während die DSGVO den Umgang mit personenbezogenen Daten regelt, zielt der von der Europäischen Kommission vorgeschlagene AI Act darauf ab, einen Rechtsrahmen für die Entwicklung und den Einsatz von Künstlicher Intelligenz (KI) zu schaffen. Ähnlich wie die DSGVO hat der AI Act einen extraterritorialen Anwendungsbereich: Er gilt für alle Anbieter, die KI-Systeme in der EU in Verkehr bringen oder einsetzen, unabhängig vom Standort des Anbieters. Dieser verfolgt einen risikobasierten Ansatz, bei dem die regulatorischen Anforderungen mit dem potenziellen Risiko eines KI-Systems für die Gesellschaft, Gesundheit, Sicherheit und Grundrechte steigen. Insbesondere für sogenannte Hochrisiko-KI-Systeme formuliert der AI Act strenge Anforderungen, von denen viele direkt auf die Qualität der zugrundeliegenden Daten abzielen.

Der AI Act ist die weltweit erste umfassende Gesetzesinitiative zur Regulierung von KI. Sein risikobasierter Ansatz unterteilt KI-Systeme in vier Kategorien: inakzeptables Risiko (verboten), hohes Risiko, begrenztes Risiko und minimales Risiko.

Ein **Hochrisiko-KI-System** ist laut AI Act ein System, das entweder selbst ein Produkt ist, das Sicherheitsvorschriften unterliegt (z. B. in Medizintechnik, Spielzeug), oder das in einem von acht spezifischen Bereichen eingesetzt wird, in denen erhebliche Grundrechtsrisiken bestehen. Dazu gehören unter anderem die biometrische Identifizierung, das Management kritischer Infrastrukturen, Bildung, Beschäftigung, Zugang zu wesentlichen Dienstleistungen und die Strafverfolgung.

Für diese Systeme werden die Anforderungen an die Datenqualität zu einer zentralen Säule der rechtlichen Konformität. Die Verpflichtungen lassen sich in mehreren Kernbereichen zusammenfassen.

Die Begriffe „Relevanz", „Repräsentativität" und „Vollständigkeit" sind direkt an die in Kapitel 3 diskutierten Datenqualitätsdimensionen gekoppelt und erhalten durch den AI Act eine rechtliche Verbindlichkeit.

Zentral ist die **Verpflichtung zu hochwertigen Trainings-, Validierungs- und Testdaten**. Der AI Acts fordert, dass diese Datensätze relevant, repräsentativ, frei von Fehlern und vollständig sind. Dies geht weit über eine rein technische Notwendigkeit hinaus und wird zur juristischen Voraussetzung für das Inverkehrbringen eines Hochrisiko-KI-Systems. Die Vermeidung und der Umgang mit Verzerrungen (**Bias**) sind dabei explizit genannt. Die Datensätze müssen die Bevölkerungsgruppen, für die das KI-System bestimmt ist, angemessen widerspiegeln, um diskriminierende Ergebnisse zu verhindern.

Darüber hinaus wird eine umfassende **Dokumentationspflicht und Nachvollziehbarkeit** gefordert. Die Datenquellen, der Erhebungskontext, die Vorverarbeitungsschritte und die angewendeten Bereinigungsverfahren müssen lückenlos dokumentiert werden. Dies entspricht dem Konzept der Datenprovenienz oder Data Lineage (siehe Abschnitt 5.2) und dient dazu, die Qualität und die Eigenschaften der Daten jederzeit überprüfen zu können.

To Do Audit-Prozess für KI-Daten

1. Definieren Sie einen Prozess zur Überprüfung und Dokumentation von Trainings-, Validierungs- und Testdaten.
2. Etablieren Sie Metriken zur Messung von Relevanz, Repräsentativität und Fehlerfreiheit der Daten.
3. Führen Sie regelmäßig Bias-Analysen durch, um potenzielle Diskriminierungen zu identifizieren.
4. Dokumentieren Sie alle Schritte der Datenvorverarbeitung und -bereinigung lückenlos (Data Lineage).

Um Innovation nicht zu hemmen, sieht der AI Act 'Regulatory Sandboxes' wesentliche Säule des AI Acts vor. Das sind kontrollierte Umgebungen, in denen Unternehmen KI-Systeme unter Aufsicht von Behörden testen können, bevor sie auf den Markt kommen.

Eine ist die **stärkere Governance über Datenquellen**. Entwickler von Hochrisiko-KI müssen Prozesse etablieren, um die Eignung ihrer Datenquellen kontinuierlich zu bewerten. Dies schließt auch die Governance für Datenerfassungs- und

Kennzeichnungsprozesse ein. Schließlich müssen Datenqualitätsprobleme als formales Risiko im **Risikomanagementsystem** des KI-Systems verankert und über dessen gesamten Lebenszyklus aktiv gemanagt werden.

Beispiel: KI im Bewerbungsmanagement

Ein Unternehmen entwickelt ein KI-System, das Bewerbungen vorsortiert und eine Rangliste der am besten geeigneten Kandidaten erstellt. Dieses System würde unter den AI Act als Hochrisiko-KI-System im Bereich „Beschäftigung" fallen. Um konform zu sein, müsste das Unternehmen nachweisen, dass die zum Training des Modells verwendeten Daten (z. B. Lebensläufe und Einstellungsentscheidungen der Vergangenheit) von hoher Qualität sind. Es müsste aktiv prüfen, ob die historischen Daten Vorurteile enthalten (z. B. eine systematische Bevorzugung männlicher Bewerber) und Maßnahmen zur Minderung dieses Bias ergreifen. Die Herkunft der Daten, die Annahmen bei der Datenaufbereitung und die Ergebnisse der Bias-Tests müssten minutiös dokumentiert werden.

7.3 Auswirkungen von Anonymisierung auf die Datenqualität

Die Anonymisierung von Daten ist eine zentrale Technik, um die Anforderungen des Datenschutzes zu erfüllen. Indem der Bezug zu einer identifizierbaren Person irreversibel entfernt wird, fallen die Daten nicht mehr unter den Anwendungsbereich der DSGVO. Dieser Schritt, der aus rechtlicher Sicht oft wünschenswert oder gar notwendig ist, hat jedoch tiefgreifende und oft nachteilige Auswirkungen auf die Datenqualität. Es entsteht ein fundamentaler Zielkonflikt zwischen dem Schutz der Privatsphäre und der Nützlichkeit der Daten für Analysen.

Die Wiederidentifizierung (Re-Identifikation) von als anonym geltenden Daten ist eine reale Gefahr. So konnten Forscher beispielsweise Personen im vermeintlich anonymisierten Netflix-Prize-Datensatz durch den Abgleich mit öffentlichen Filmbewertungen auf der Internet Movie Database (IMDb) deanonymisieren.

Anonymisierung ist der Prozess, personenbezogene Daten derart zu verändern, dass die betroffene Person nicht mehr oder nur mit einem unverhältnismäßig großen Aufwand an Zeit, Kosten und Arbeitskraft identifiziert werden kann. Im Gegensatz zur **Pseudonymisierung**, bei der die Zuordnung durch Hinzunahme zusätzlicher Informationen (z. B. eine separate Schlüsseldatei) wiederhergestellt werden kann, ist die Anonymisierung irreversibel.

Der primäre Effekt der Anonymisierung ist ein **Informationsverlust**, der sich auf verschiedene Datenqualitätsdimensionen auswirkt. Techniken wie die Generalisierung

(z. B. Ersetzen des genauen Alters durch eine Altersgruppe wie „30-39“) oder die Aggregation (z. B. Angabe des Durchschnittseinkommens für einen Postleitzahlenbereich statt einzelner Einkommen) führen zwangsläufig zu einem **Verlust an Granularität**. Fein-detaillierte Analysen auf individueller Ebene werden unmöglich. Dieser Informationsverlust kann zudem zu einer **Verzerrung statistischer Verteilungen** führen. Sei $X = \{x_1, \ldots, x_n\}$ ein Satz von Originaldatenpunkten, zum Beispiel das genaue Alter von n Personen. Eine Anonymisierungstechnik könnte die Daten in k Gruppen (Bins) $B_1, \ldots, B_k$ einteilen und jeden Wert x_i durch den Mittelwert seines Bins B_j ersetzen. Der neue Datensatz X' hat dann eine andere Verteilung. Die ursprüngliche Varianz wird künstlich reduziert. Die Varianz des anonymisierten Datensatzes $\mathrm{Var}(X')$ wird in der Regel geringer sein als die des Originaldatensatzes:

Sogenannte Quasi-Identifikatoren sind Attribute, die für sich genommen nicht eindeutig sind, aber in Kombination eine Person identifizierbar machen können (z. B. Postleitzahl, Geburtsdatum und Geschlecht).

$$\mathrm{Var}(X') \leq \mathrm{Var}(X) \tag{7.1}$$

Dies kann die Ergebnisse von Regressionsanalysen oder anderen statistischen Modellen erheblich beeinflussen. Ein bekanntes Anonymisierungkonzept ist die **k-Anonymität**. Ein Datensatz wird als k-anonym bezeichnet, wenn jeder Datensatz bezüglich einer Reihe von quasi-identifizierenden Attributen (z. B. Alter, PLZ, Geschlecht) von mindestens $k-1$ anderen Datensätzen nicht unterscheidbar ist. Dies erschwert die Re-Identifikation erheblich.

Des Weiteren wird die **Nachvollziehbarkeit und Datenverknüpfung** stark beeinträchtigt. Werden Identifikatoren entfernt oder generalisiert, können Datensätze aus unterschiedlichen Quellen nicht mehr miteinander verknüpft werden, um ein umfassenderes Bild zu erhalten (z. B. die Verknüpfung von Kundendaten mit Support-Tickets). Die Eignung für **individuelle oder kontextbezogene Analysen**, wie sie für personalisierte Medizin oder maßgeschneiderte Produktempfehlungen erforderlich ist, wird stark reduziert. Paradoxerweise führt Anonymisierung aber zu einer **Steigerung der regulatorischen Datenqualität**. Aus Sicht der Compliance-Abteilung ist ein anonymisierter Datensatz qualitativ hochwertiger, da er das Risiko von Datenschutzverletzungen und den damit verbundenen Strafen minimiert. Dies zeigt, wie kontextabhängig der Begriff der Datenqualität sein kann. Differential Privacy ist ein modernerer Ansatz, der mathematische Garantien bietet, dass das Hinzufügen oder Entfernen eines einzelnen Datensatzes das Analyseergebnis nicht wesentlich verändert, was den Schutz der Privatsphäre stärkt.

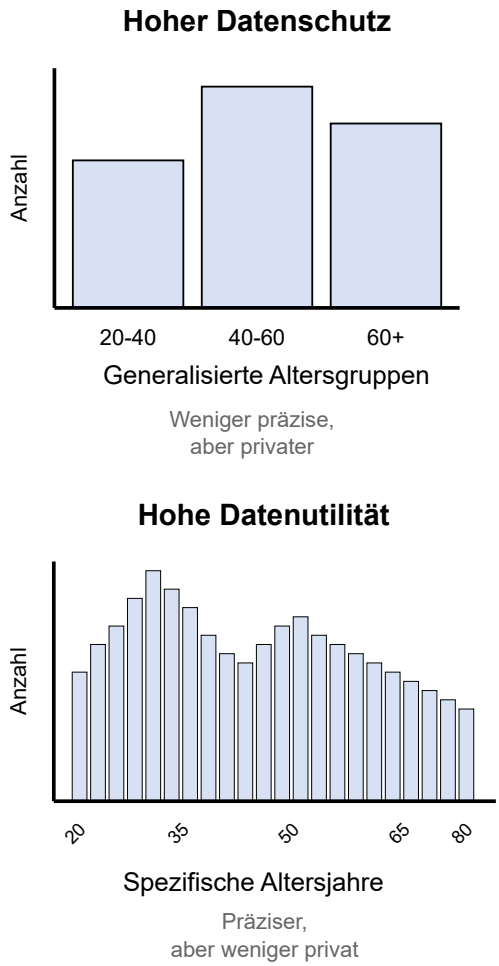


Abbildung 7.1: Der Trade-off zwischen Datenschutz durch Anonymisierung und Datenqualität. Starke Anonymisierung (oben) führt zum Verlust von Granularität und Informationsgehalt, während hohe Datenutilität (unten) oft detailliertere und damit potenziell identifizierbare Daten erfordert.

7.4 Ethische Aspekte: Fairness und Bias in Daten

Die Einhaltung von Gesetzen wie der DSGVO und dem zukünftigen AI Act ist eine notwendige, aber keine hinreichende Bedingung für einen verantwortungsvollen Umgang mit Daten. Über die rechtlichen Mindestanforderungen hinaus gibt es eine wachsende gesellschaftliche Erwartung, dass Daten ethisch integer gehandhabt werden. Im Mittelpunkt der Datenethik stehen Konzepte wie Fairness, Verantwortlichkeit und Transparenz. Ein zentrales Problemfeld, das sowohl technische als auch ethische Dimensionen hat, ist der Umgang mit Bias in Daten.

Potter Stewart, ein ehemaliger Richter am Obersten Gerichtshof der USA, formulierte treffend: „Ethik ist das Wissen um den Unterschied zwischen dem, was du tun darfst, und dem, was richtig ist zu tun."

Datenfehler und Bias können zu diskriminierenden Algorithmen führen. Algorithmen des maschinellen Lernens lernen Muster aus den Daten, mit denen sie trainiert werden. Wenn diese Daten historische oder gesellschaftliche Vorurteile widerspiegeln, werden die Algorithmen diese Vorurteile nicht nur reproduzieren, sondern potenziell sogar verstärken. Die Kausalkette ist ebenso einfach wie verheerend: Historisch

Historischer Bias entsteht, wenn die Daten vergangene gesellschaftliche Ungerechtigkeiten oder Vorurteile widerspiegeln, die ein KI-Modell dann als zu lernendes Muster aufgreift und fortschreibt.

verzerrte Daten führen zu einem verzerrten Modell, das wiederum zu diskriminierenden Entscheidungen in der realen Welt führt. Schlechte Datenqualität in Form von systematischer Verzerrung führt hier direkt zu ethisch inakzeptablen und oft auch illegalen Ergebnissen.

Beispiel: Amazons diskriminierendes Recruiting-Tool

Ein bekanntes Negativbeispiel ist ein von Amazon entwickeltes KI-Tool zur Automatisierung der Bewerber-Vorauswahl. Das System wurde mit den Lebensläufen der Bewerber der vorangegangenen zehn Jahre trainiert. Da die Tech-Industrie in dieser Zeit stark männderdominiert war, lernte das Modell, dass männliche Kandidaten zu bevorzugen seien. Es bestrafte Lebensläufe, die Wörter wie „Frauen-“ enthielten (z. B. bei der Nennung eines „Frauen-Schachclubs“) oder die auf den Besuch von reinen Frauen-Colleges hindeuteten. Obwohl das Geschlecht als Merkmal nicht explizit verwendet wurde, lernte das Modell den Bias aus den historischen Daten. Amazon hat das Projekt schließlich eingestellt.

Der COMPAS-Algorithmus, der in US-Gerichten zur Vorhersage der Rückfallwahrscheinlichkeit von Straftätern eingesetzt wird, stand stark in der Kritik. Eine ProPublica-Analyse aus dem Jahr 2016 zeigte, dass der Algorithmus bei schwarzen Angeklagten fälschlicherweise eine doppelt so hohe Rückfallquote vorhersagte wie bei weißen Angeklagten.

Der verantwortungsvolle Umgang mit Daten, oft als **Datenethik**bezeichnet, erfordert einen proaktiven Ansatz zur **Erkennung und Minderung von Bias**. Die Erkennung (Detection) kann durch statistische Analysen erfolgen. Ein gängiges Fairness-Kriterium ist die demografische Parität. Für ein Modell, das eine positive Entscheidung $\hat{Y} = 1$ (z. B. Kredit bewilligt) vorhersagt, und eine geschützte Gruppe A (z. B. Ethnie) sollte gelten:

$$P(\hat{Y} = 1|A = \text{Gruppe 1}) \approx P(\hat{Y} = 1|A = \text{Gruppe 2}) \quad (7.2)$$

Die Minderung (Mitigation) von Bias kann in verschiedenen Phasen des Modellentwicklungsprozesses ansetzen: durch Anpassung der Trainingsdaten (Pre-Processing), durch Einbau von Fairness-Bedingungen in die Lernalgorithmen (In-Processing) oder durch Korrektur der Modellergebnisse (Post-Processing).

Die Forschungsgemeinschaft rund um ’Fairness, Accountability, and Transparency in Machine Learning’ (FAT/ML) entwickelt aktiv technische und ethische Lösungsansätze, um die hier diskutierten Probleme zu adressieren.

Ein weiterer Eckpfeiler der Datenethik ist die **Datenprovenienz**. Das Wissen um die Herkunft der Daten, ihren Erhebungskontext und ihre Transformationsgeschichte ist unerlässlich, um das Potenzial für Bias einschätzen zu können und Vertrauen in die Daten und die darauf basierenden Analysen aufzubauen. Transparenz über die Datenherkunft ist eine Grundvoraussetzung für eine ethische Bewertung.

7.5 Zusammenfassung

Dieses Kapitel hat die tiefgreifende Verflechtung von Datenqualität mit rechtlichen und ethischen Rahmenbedingungen aufgezeigt. Es wurde deutlich, dass Datenqualität weit mehr ist als eine rein technische Disziplin; sie ist eine fundamentale Anforderung für gesetzeskonformes und ethisch verantwortungsvolles Handeln in einer datengetriebenen Welt.

Die zentralen Erkenntnisse lassen sich wie folgt zusammenfassen: Rechtliche Rahmenbedingungen wie die DSGVO, insbesondere durch das Recht auf Berichtigung in Art. 16, und der kommende AI Act erheben Datenqualitätsdimensionen wie Richtigkeit, Vollständigkeit und Fairness zu einklagbaren gesetzlichen Pflichten. Die Nichteinhaltung kann erhebliche finanzielle und rufschädigende Konsequenzen haben. Die Technik der Anonymisierung, die für den Datenschutz unerlässlich ist, steht in einem direkten Zielkonflikt mit der Datenqualität. Sie führt unweigerlich zu Informationsverlust und potenziellen statistischen Verzerrungen, was einen sorgfältigen Trade-off zwischen Privatsphäre und Datennützlichkeit erfordert. Ethische Überlegungen, allen voran das Streben nach Fairness und die Vermeidung von Bias, gehen über die gesetzlichen Mindestanforderungen hinaus. Die Erkennung und Minderung von systematischer Verzerrung in Daten ist eine entscheidende Aufgabe, um diskriminierende Ergebnisse durch algorithmische Systeme zu verhindern. Konzepte der Daten-Governance, insbesondere eine transparente Datenprovenienz (Data Lineage), sind essenzielle Werkzeuge, um sowohl die rechtlichen Anforderungen zu erfüllen als auch eine Grundlage für ethische Bewertungen und Vertrauen zu schaffen.

8 Feature Reliability Score

In Kapitel 3: "Dimensionen der Datenqualität" wurde bereits die Bedeutung und Berechnung der Datenqualität hervorgehoben.

Hier wird ein Messkonzept - der Feature Reliability Score - vorgestellt, der systematisch die Ausführungen aus Kapitel 3 in einen Gesamt-Score umsetzt.

Der Feature Reliability Score (FRS) ist ein Konzept, das darauf abzielt, die Verlässlichkeit eines Merkmals anhand verschiedener Qualitätsdimensionen quantitativ zu bewerten.

Unity verarbeitete fehlerhafte Daten eines Großkunden. Dies führte zu einer Verzerrung der Trainingsdaten für die Machine-Learning-Algorithmen und reduzierte die Werbeeinnahmen um 110 Mio. USD ([8]).

Zunächst ein wichtiger Hinweis, um "entspannter" mit Qualitätsmessungen der eigenen Daten umzugehen:

Hinweis:

Felder, deren Inhalte schlechte Datenqualität aufweisen, müssen nicht zwingend kostenintensiv bereinigt oder nacherfasst werden. Für Datenmodelle (z.B. logitische Regression, Neuronale Netze etc.) können jedoch signifikant bessere Ergebnisse erzielt werden, wenn Inhalte minderer Qualität nicht berücksichtigt werden, d.h. das Weglassen von "schlechter" Qualität kann das Modellergebnis signifikant erhöhen.

8.1 Grundlagen des Feature Reliability Scores

Bevor die einzelnen Dimensionen des Feature Reliability Scores detailliert betrachtet werden, ist es notwendig, ein grundlegendes Verständnis für seine Definition, Zielsetzung und die beteiligten Komponenten zu entwickeln.

8.1.1 Definition und Zielsetzung des FRS

Der Feature Reliability Score macht die Datenqualität einzelner Felder vergleichbar und ist wie folgt definiert:

Der **Feature Reliability Score (FRS)** ist eine quantitative Metrik, die den Grad der Verlässlichkeit und Vertrauenswürdigkeit eines einzelnen Datenmerkmals (Spalte in einer Tabelle, Attribut eines Objekts) in einem Datensatz zusammenfasst. Er aggregiert Bewertungen aus

verschiedenen Datenqualitätsdimensionen zu einem einzelnen Wert oder einem Vektor von Werten.

Der FRS ist kein standardisierter Score, sondern muss oft an den spezifischen Anwendungsfall angepasst werden.

Das primäre Ziel des Feature Reliability Score ist es, eine schnelle und vergleichbare Einschätzung der Qualität einzelner Merkmale zu ermöglichen. Dies unterstützt Datenanalysten, Data Scientists und Fachexperten bei verschiedenen Aufgaben.

Hinweis:

Nur ursprünglich erfasste, nicht berechnete Felder sollten für Feature Reliability Scores berücksichtigt werden. Berechnete Felder können Qualitätsmängel der zugrunde liegenden Daten verschleiern und zu einer irreführenden Bewertung der Datenqualität führen, da sie keine eigenständigen Informationen liefern, sondern von der Zuverlässigkeit der Originaldaten abhängen. Nur qualitätsgesicherte Originaldaten können anschließend zu weiteren Kennziffern zusammengeführt werden.

Hat man beispielsweise ein zuverlässiges Feld wie "Körpergewicht" und ein unzuverläßiges Feld wie "Körpergröße", das nur mit einem Standard-Wert von 1,80m belegt ist, dann wird ein Quotient aus Gewicht und Körpergröße nach wie vor eine gute satistische Qualität haben. Das Feld als solches hat aber keine zusätzliche Aussagekraft.

Der FRS hilft bei der **Identifikation** von Merkmalen mit potenziell geringer Qualität, die einer genaueren Untersuchung oder Bereinigung bedürfen. So können unzuverlässige oder fehlerhafte Daten erkannt und gezielt bereinigt oder weiter analysiert werden.

Durch die Bewertung der Datenqualität unterstützt der FRS die **Entscheidungshilfe** bei der Merkmalsauswahl (**Feature Selection**) für Modelle des maschinellen Lernens. Zuverlässige Merkmale werden ausgewählt, um die Modellgenauigkeit und -stabilität zu verbessern.

Der FRS ermöglicht das **Monitoring** der Datenqualität über die Zeit. Veränderungen oder Verschlechterungen in den Daten können frühzeitig erkannt werden, um die Datenintegrität langfristig zu sichern.

Der FRS bietet eine klare und verständliche Metrik zur **Kommunikation** der Datenqualität an Stakeholder. Dies fördert transparente und fundierte Entscheidungen auf Basis der Datenqualität.

Ein gut konzipierter FRS kann somit maßgeblich zur Effizienzsteigerung in Datenprojekten und zur Verbesserung der Ergebnisqualität beitragen.

Beispiel: Anwendung im Kreditrisikomanagement

Eine Bank möchte ein Modell zur Vorhersage von Kreditausfällen entwickeln. Dafür steht ein umfangreicher Datensatz mit Merkmalen wie Einkommen, Alter, Beschäftigungsdauer, Kredithöhe, bisherige Zahlungsmoral etc. zur Verfügung. Bevor das Modell

trainiert wird, berechnet das Data-Science-Team für jedes Merkmal einen FRS. Merkmale mit einem sehr niedrigen FRS, beispielsweise aufgrund eines hohen Anteils fehlender Werte oder vieler ungültiger Einträge, werden entweder priorisiert bereinigt oder vorerst vom Modelltraining ausgeschlossen. Das Merkmal „Anzahl Kinder" könnte beispielsweise einen niedrigen FRS aufweisen, wenn es in vielen Altdatensätzen nicht erfasst wurde.

Der Feature Reliability Score ist auch in der Produktentwicklung und im industriellen Qualitätsmanagement ein relevantes Konzept.

8.1.2 Komponenten des Scores

Der FRS ist typischerweise ein Komposit-Score, der sich aus Bewertungen verschiedener Datenqualitätsdimensionen zusammensetzt. Die Auswahl und Gewichtung dieser Dimensionen hängt stark vom Kontext und den spezifischen Anforderungen ab. Zu den gängigsten Dimensionen, die in diesem Kapitel detailliert behandelt werden, gehören:

1. **Füllgrad (Completeness)**: Wie vollständig ist das Merkmal ausgefüllt? (siehe Abschnitt 3.1.1)
2. **Diversität (Distinctness/Uniqueness)**: Wie viele unterschiedliche Werte weist das Merkmal auf? (siehe Abschnitt 3.2.1 für verwandte Konzepte)
3. **Klumpenbildung (Clumpiness)**: Wie stark konzentrieren sich die beobachteten Werte auf wenige dominante Ausprägungen?
4. **Verteilung (Value Distribution / Entropie)**: Wie sind die Werte des Merkmals verteilt?
5. **Plausibilität (Validity)**: Entsprechen die Werte vordefinierten Regeln und Wertebereichen? (siehe Abschnitt 3.2.2)
6. **Konsistenz zu anderen Feldern (Cross-field Consistency)**: Sind die Werte des Merkmals logisch konsistent mit Werten in anderen Merkmalen desselben Datensatzes? (siehe Abschnitt 3.1.3)
7. **Ausreißer-Belastung (Outlier Score)**: Wie viele extreme Werte enthält das Merkmal? (siehe Kapitel 9)

Jede dieser Dimensionen beleuchtet einen anderen Aspekt der Merkmalsqualität und trägt dazu bei, ein umfassendes Bild der Verlässlichkeit zu zeichnen. Weitere mögliche Dimensionen könnten Aktualität (Timeliness) oder Genauigkeit (Accuracy) sein, deren Messung oft aufwändiger ist.

8.2 Füllgrad (Completeness)

Der Füllgrad ist eine der fundamentalsten und am einfachsten zu ermittelnden Dimensionen der Datenqualität. Er gibt an, inwieweit ein Datenmerkmal mit Werten befüllt ist oder ob es fehlende Einträge (Missing Values) aufweist. Eine detaillierte Einführung in die Vollständigkeit als Qualitätsdimension findet sich in Abschnitt 3.1.1.

Hinweis:

Fehlende Werte können wichtige Informationen enthalten. Bei einer fehlenden Bilanzzahl eines Unternehmens in einer Kreditanalysedatenbank bei einer Bank gibt es im Wesentlichen vier Möglichkeiten: Die Zahl wurde von der Bank nicht erfasst, vom Unternehmen vergessen zu melden, vom Unternehmen absichtlich nicht gemeldet oder sie existiert nicht. Je nach Ursache hat das Fehlen eine unterschiedliche Bedeutung und erfordert eine differenzierte Interpretation.

Es ist wichtig, NULL-Werte klar von Leerstrings (') oder Werten wie "unbekannt" zu unterscheiden, da diese unterschiedliche semantische Bedeutungen haben können.

8.2.1 Definition und Berechnung

Der Anteil fehlender Werte P_{missing} ist dann:

$$P_{\text{missing}} = \frac{N_{\text{missing}}}{N}$$

Der Score für den Füllgrad S_{compl} wird dann üblicherweise als

$$S_{\text{compl}} = 1 - P_{\text{missing}} = 1 - \frac{N_{\text{missing}}}{N} = \frac{N - N_{\text{missing}}}{N}$$

definiert. Dieser Score liegt im Bereich $[0, 1]$, wobei 1 einen vollständig gefüllten Datensatz ohne fehlende Werte und 0 einen komplett leeren Datensatz (nur fehlende Werte) repräsentiert.

Beispiel: Füllgradberechnung für Kundenalter

Ein Datensatz enthält Informationen zu 1.000 Kunden. Das Merkmal "Alter" weist in 50 Fällen keinen Eintrag auf.

- $N = 1.000$
- $N_{\text{missing}} = 50$
- $P_{\text{missing}} = \frac{50}{1.000} = 0,05$
- $S_{\text{completeness}} = 1 - 0,05 = 0,95$

Der Füllgrad-Score für das Merkmal Älter" beträgt 0,95, was bedeutet, dass 95% der Alterseinträge vorhanden sind.

8.2.2 Interpretation und Auswirkungen

Ein hoher Füllgrad-Score ist in der Regel wünschenswert, da er anzeigt, dass für die meisten Beobachtungen Informationen im betreffenden Merkmal vorhanden sind. Ein

niedriger Score hingegen deutet auf eine hohe Anzahl fehlender Werte hin, was verschiedene negative Auswirkungen haben kann:

- **Verzerrte Analysen:** Wenn fehlende Werte nicht zufällig auftreten (Missing Not At Random - MNAR), können Analysen und Modelle verzerrte Ergebnisse liefern.
- **Reduzierte Stichprobengröße:** Viele statistische Verfahren und Algorithmen des maschinellen Lernens können nicht direkt mit fehlenden Werten umgehen und erfordern deren Entfernung (Listenweiser Ausschluss) oder Imputation. Dies kann die effektive Stichprobengröße reduzieren und die statistische Aussagekraft schwächen.
- **Verlust von Information:** Jede fehlende Information kann potenziell wertvoll sein.

Hinweis:

Fehlende Daten, die für fundierte Schlussfolgerungen unerlässlich sind (z.B. Bilanzkennziffern bei Kreditanalysen), sollten gründlich untersucht werden. Das Auslassen solcher Datensätze kann zu erheblich verzerrten Analysen und fehlerhaften Schlussfolgerungen führen.

Die Interpretation, ab wann ein Füllgrad als "gut" oder "schlecht" gilt, ist stark kontextabhängig. Für kritische Merkmale kann bereits ein geringer Anteil fehlender Werte problematisch sein, während bei weniger wichtigen Merkmalen auch höhere Fehlerraten tolerierbar sein können.

Branchenspezifische Akzeptanzschwellen für den Füllgrad sind üblich. In der medizinischen Forschung können z.B. strengere Anforderungen gelten als im E-Commerce.

8.2.3 Strategien zur Verbesserung des Füllgrads

Wenn ein Merkmal einen unzureichenden Füllgrad aufweist, können verschiedene Strategien zur Verbesserung in Betracht gezogen werden:

- **Datenquelle überprüfen:** Liegt das Problem bei der Datenerfassung oder -übertragung? Können die fehlenden Werte nachträglich aus der Quelle beschafft werden?
- **Imputationstechniken**: Fehlende Werte können durch Schätzwerte ersetzt werden. Gängige Methoden umfassen:

  1. Mittelwert-/Median-/Modus-Imputation
  2. Regressionsimputation
  3. K-Nächste-Nachbarn-Imputation (KNN)
  4. Multiple Imputationmm]Bei der **K-Nächste-Nachbarn-Imputation** (KNN-Imputation) werden die fehlenden Werte einer Variablen basierend auf den

Bei der **Regressionsimputation** werden die fehlenden Werte einer Variablen (der abhängigen Variable) basierend auf den bekannten Werten anderer Variablen (der unabhängigen Variablen oder Prädiktoren) im Datensatz mit einer Regression vorhergesagt.

Werten der "ähnlichsten" Beobachtungen im Datensatz geschätzt. Dabei werden die K nächstgelegenen Datenpunkte als Referenz herangezogen.

- **Entfernung des Merkmals:** Wenn der Anteil fehlender Werte extrem hoch ist und eine Imputation nicht sinnvoll erscheint, kann das gesamte Merkmal aus der Analyse ausgeschlossen werden.

To Do Füllgradanalyse

1. Identifiziere alle Merkmale mit einem Füllgrad unter einem definierten Schwellenwert (z.B. 90%).
2. Analysiere für jedes dieser Merkmale die möglichen Ursachen für die fehlenden Werte (z.B. optionale Eingabefelder, technische Fehler, nachträglich hinzugefügte Felder).
3. Entscheide pro Merkmal, ob eine Imputation sinnvoll ist oder ob alternative Strategien (z.B. Nacherfassung, Entfernung) verfolgt werden sollen. Dokumentiere die Entscheidungen.

8.3 Diversität (Distinctness)

Die Diversität eines Merkmals gibt Aufschluss darüber, wie viele unterschiedliche Werte es enthält. Dies ist ein wichtiger Indikator, um die Informationshaltigkeit und potenzielle Probleme wie Quasi-Konstanten oder fehlerhafte Datenerfassung zu identifizieren. Verwandte Konzepte zur Eindeutigkeit sind in Abschnitt 3.2.1 beschrieben.

8.3.1 Definition und Berechnung

Uniqueness ist ein Spezialfall von Distinctness, bei dem die Anzahl einzigartiger Werte gleich der Anzahl der Zeilen ist ($S_{\text{distinctness}} = 1$).

Der Score für die Diversität $S_{\text{distinctness}}$ wird wie folgt berechnet:

Sei N die Gesamtzahl der Zeilen. Sei U die Anzahl der einzigartigen (distinct) Werte im Merkmal.

$$S_{\text{distinctness}} = \frac{U}{N}$$

Der Distictness-Score zählt nur das Auftreten eines Wertes. Wie oft er auftritt, spielt keine Rolle. Hierfür ist der Distribution-Score "zuständig".

Dieser Score liegt im Bereich $[\frac{1}{N}, 1]$. Ein Wert nahe 1 (oder genau 1) bedeutet eine hohe Diversität, d.h., viele oder alle Werte sind unterschiedlich. Ein Wert nahe $\frac{1}{N}$ (oder genau $\frac{1}{N}$ bei $U = 1$) bedeutet eine sehr geringe Diversität, im Extremfall ist nur ein einziger Wert vorhanden (das Merkmal ist eine Konstante).

Beispiel: Diversität für verschiedene Feldtypen

Ein Datensatz hat 1000 Zeilen.

Merkmal "KundenID“: Enthält 1.000 einzigartige IDs. $U = 1.000$, $N = 1.000$. $S_{\text{distinctness}} = \frac{1.000}{1.000} = 1,0$. Dies erwartet man für ein Primärschlüsselfeld.

Merkmal "Geschlecht“: Enthält die Werte "männlich“, "weiblich“, "divers“. $U = 3$, $N = 1.000$.
$S_{\text{distinctness}} = \frac{3}{1.000} = 0,003$. Ein niedriger Score ist für kategoriale Merkmale mit wenigen Ausprägungen normal und erwartet.

Merkmal "Newsletter abonniert“: Enthält in 998 Fällen "Ja“ und in 2 Fällen "Nein“. $U = 2$, $N = 1.000$. $S_{\text{distinctness}} = \frac{2}{1.000} = 0,002$. Obwohl nur zwei Werte, ist die Verteilung hier sehr ungleich (siehe Score Verteilung).

Merkmal "Produktkategorie“: Enthält nur den Wert Ëlektronik“. $U = 1$, $N = 1.000$. $S_{\text{distinctness}} = \frac{1}{1.000} = 0,001$. Dies deutet auf ein konstantes oder quasi-konstantes Merkmal hin.

8.3.2 Interpretation und Fallstricke

Der Datentyp beeinflusst die erwartete Diversität. Numerische kontinuierliche Variablen haben tendenziell eine höhere Diversität als kategoriale Variablen.

Die Interpretation des Diversitäts-Scores hängt stark von der erwarteten Natur des Merkmals ab. Ein **hoher Diversitäts-Score** (nahe 1) ist für **Identifikatoren** wie Primärschlüssel oder Transaktions-IDs erwartet und wünschenswert, da diese Merkmale eindeutig sein sollten. Bei anderen Merkmalstypen kann ein hoher Score jedoch auf eine sehr hohe Kardinalität oder sogar auf **Datenfehler** hinweisen, etwa durch Tippfehler, die als unterschiedliche Werte interpretiert werden.

Ein **mittlerer Diversitäts-Score** ist typisch für **kategoriale Merkmale** mit mehreren Ausprägungen, wie Produktkategorien oder Länder, bei denen eine moderate Vielfalt erwartet wird.

Ein **geringer Diversitäts-Score** (nahe 0) kann für **binäre Merkmale** (z.,B. Ja/Nein) oder Merkmale mit wenigen legitimen Ausprägungen normal sein. Ein extrem niedriger Wert, etwa $U = 1$ oder $U = 2$ bei Tausenden von Zeilen, kann jedoch bei Merkmalen, von denen mehr Vielfalt erwartet wird, auf Probleme hinweisen.

Ein solches Problem könnte ein **konstantes Merkmal** sein, das keine Information zur Unterscheidung der Datensätze beiträgt. Solche Merkmale sind für viele Analysen und Modelle nutzlos.

Ein weiteres Problem könnte eine **fehlerhafte Datenerfassung** sein, bei der immer derselbe Standardwert eingetragen

wurde oder ein zu restriktiver Filter bei der Datenextraktion angewendet wurde.

Auch eine **Überrepräsentation eines Wertes** kann vorliegen, wenn ein Wert so dominant ist, dass die effektive Diversität gering bleibt. Solche Fälle werden besser durch **Verteilungsscore** erfasst.

8.3.3 Praktische Anwendungsszenarien

Die Analyse der Diversität ist besonders nützlich für:

- **Feature Engineering** und Selektion: Identifikation von konstanten oder quasi-konstanten Merkmalen, die oft wenig prädiktiven Wert haben und entfernt werden können, um die Modellkomplexität zu reduzieren und die Trainingszeit zu verkürzen.
- **Datenbereinigung**: Aufdeckung von potenziellen Problemen, wie z.B. ein Feld, das immer den Default-Wert enthält, oder ein Freitextfeld, das fälschlicherweise als kategoriales Feld mit geringer Diversität interpretiert wird.
- **Verständnis der Datenstruktur**: Ein besseres Gefühl dafür bekommen, welche Arten von Informationen in den einzelnen Merkmalen stecken.

Bei Freitextfeldern ist die reine Zählung einzigartiger Werte oft nicht aussagekräftig. Hier sind Techniken aus dem NLP (z.B. Tokenisierung, Stemming) nötig, um die semantische Diversität zu erfassen.

To Do Diversitätsprüfung

1. Berechne den Diversitätsscore für alle kategorialen und numerischen Merkmale im Datensatz.
2. Identifiziere Merkmale mit $S_{\text{distinctness}} < 0,01$ (oder einem anderen anwendungsspezifischen Schwellenwert).
3. Untersuche diese Merkmale genauer:
   - Ist die geringe Diversität erwartet (z.B. Geschlecht, Status-Flag)?
   - Handelt es sich um eine Quasi-Konstante, die entfernt werden könnte?
   - Liegt möglicherweise ein Fehler in der Datenerfassung vor?
4. Dokumentiere die Ergebnisse und getroffenen Entscheidungen.

8.4 Klumpenbildung (Clumpiness)

Die Klumpenbildung eines Merkmals beschreibt, wie stark sich die beobachteten Werte auf wenige dominante Ausprägungen konzentrieren. Ein hoher Grad an Klumpenbildung ist häufig ein Indikator für ungleiche Datenverteilungen und kann problematisch sein, wenn eine ausgewogene Verteilung erwartet wird, z.B. für statistische Analysen oder das Training von Machine-Learning-Modellen.

8.4.1 Definition und Berechnung

Der **Klumpen-Score** $S_{\text{clumpiness}}$ ist definiert als 1 minus den Anteil der n häufigsten Werte an der Gesamtanzahl der Beobachtungen: $S_{\text{clumpiness}} = 1 - \frac{C_{top\,n}}{N}$

wobei $C_{top,n}$ die Summe der Häufigkeiten der n häufigsten Werte und N die Gesamtzahl der gültigen (nicht fehlenden) Werte ist.

Beispiel: Klumpenbildung

Ein Datensatz enthält 1000 Werte eines Merkmals:

Merkmal "Bundesland“: Die fünf häufigsten Bundesländer machen 700 von 1.000 Einträgen aus.
$C_{top\,5} = 700$, $N = 1.000$
$S_{\text{clumpiness}} = 1 - \frac{700}{1.000} = 0,3$
Hier liegt eine deutliche Klumpenbildung vor.

Merkmal "Produkt-ID“: Jeder Wert kommt ungefähr gleich oft vor, die Top 5 machen zusammen nur 7 Einträge aus. $C_{top,5} = 7$, $N = 1.000$ $S_{\text{clumpiness}} = 1 - \frac{7}{1.000} = 0,993$ Es besteht praktisch keine Klumpenbildung.

Merkmal “Status-Flag“: “Aktiv“ kommt 950-mal, “Inaktiv“ 50-mal vor. $C_{top,2} = 1.000$, $N = 1.000$ $S_{\text{clumpiness}} = 1 - \frac{1.000}{1.000} = 0,0$

Maximale Klumpenbildung, alle Werte entfallen auf nur zwei Ausprägungen.

Merke:

Muss-Felder bergen oft größere Probleme als **Kann-Felder**. Da Datensätze in der Regel erst gespeichert werden können, wenn alle Muss-Felder ausgefüllt sind, besteht die Gefahr, dass diese Felder mit **fiktiven Daten** wie “123“ oder Platzhaltern befüllt werden, um die Datenerfassung abzuschließen. Dies kann die Datenqualität erheblich beeinträchtigen. Diese bilden gegebenenfalls Klumpen oder können durch Datenhistogramme oder statitsche Test (z.B. Benford-Test) identifiziert werden.

8.4.2 Interpretation und Fallstricke

Ein **niedriger Klumpen-Score** (nahe 0) zeigt an, dass wenige Werte das Merkmal dominieren. Dies kann akzeptabel sein, beispielsweise bei seltenen Ereignissen oder einer legitimen Klassenverteilung. Allerdings kann es auch auf **Verzerrungen** oder **Datenprobleme** hinweisen, wie Vorbelegungen,

systematische Fehler oder ungleiche Stichproben.

Ein **hoher Klumpen-Score** (nahe 1) bedeutet, dass die Werte breit gestreut sind und keine Ausprägung dominiert. Dies ist wünschenswert für Merkmale, bei denen eine ausgeglichene Verteilung erwartet wird, etwa bei Trainingsdaten für **Klassifikatoren**.

Die Wahl von n beeinflusst die **Sensitivität** des Scores. Für binäre Merkmale ist $n = 1$ geeignet, während für kategoriale Felder mit mehreren Ausprägungen $n = 5$ oder $n = 10$ gewählt werden kann.

Bei **numerischen Merkmalen** mit sehr vielen unterschiedlichen Werten ist der Score oft hoch und wenig aussagekräftig. In solchen Fällen kann eine **Binning-Strategie** sinnvoll sein, um die Interpretierbarkeit zu verbessern.

Eine **Binning-Strategie** teilt kontinuierliche numerische Daten in diskrete Intervalle („Bins"), um die Analyse zu vereinfachen und Muster sichtbar zu machen. Sie reduziert die Komplexität, indem ähnliche Werte zusammengefasst werden, was besonders bei Merkmalen mit vielen einzigartigen Werten, wie beim **Klumpen-Score**, die Interpretierbarkeit verbessert. Methoden wie gleichbreites oder gleichfrequentes Binning sowie benutzerdefinierte oder statistische Ansätze werden je nach Datenverteilung gewählt.

8.4.3 Praktische Anwendungsszenarien

Die Analyse der Klumpenbildung ist nützlich für:

- **Klassifikationsaufgaben**: Erkennen von Klassendominanz und potenziellen Problemen bei unausgewogenen Datensätzen (Imbalance).
- **Feature Engineering**: Identifikation von Feldern mit geringer Varianz, die als Prädiktor wenig beitragen oder das Modell verzerren können.
- **Sampling und Bias-Erkennung**: Feststellen, ob eine Datenquelle einzelne Ausprägungen überproportional oft enthält (z.B. durch technische oder organisatorische Effekte).

To Do Klumpenbildungsanalyse

1. Berechne für alle Merkmale den Klumpen-Score mit geeignetem n (z.B. $n = 5$ für viele kategoriale Merkmale).
2. Identifiziere Merkmale mit besonders niedrigen Scores ($S_{\text{clumpiness}} < 0,2$ als Faustregel).
3. Prüfe, ob die Klumpenbildung sachlich begründet ist oder auf Datenprobleme hindeutet.
4. Dokumentiere auffällige Merkmale und triff ggf. Maßnahmen (z.B. Rebalancing, Entfernen als Feature).

8.5 Verteilung (Entropie)

Während der Diversitätsscore die Anzahl unterschiedlicher Werte betrachtet, analysiert die Verteilung, *wie* diese Werte über die Datensätze verteilt sind. Sind sie gleichmäßig verteilt, oder dominieren bestimmte Werte das Geschehen? Maße wie die Entropie oder der Gini-Index helfen, diese Aspekte zu quantifizieren.

8.5.1 Konzept der Datenverteilung

Die **Datenverteilung** eines Merkmals beschreibt die Häufigkeit, mit der jeder einzelne Wert (oder Wertebereich bei kontinuierlichen Daten) in dem Merkmal auftritt. Die Analyse der Verteilung hilft zu verstehen, ob Werte konzentriert oder gestreut sind.

Für kategoriale Merkmale ist die Verteilung einfach die Auflistung der Häufigkeiten jeder Kategorie. Eine stark ungleiche Verteilung, bei der ein oder wenige Werte sehr häufig und andere sehr selten sind, kann für bestimmte Analysemethoden problematisch sein (z.B. unausgewogene Klassen bei Klassifikationsaufgaben).

Bei kontinuierlichen Merkmalen wird die Verteilung oft durch Histogramme oder Dichtefunktionen visualisiert und durch Maße wie Mittelwert, Median, Varianz, Schiefe (Skewness) und Kurtosis beschrieben.

8.5.2 Entropie als Maß für die Gleichverteilung

Die Shannon-Entropie ist ein Konzept aus der Informationstheorie und kann verwendet werden, um die Ungewissheit oder "Überraschung" in einer Verteilung zu messen. Eine hohe Entropie entspricht einer gleichmäßigeren Verteilung der Werte, während eine niedrige Entropie auf eine konzentrierte Verteilung (Dominanz weniger Werte) hindeutet.

Die Shannon-Entropie wurde 1948 vom US-Amerikaner Claude Shannon in seiner Arbeit "A Mathematical Theory of Communication" vorgestellt. Sie bildet die Grundlage der Informationstheorie.

Für ein diskretes Merkmal X mit k möglichen Werten $x_1, \ldots, x_k$, wobei $P(x_i)$ die Wahrscheinlichkeit (relative Häufigkeit) des Auftretens von Wert x_i ist, ist die **Shannon-Entropie** $H(X)$ definiert als: $H(X) = -\sum_{i=1}^{k} P(x_i) \log_2(P(x_i))$

Wobei $P(x_i) \log_2(P(x_i))$ als 0 definiert ist, wenn $P(x_i) = 0$.

Die Entropie ist maximal, wenn alle Werte gleich wahrscheinlich sind ($P(x_i) = 1/k$ für alle i). In diesem Fall ist die maximale Entropie $H_{max}(X) = \log_2(k)$.

Die Entropie ist minimal (gleich 0), wenn nur ein Wert mit Wahrscheinlichkeit 1 auftritt (das Merkmal ist eine Konstante).

Der Score für die Verteilungsgleichheit $S_{\text{distribution}}$ basierend auf der normalisierten Entropie kann wie folgt berechnet werden: $S_{\text{distribution}} = \frac{H(X)}{H_{max}(X)} = \frac{H(X)}{\log_2(U)}$

wobei U die Anzahl der einzigartigen Werte im Merkmal ist (wie in der Diversitätssektion, $U = k$). Wenn $U = 1$ (konstantes Merkmal), ist $\log_2(U) = 0$. In diesem Fall wird $S_{\text{distribution}}$ oft als 0 definiert, oder die Entropie selbst wird als 0 betrachtet, was eine minimale Verteilungsvielfalt anzeigt.

Der Gini-Index ist eine alternative Metrik zur Messung der Ungleichheit einer Verteilung, oft verwendet in der Ökonomie (Lorenzkurve) und bei Entscheidungsbäumen.

Der Score liegt im Bereich $[0; 1]$. Ein Wert nahe 1 deutet auf eine hohe Gleichverteilung hin, ein Wert nahe 0 auf eine starke Konzentration.

Beispiel: Entropie-Score für ein kategoriales Merkmal

Ein Merkmal "Farbe“ hat 100 Einträge und folgende Verteilung:

- Rot: 50 Mal ($P(Rot) = 0,5$)
- Grün: 25 Mal ($P(Grn) = 0,25$)
- Blau: 25 Mal ($P(Blau) = 0,25$)

Anzahl einzigartiger Werte $U = 3$. Maximale Entropie $H_{max} = \log_2(3) \approx 1.585$ Bits.

Berechnete Entropie $H(Farbe)$:

$$
\begin{aligned}
&= -[(0,5 \log_2 0,5) + (0,25 \log_2 0,25) + (0,25 \log_2 0,25)] \\
&= -[(0,5 \times -1) + (0,25 \times -2) + (0,25 \times -2)] \\
&= -[-0,5 - 0,5 - 0,5] \\
&= -[-1,5] = 1,5 \text{ Bits}
\end{aligned}
$$

Verteilungsscore $S_{\text{distribution}} = \frac{1,5}{1.585} \approx 0,946$. Dieser Wert ist relativ hoch, aber nicht 1, was die leichte Dominanz von "Rot“ widerspiegelt. Wären alle drei Farben gleich häufig (z.B. 33, 33, 34 Mal), wäre der Score näher an 1. Wäre eine Farbe extrem dominant (z.B. Rot: 98, Grün: 1, Blau: 1), wäre der Score deutlich niedriger.

Bits sind die Grundeinheit für Information. Ein Bit steht für eine Entscheidung zwischen zwei Zuständen (z.B. 0 oder 1). Die Entropie in Bits gibt an, wie viel Information durchschnittlich nötig ist, um ein Ereignis zu beschreiben. Beispiel: Beim Münzwurf (50/50) beträgt die Entropie 1 Bit – es genügt 1 Bit, um das Ergebnis zu kodieren.

8.5.3 Interpretation des Verteilungsscores

- **Hoher Score (nahe 1):** Die verschiedenen Werte des Merkmals kommen ungefähr gleich häufig vor. Dies

kann für manche Analyseverfahren vorteilhaft sein (z.B. Vermeidung von Bias durch dominante Klassen).

- **Niedriger Score (nahe 0):** Ein oder wenige Werte dominieren das Merkmal stark. Dies kann auf eine Quasi-Konstante hindeuten oder auf eine "Long-Tail"-Verteilung, bei der viele Werte sehr selten sind.

Die Interpretation, ob eine hohe oder niedrige Gleichverteilung "gut" oder "schlecht" ist, hängt wiederum vom Kontext ab. Manchmal ist eine ungleiche Verteilung natürlich und erwartet (z.B. die Verteilung von Krankheiten in einer Population).

Für kontinuierliche Merkmale können statt der Entropie auch Maße wie Schiefe (Skewness) und Kurtosis herangezogen werden, um die Form der Verteilung zu charakterisieren.

8.5.4 Anwendungsbeispiele und Herausforderungen

Die Analyse der Werteverteilung ist relevant für:

- **Modellierung**: Erkennung von unausgewogenen Klassen (imbalanced classes) in Klassifikationsaufgaben. Spezielle Techniken (Oversampling, Undersampling, SMOTE) können erforderlich sein.
- **Datensegmentierung**: Identifikation von natürlich vorkommenden Clustern oder Segmenten basierend auf der Häufigkeit von Werten.
- **Anomalieerkennung**: Sehr seltene Werte (Ausreißer in der Verteilung) können auf Anomalien oder Datenfehler hindeuten.

To Do Untersuchung dominanter Werte

1. Berechne den Entropie-basierten Verteilungsscore für alle relevanten kategorialen Merkmale.
2. Identifiziere Merkmale mit einem sehr niedrigen Verteilungsscore (z.B. < 0,2).
3. Für diese Merkmale:
   - Liste die häufigsten Werte und ihre prozentualen Anteile auf.
   - Prüfe, ob die Dominanz eines Wertes fachlich plausibel ist oder auf ein Datenproblem (z.B. Standardwerte, Erfassungsfehler) hindeutet.
   - Erwäge, ob dominante Werte für bestimmte Analysen zusammengefasst oder speziell behandelt werden müssen.

8.6 Plausibilität (Validity)

Die Plausibilität oder Gültigkeit von Datenwerten ist ein Eckpfeiler der Datenqualität. Sie stellt sicher, dass die Daten formalen Regeln, Wertebereichen und Formaten entsprechen, die für das jeweilige Merkmal definiert wurden. Eine detaillierte Einführung in die Validität als Qualitätsdimension findet sich in Abschnitt 3.2.2.

8.6.1 Definition und Bedeutung

Plausibilität (Validity) prüft die Konformität zu Regeln. Korrektheit (Accuracy) prüft die Übereinstimmung mit der Realität, was oft schwerer messbar ist.

Ungültige Daten können zu schwerwiegenden Fehlern in Analysen, Berichten und operativen Prozessen führen. Beispielsweise kann eine Postleitzahl mit Buchstaben oder ein negatives Alter zu falschen Schlussfolgerungen oder Systemabstürzen führen. Die Sicherstellung der Plausibilität ist daher ein fundamentaler Schritt in der Datenaufbereitung.

8.6.2 Typen von Plausibilitätsregeln

Plausibilitätsregeln können vielfältig sein und hängen stark vom spezifischen Merkmal und dessen Bedeutung ab. Gängige Typen sind:

- **Wertebereichsprüfungen**: Überprüfung, ob numerische Werte innerhalb eines erlaubten Minimums und Maximums liegen (z.B. Alter zwischen 0 und 120 Jahren; Score zwischen 0 und 100).
- **Formatprüfungen**: Sicherstellung, dass Werte einem bestimmten Format entsprechen (z.B. Datumsformate wie YYYY-MM-DD, E-Mail-Adressformate).
- **Mustervalidierung (Regular Expressions)**: Prüfung, ob Zeichenketten einem bestimmten Muster entsprechen (z.B. Postleitzahlen, Telefonnummern, Artikelnummern).
- **Typkonsistenz**: Sicherstellung, dass der Datentyp des Wertes dem erwarteten Datentyp des Merkmals entspricht (z.B. keine Buchstaben in einem numerischen Feld).
- **Zulässige Werte (Set Membership)**: Überprüfung, ob kategoriale Werte aus einer vordefinierten Liste zulässiger Ausprägungen stammen (z.B. Länderkürzel gemäß

ISO-Standard, Produktstatus aus offen', in Bearbeitung', 'geschlossen').

- **Prüfziffernverfahren**: Validierung von Identifikationsnummern (z.B. ISBN, Kreditkartennummern) mittels eingebauter Prüfziffern.

8.6.3 Berechnung des Plausibilitätsscores

Der Plausibilitätsscore S_{validity} für ein Merkmal wird typischerweise als der Anteil der Werte berechnet, die alle definierten Plausibilitätsregeln für dieses Merkmal erfüllen.

Sei N die Gesamtzahl der nicht-fehlenden Werte im Merkmal. Sei N_{invalid} die Anzahl der Werte, die mindestens eine Plausibilitätsregel verletzen.

Der Anteil ungültiger Werte P_{invalid} ist: $P_{\text{invalid}} = \frac{N_{\text{invalid}}}{N}$

Der Score für die Plausibilität S_{validity} ist dann: $S_{\text{validity}} = 1 - P_{\text{invalid}} = \frac{N - N_{\text{invalid}}}{N}$.

Dieser Score liegt im Bereich $[0; 1]$, wobei 1 bedeutet, dass alle (nicht-fehlenden) Werte gültig sind, und 0, dass kein einziger Wert gültig ist.

Domänenwissen ist entscheidend für die Definition sinnvoller Plausibilitätsregeln. Ohne dieses Wissen können wichtige Fehler übersehen oder harmlose Abweichungen fälschlich als Fehler markiert werden.

Beispiel: Validierung von Postleitzahlen (PLZ)

Ein Merkmal "PLZ“ in einem deutschen Adressdatensatz soll 5-stellige numerische Werte enthalten. Es gibt 1.000 Einträge.

- Regel 1: Muss 5 Zeichen lang sein.
- Regel 2: Muss nur aus Ziffern bestehen.

Analyse der Daten:

- 950 Einträge sind korrekte 5-stellige Zahlen (z.B. "10115“).
- 20 Einträge sind 4-stellig (z.B. "1234“).
- 15 Einträge enthalten Buchstaben (z.B. "D-80333“).
- 10 Einträge sind 6-stellig (z.B. "123456“).
- 5 Einträge sind leer (werden hier als fehlend betrachtet und nicht in die N für Validität einbezogen, da sie bereits beim Füllgrad berücksichtigt wurden).

Somit ist $N = 1.000 - 5 = 995$. Anzahl ungültiger Werte $N_{\text{invalid}} = 20(\text{Länge}) + 15(\text{Zeichen}) + 10(\text{Länge}) = 45$. $S_{\text{validity}} = 1 - \frac{45}{995} \approx 1 - 0,0452 = 0,9548$. Der Plausibilitätsscore für das Merkmal "PLZ“ ist ca. 0,955.

8.6.4 Umgang mit nicht-plausiblen Werten

Werden nicht-plausible Werte identifiziert, müssen Strategien zu deren Behandlung entwickelt werden:

- **Fehleranalyse**: Untersuchung der Ursachen für die ungültigen Werte. Handelt es sich um Tippfehler, Systemfehler, veraltete Regeln?
- **Korrektur**: Wenn möglich, sollten die fehlerhaften Werte korrigiert werden (manuell oder automatisiert).
- **Zurückweisung**: In manchen Systemen werden ungültige Daten direkt bei der Erfassung zurückgewiesen.
- **Markierung/Ausnahmebehandlung**: Werte können als ungültig markiert und von bestimmten Analysen ausgeschlossen oder gesondert behandelt werden.
- **Regelanpassung**: Manchmal stellt sich heraus, dass die Plausibilitätsregeln zu streng oder veraltet sind und angepasst werden müssen.

To Do Fehleranalyse und -korrektur

1. Definiere für jedes kritische Merkmal im Datensatz klare Plausibilitätsregeln.
2. Implementiere Skripte zur automatischen Überprüfung dieser Regeln.
3. Erstelle einen Bericht über die Anzahl und Art der Plausibilitätsverletzungen pro Merkmal.
4. Initiiere einen Prozess zur Analyse der häufigsten Fehlerursachen und zur Entwicklung von Korrekturstrategien. Priorisiere dabei Merkmale mit hohem Fehleranteil oder hoher Wichtigkeit.

8.7 Cross-field Consistency

Neben der Betrachtung einzelner Merkmale ist es oft entscheidend, die Beziehungen und Abhängigkeiten zwischen verschiedenen Merkmalen innerhalb desselben Datensatzes (derselben Zeile) zu prüfen. Dies wird als feldübergreifende Konsistenz bezeichnet. Eine detaillierte Einführung in die Konsistenz als Qualitätsdimension findet sich in Abschnitt 3.1.3.

8.7.1 Grundlagen der feldübergreifenden Konsistenz

Inkonsistenzen zwischen Feldern können auf logische Fehler in den Daten hinweisen, die zu falschen Schlussfolgerungen oder fehlerhaften Prozessabläufen führen. Beispielsweise wenn das Geburtsdatum nach dem Einstellungsdatum liegt oder der Status eines Auftrags "versendet“ ist, obwohl das Versanddatum fehlt.

Konsistenzprüfungen erfordern oft ein tiefes Verständnis der Geschäftslogik und der Semantik der Daten. Sie sind meist komplexer zu definieren als Prüfungen einzelner Merkmale.

8.7.2 Beispiele für Konsistenzprüfungen

Die Art der Konsistenzregeln ist sehr vielfältig und domänenspezifisch. Typische Beispiele umfassen:

- **Zeitliche Abhängigkeiten**:
  - Geburtsdatum<Einstellungsdatum
  - Bestelldatum≤Versanddatum
  - Kreditauszahlungsdatum<Kreditende_Datum
- **Summen- und Teilprüfungen**:
  - Gesamtbetrag=Nettobetrag+Mehrwertsteuerbetrag
  - Anzahl_Produkte_im_Warenkorb = SUMME (AnzahlproWarenkorbposition)
- **Abhängigkeiten zwischen kategorialen Werten**:
  - Wenn Land = "Deutschland“, dann muss Währung = ËUR“.
  - Wenn Familienstand = "ledig“, dann darf Hochzeitsdatum nicht gefüllt sein.
  - Wenn Auftragsstatus = ßtorniert“, dann sollte Rechnungsnummer leer sein.
- **Bedingte Wertebereichsprüfungen**:
  - Wenn Kundentyp = "Privat“, dann Rabatt ≤ 10
  - Wenn Kundentyp = "Geschäftlich“, dann Rabatt‘ ≤ 30

Beispiel: Konsistenzprüfung Kreditauszahlung vs. Kreditende

In einem Datensatz über Kredite gibt es die Felder Auszahlungs-

datum (Feld10) und Vertragsende_Datum (Feld11). Eine Konsistenzregel besagt: Auszahlungsdatum muss vor oder am selben Tag wie Vertragsende_Datum liegen. Ein Datensatz mit Auszahlungsdatum = 2025-05-10 und Vertragsende_Datum = 2025-03-15 wäre inkonsistent.

8.7.3 Quantifizierung und Herausforderungen

Die Messung der Konsistenz kann auf verschiedene Arten erfolgen:

- **Anteil konsistenter Datensätze**: Man zählt die Anzahl der Datensätze (Zeilen), die alle definierten Konsistenzregeln erfüllen, und teilt diese durch die Gesamtzahl der relevanten Datensätze.
  $S_{\text{consis}} = \frac{\text{Anzahl konsistenter Datensätze}}{\text{Gesamtzahl geprüfter Datensätze}}$
- **Regelspezifische Scores**: Für jede einzelne Konsistenzregel kann ein eigener Score berechnet werden (Anteil der Datensätze, die diese spezielle Regel erfüllen).

Die Herausforderungen bei der Konsistenzprüfung sind:

- **Definition der Regeln**: Das Erfassen und Formalisieren aller relevanten Geschäftsregeln kann sehr aufwendig sein und erfordert enge Zusammenarbeit mit Fachexperten.
- **Komplexität der Regeln**: Manche Regeln können sehr komplex sein und mehrere Felder oder sogar Tabellen umfassen.
- **Datenvolumen**: Die Überprüfung aufwendiger Regeln bei großen Datenmengen kann rechenintensiv sein.

8.8 Ausreißer-Belastung (Outlier Score)

Die Analyse von Ausreißern in einzelnen Merkmalen ist ein wichtiger Aspekt der Datenqualitätsbewertung, der bisher in den klassischen Dimensionen des Feature Reliability Score unterrepräsentiert war. Ausreißer können nicht nur die statistische Analyse verzerren, sondern auch auf systematische Probleme in der Datenerfassung oder -verarbeitung hinweisen. Ein dedizierter Outlier-Score erweitert daher den FRS um eine wichtige Qualitätsdimension.

Die Berücksichtigung von Ausreißern im FRS ist besonders wichtig für Machine-Learning-Anwendungen, da einzelne extreme Werte die Modellleistung erheblich beeinträchtigen können.

8.8.1 Definition und Bedeutung des Outlier-Scores

Der **Outlier-Score** S_{outlier} quantifiziert den Anteil der Datenpunkte in einem Merkmal, die als statistische Ausreißer klassifiziert werden. Er berechnet sich als das Komplement des Anteils identifizierter Ausreißer:

$$S_{\text{outlier}} = 1 - \frac{N_{\text{outlier}}}{N}$$

wobei N_{outlier} die Anzahl der als Ausreißer identifizierten Werte und N die Gesamtzahl der gültigen (nicht-fehlenden) Werte ist.

Ein hoher Outlier-Score (nahe 1) bedeutet, dass das Merkmal nur wenige oder keine Ausreißer aufweist und damit eine homogene Werteverteilung besitzt. Ein niedriger Score deutet auf eine hohe Anzahl extremer Werte hin, was verschiedene Probleme signalisieren kann: von Datenerfassungsfehlern über systematische Verzerrungen bis hin zu legitimen, aber seltenen Ereignissen.

Ein Merkmal mit vielen Ausreißern ist nicht zwangsläufig ßchlecht"– es könnte wichtige seltene Ereignisse enthalten. Die Interpretation muss daher immer kontextabhängig erfolgen.

8.8.2 Methoden zur Ausreißererkennung im FRS-Kontext

Für die Berechnung des Outlier-Scores im Rahmen des Feature Reliability Score wird auf Kapitel 9) verwiesen.

8.9 Berechnung eines Gesamt-Scores

Ausgangspunkt sind die 7 oben erwähnten Einzelscores S_i,$i = 1, \ldots 7$ die einen Gesamt-Score – den Feature Reliability Score für ein Datenfeld – bilden sollen.

Der einfachste Ansatz ist der **ungewichtete arithmetische Durchschnitt**, bei dem alle Dimensionen gleichwertig behandelt werden:

$$FRS_{\text{gesamt}} = \frac{1}{7} \sum_{i=1}^{7} S_i$$

Eine robustere Alternative stellt das **geometrische Mittel** dar, welches besonders sensitiv auf niedrige Einzelwerte

reagiert und damit kritische Datenqualitätsprobleme stärker gewichtet:

$$FRS_{\text{gesamt}} = \left(\prod_{i=1}^{7} S_i \right)^{1/7}$$

Für Anwendungen, die eine hohe Zuverlässigkeit in allen Dimensionen erfordern, eignet sich das **harmonische Mittel**, das noch stärker von schlechten Einzelwerten dominiert wird:

$$FRS_{\text{gesamt}} = \frac{7}{\sum_{i=1}^{7} \frac{1}{S_i}}$$

Ein extremer Ansatz ist der **Minimum-basierte Score**, der dem Prinzip des schwächsten Glieds folgt:

$$FRS_{\text{gesamt}} = \min_i \{S_i\}$$

Das **quadratische Mittel** verstärkt höhere Werte und dämpft niedrige:

$$FRS_{\text{gesamt}} = \sqrt{\frac{1}{7} \sum_{i=1}^{7} S_i^2}$$

Ein **gewichtetes geometrisches Mittel** ermöglicht eine flexiblere Gewichtung mit $\sum_{i=1}^{7} w_i = 1$:

$$FRS_{\text{gesamt}} = \prod_{i=1}^{7} S_i^{w_i}$$

Ein **mehrstufiger hierarchischer Ansatz** strukturiert die Bewertung durch eine Gruppierung der Dimensionen:

$$\begin{aligned} FRS_{\text{structure}} &= w_a \cdot S_{\text{completeness}} + w_b \cdot S_{\text{distinctness}} + w_c \cdot S_{\text{clumpiness}} \\ FRS_{\text{content}} &= w_d \cdot S_{\text{distribution}} + w_e \cdot S_{\text{validity}} + w_f \cdot S_{\text{consistency}} \\ FRS_{\text{gesamt}} &= \alpha \cdot FRS_{\text{structure}} + \beta \cdot FRS_{\text{content}} + \gamma \cdot S_{\text{outlier}} \end{aligned}$$

8.9.1 Praxisbeispiel: Kardiodaten

Die nachfolgend verwendeten Daten stammen aus dem Cardiovascular Disease Dataset von *Kaggle* [9]. Der Datensatz besteht aus 70.000 Patientendatensätzen mit 11 Merkmalen plus Zielvariable (34.979 Patienten mit und 35.021 Patienten ohne Herz-Kreislauf-Erkrankungen).

Prompt für Feature Reliability Score

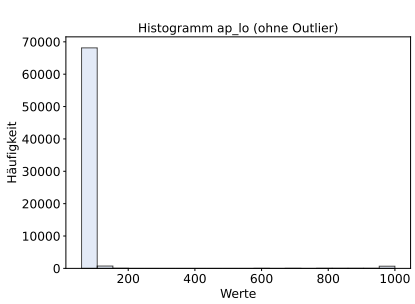
Das Verzeichnis ist *C:\Daten*. Die zu verwendende Datei ist in *C:\Daten\cardio_train.csv* enthält in der ersten Zeile die Feldnamen. Bitte nutze diese Datei. Beachte: Die Trennung ist der Spalten ist mit ";".

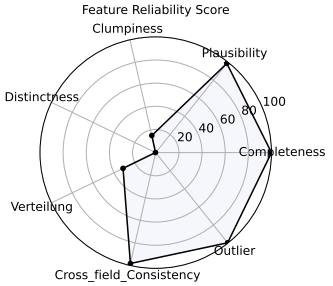
Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'ap_lo'` und `'ap_hi'` für die Analyse aus.
2. Berechne für das Feld `'ap_lo'` einen **Feature Reliability Score**. Dazu sollen folgende Kennzahlen bestimmt und ausgegeben werden (jeweils als *100 auf 2 Stellen gerundet). Bis einschließlich 2. ist die Basis alle Datensätze, ab 3. die um leere Werte bereinigten Daten:
   a) **Anzahl der Datensätze**
   b) **Completeness**: 1 - Anteil der leeren Werte.
   c) **Plausibility**: 1 - Anteil der Werte, die außerhalb von 40 und 200 liegen.
   d) **Clumpiness**: 1 - Anteil der Daten, die auf die drei häufigsten Werte entfallen.
   e) **Distinctness**: Anteil der unterschiedlichen Werte.
   f) **Verteilung**: normalisierte Entropie der Werteverteilung.
   g) **Cross-field Consistency**: Anteil konsistenter Datensätze, bei denen `ap_lo` ≤ `ap_hi` ist.
   h) **Outlier**: 1 - Anteil der Werte oberhalb des 99%-Quantils auf Basis des Z-Scores.
3. Erstelle ein Grafik als Histogramm der Daten mit 20 bins ohne die Outlierwerte
4. Erstelle einen Radar-Chart mit den einzelne Scores
5. Erstelle eine Grafik mit den Scores als Tabelle
6. Erstelle eine Grafik mit Max, Min, Mean, Median, MAD, Standardabweichung, Top 5 Werte als Tabelle - Rundung: 2 Stellen
7. Speichere aus den oben erwähnten Grafiken eine 2x2 Grafik als pdf unter c:\Daten\FRS_ap_lo.pdf.
   Benutze nur den Farbcode #D8E1F4 und schwarz. Als Schriftgröße nimm 16px. Achte darauf, dass die Tabellen zueinander oben ausgerichtet sind und ausreichend Platz dazwischen ist.

Prompt 8.1: Prompt für Feature Reliability Score

Das Ergebnis des Python-Script aus obigen Prompt ist folgende Grafik:





Feature Reliability Scores

Metrik	Wert
Anzahl_Datensätze	70,000
Completeness	100.00%
Plausibility	98.55%
Clumpiness	15.13%
Distinctness	0.22%
Verteilung	31.34%
Cross_field_Consistency	98.24%
Outlier	99.64%

Statistische Kennzahlen

Statistik	Wert
Maximum	11000.00
Minimum	-70.00
Mittelwert	96.63
Median	80.00
MAD	1.00
Standardabweichung	188.47
Top 1 Wert	80.00 (34847x)
Top 2 Wert	90.00 (14316x)
Top 3 Wert	70.00 (10245x)
Top 4 Wert	100.00 (4082x)
Top 5 Wert	60.00 (2727x)

Abbildung 8.1: Auswertung des Feature Reliability Score des Feldes 'ap_lo' (diastolischer Blutdruck) aus der Datei *C:\Daten\cardio_train.csv*

Die 70 000 Datensätze enthalten zwar keine Leerwerte für den diastolischen Blutdruck (Completeness=100.00%), allerdings gibt es zahlreiche unplausible Werte.

Zudem gibt es zum systolischen Blutdruck (sollte immer über dem diastolischen Werten liegen) Unstimmigkeiten (Cross Field Consistency = 98.24%).

Der Outlier Score ist relativ niedrig. Dies liegt am Maskierungseffekt der Outlier auf die Standardabweichung (188.47).

Die Daten sind zudem verklumpt. Nahezu 50 % der Daten fallen auf den Wert 80. Die starke Verklumpung zeigt sich auch am kleinen MAD von 1 (Median der absoulten Abweichungen vom Median).

Die Daten müssen vor einer weiteren Bearbeitung bereinigt werden:

Eine vollständige Bereinigung auf plausible, physiologische Daten aus *C:\Daten\cardio_train.csv* wird mit folgendem Grenzen durchgeführt:

Blutdruck zwischen 20 und 200 (diastolisch) und 40 und 300 (systolisch), Alter zwischen 15 Jahren und 100 Jahren, Größe zwischen 100 cm und 230 cm, Gewicht zwischen 40 kg und 300 kg.

Das lässt noch Ausreißer zu, eliminiert aber Daten die sicher Fehlmessungen sind. Der Prompt dazu ist:

Prompt für Gesamt-Bereinigung

Das Verzeichnis ist *C:\Daten*. Die zu verwendende Datei ist in *C:\Daten\cardio_train.csv* enthält in der ersten Zeile die Feldnamen. Bitte nutze diese Datei. Beachte: Die Trennung ist der Spalten ist mit ";".

Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'age'`, `'height'`, `'weight'`, `'ap_lo'` und `'ap_hi'` für die Analyse aus.
2. Ermittle über das Feld `'ap_lo'` die Datensätze, die folgende Eigenschaft haben:
   a) Das Feld ist leer.
   b) Die Werte liegen außerhalb von 40 und 200.
   c) Es ist `ap_lo` > `ap_hi`.
3. Ermittle über das Feld `'age'` die Datensätze, die folgende Eigenschaft haben:
   a) Das Feld ist leer.
   b) Die Werte liegen außerhalb von 15*365 und 100*365.
4. Ermittle über das Feld `'height'` die Datensätze, die folgende Eigenschaft haben:
   a) Das Feld ist leer.
   b) Die Werte liegen außerhalb von 110 und 230.
5. Ermittle über das Feld `'weight'` die Datensätze, die folgende Eigenschaft haben:
   a) Das Feld ist leer.
   b) Die Werte liegen außerhalb von 40 und 300.
6. Ermittle über das Feld `'ap_hi'` die Datensätze, die folgende Eigenschaft haben:
   a) Das Feld ist leer.
   b) Die Werte liegen außerhalb von 40 und 300.

   Dies sind die Auschlussdatensätze.
7. Erstelle ein neues csv-File *C:\Daten\cardio_train_bereinigt.csv* mit allen Daten ohne die Ausschlussdatensätze
8. Erstelle ein neues csv-File *C:\Daten\cardio_train_ausschluss.csv*, das die Ausschlussdatensätze enthält.

Prompt 8.2: Prompt für Bereinigung Cardiovascular Dataset

Nachfolgende Tabelle zeigt die ermittelten unplausiblen Werte:

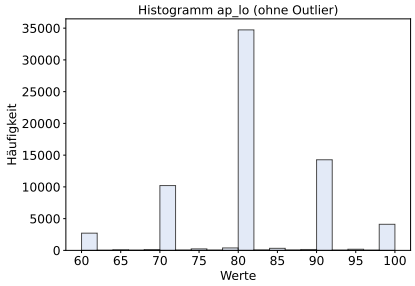
Prüfung	Anzahl Datensätze
ap_lo außerhalb Bereich	991
ap_lo > ap_hi	1234
height außerhalb Bereich	40
weight außerhalb Bereich	52
ap_hi außerhalb Bereich	228
entfernte Datensätze	1392

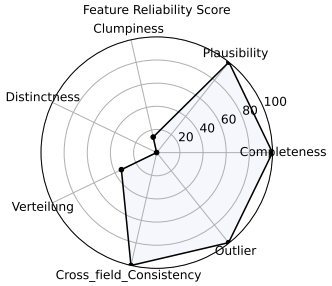
Tabelle 8.1: Ergebnisse der Plausibilitätsprüfungen

Die Anzahl der entfernten Datensätze ist wegen Doppelzählungen geringer als die entsprechende Summenspalte in der Tabelle.

Wir führen mit der neuen Datei *C:\Daten\cardio_train_berei-*

nigt.csv nochmals das gleiche Script zur Feature Reliability Score Generierung aus und erhalten jetzt:





Feature Reliability Scores

Metrik	Wert
Anzahl_Datensätze	68.608
Completeness	100.00%
Plausibility	99.97%
Clumpiness	13.78%
Distinctness	0.12%
Verteilung	33.89%
Cross_field_Consistency	100.00%
Outlier	99.51%

Statistische Kennzahlen

Statistik	Wert
Maximum	182.00
Minimum	20.00
Mittelwert	81.30
Median	80.00
MAD	0.00
Standardabweichung	9.50
Top 1 Wert	80.00 (34706x)
Top 2 Wert	90.00 (14252x)
Top 3 Wert	70.00 (10196x)
Top 4 Wert	100.00 (4069x)
Top 5 Wert	60.00 (2705x)

Abbildung 8.2: Auswertung des Feature Reliability Score nach Bereinigung des Feldes 'ap_lo' (diastolischer Blutdruck). Es wurden 1392 Daten entfernt und in der Datei cardio_train_ausschluss.csv archiviert. Das Histogramm zeigt, dass die Messwerte zumeist nur in 10-Stufen vorliegen. Aufgrund seiner Häufungen ist das Feld eher eine kategorisches Feld.

8.10 Zusammenfassung

Dieses Kapitel hat den Feature Reliability Score (FRS) als ein umfassendes Maß zur Bewertung der Verlässlichkeit von Datenmerkmalen vorgestellt. Es wurde dargelegt, dass der FRS typischerweise aus der Bewertung verschiedener fundamentaler Datenqualitätsdimensionen zusammengesetzt wird.

Die wichtigsten Erkenntnisse und Lerninhalte umfassen:

- Die Notwendigkeit, die Qualität einzelner Merkmale systematisch zu bewerten, um die Güte von Datenanalysen und datengetriebenen Entscheidungen sicherzustellen.
- Die Definition und Berechnung des **Füllgrads (Completeness)** als Maß für das Vorhandensein von Daten, wobei der Score 1 – (Anteil fehlender Werte) berechnet wird. Ein hoher Füllgrad ist meist wünschenswert.
- Die Bedeutung der **Diversität (Distinctness/Uniqueness)**, die als (Anzahl einzigartiger Werte / Anzahl Zeilen) quantifiziert wird. Ein extrem niedriger Wert kann auf konstante oder quasi-konstante Merkmale hindeuten, deren Informationsgehalt gering ist.

- Die Analyse der **Klumpenbildung (Clumpiness)**, die misst, wie stark sich Werte auf wenige dominante Ausprägungen konzentrieren. Sie hilft bei der Erkennung von Verzerrungen und unausgewogenen Datenverteilungen.
- Die Bewertung der **Werteverteilung**, oft mittels Entropie, wobei der Score (berechnete Entropie / maximal mögliche Entropie) Aufschluss über die Gleichförmigkeit der Werteverteilung gibt. Ungleichverteilungen sind nicht per se schlecht, müssen aber im Kontext betrachtet werden.
- Die Relevanz der **Plausibilität (Validity)**, die den Anteil der Werte misst, die vordefinierten Regeln (Wertebereiche, Formate, etc.) entsprechen. Die Definition aussagekräftiger Regeln erfordert Domänenwissen.
- Die Prüfung der **Konsistenz zu anderen Feldern (Cross-field Consistency)**, die logische Beziehungen zwischen Merkmalen eines Datensatzes validiert. Dies ist oft komplex, aber entscheidend für die Datenintegrität.
- Die Einführung des **Outlier-Scores**, der den Anteil extremer Werte in einem Merkmal quantifiziert. Diese siebte Dimension erweitert den FRS um wichtige Aspekte der Datenhomogenität und Anomalieerkennung.
- Die Methoden zur **Aggregation** der Einzel-Scores zu einem Gesamt-FRS, insbesondere die Verwendung gewichteter Durchschnitte, sowie die Bedeutung einer transparenten Gewichtung und Visualisierung der Ergebnisse.

Die Anwendung dieser Konzepte ermöglicht es Organisationen, die Qualität ihrer Datenmerkmale proaktiv zu managen, Problembereiche zu identifizieren und die Verlässlichkeit ihrer Datenbasis kontinuierlich zu verbessern.

Teil II: Qualitäts- und Outlier-Analyse univariater Daten

9 Univariate Ausreißer-Analyse

Die Identifikation von Ausreißern in univariaten Daten – also bei der Betrachtung jeweils nur einer einzelnen Variable – bildet den ersten und fundamentalen Schritt in der systematischen Anomalieerkennung. Obwohl diese Methoden auf den ersten Blick einfach erscheinen mögen, sind sie in der Praxis von enormer Bedeutung, da sie oft bereits einen Großteil der Datenqualitätsprobleme aufdecken können.

Obwohl einfach, ist die univariate Analyse ein unverzichtbarer erster Schritt vor komplexeren Analysen. Datenanalytiker verbringen typischerweise 80% ihrer Zeit mit Datenaufbereitung und -bereinigung, während nur 20% für die eigentliche Analyse verbleibt ([10]).

Ausreißer in einzelnen Variablen können vielfältige Ursachen haben: von einfachen Mess- oder Eingabefehlern über besondere, aber korrekte Ereignisse bis hin zu systematischen Problemen im Datenerfassungsprozess. Ihre frühzeitige Erkennung ist entscheidend, da sie nicht nur die Datenqualität gefährden, sondern auch nachgelagerte multivariate Analysen beeinträchtigen können. Ein einzelner extremer Wert kann beispielsweise Korrelationsanalysen verzerren oder Machine-Learning-Modelle in die Irre führen.

Univariat bedeutet, dass jeweils nur eine einzelne Variable oder ein einzelnes Merkmal betrachtet wird, im Gegensatz zu **multivariat**, wo die Beziehungen zwischen mehreren Variablen gleichzeitig analysiert werden.

Dieses Kapitel systematisiert die wichtigsten Verfahren zur univariaten Ausreißererkennung und behandelt sowohl klassische statistische Ansätze als auch moderne, robuste Methoden. Dabei wird besonderer Wert auf die praktische Anwendbarkeit und die Grenzen der verschiedenen Verfahren gelegt. Im Anhang A sind zudem die wichtigsten statistischen Grundlagen zusammengefasst, die für das Verständnis notwendig sind.

9.1 Grundlegende regelbasierte Methoden

Regelbasierte Methoden definieren explizite, mathematische Kriterien zur Klassifikation von Datenpunkten als Ausreißer. Sie haben den Vorteil der Objektivität und Reproduzierbarkeit, sind aber gleichzeitig von den getroffenen Annahmen über die Datenverteilung abhängig.

So unterstellt man beispielsweise beim Z-Score, dass die Daten annähernd normalverteilt sind. Bei schiefen oder multimodalen Verteilungen kann dies zu falschen Klassifikationen führen.

9.1.1 Die Z-Score-Methode

Die Z-Score-Methode, auch Standardisierung genannt, misst die Abweichung eines Datenpunktes vom Mittelwert in Ein-

heiten der Standardabweichung. Sie basiert auf der Annahme, dass die Daten annähernd normalverteilt sind.

Diese Transformation entspricht dem StandardScaler in scikit-learn und ist essentiell für viele Machine Learning-Algorithmen (z.B. SVM, Neuronale Netze, k-NN), da sie unterschiedlich skalierte Features vergleichbar macht.

Der **Z-Score** eines Datenpunktes x_i wird berechnet als:

$$z_i = \frac{x_i - \bar{x}}{s} \tag{9.1}$$

wobei $\bar{x}$ der Mittelwert und s die Standardabweichung der Daten ist.

Ein Z-Score von 0 bedeutet, der Datenpunkt entspricht exakt dem Mittelwert. Ein Z-Score von +2 bedeutet, der Punkt liegt zwei Standardabweichungen über dem Mittelwert. Eine gängige Faustregel klassifiziert alle Datenpunkte, deren absoluter Z-Score größer als 3 ist ($|z| > 3$), als Ausreißer.

Beispiel: Anwendung des Z-Scores

Gegeben sei der Datensatz von Testergebnissen:
$\{78, 82, 85, 88, 90, 91, 92, 95, 98, 125\}$.

1. **Mittelwert:** $\bar{x} = \frac{78+\cdots+125}{10} = 92.4$.
2. **Standardabweichung (Stichprobe):** $s = \sqrt{\frac{\sum(x_i-\bar{x})^2}{n-1}} \approx 12.90$.
3. **Z-Scores berechnen:** $z_i = \frac{x_i-\bar{x}}{s}$. Beispiele:
   $z_{78} \approx \frac{78-92.4}{12.90} \approx -1.12, \quad z_{125} \approx \frac{125-92.4}{12.90} \approx 2.53$.
4. **Ausreißer identifizieren (Faustregel $|z_i| > 3$):** Keiner der Werte überschreitet die Schwelle. Insbesondere ist $|z_{125}| \approx 2.53 < 3$, also *kein* Ausreißer nach der klassischen Z-Score-Regel.

Die Regel $|z| > 3$ leitet sich aus der 3-Sigma-Regel der Normalverteilung ab. Demnach liegen 99.7% aller Werte innerhalb von ±3 Standardabweichungen vom Mittelwert.

Die entscheidende Einschränkung dieser Methode ist ihre mangelnde Robustheit. Da sowohl der Mittelwert ($\bar{x}$) als auch die Standardabweichung (s) selbst stark durch Ausreißer beeinflusst werden, kann die Anwesenheit von Ausreißern die Z-Scores verzerren und die Methode unzuverlässig machen.

Der **Maskierungseffekt** ist ein großes Problem des Z-Scores. Ein extremer Ausreißer bläht den Mittelwert und die Standardabweichung so stark auf, dass sein eigener Z-Score und die Z-Scores anderer, weniger extremer Ausreißer kleiner werden. Dadurch können Ausreißer „maskiert“ und übersehen werden.

Beispiel: Maskierungseffekt beim Z-Score

Gegeben sei ein Datensatz mit Verkaufszahlen:
$\{100, 105, 95, 110, 98, 5000\}$.

Ohne den Ausreißer 5000: $\bar{x} = 101.6$, $s = 6.5$
Der Z-Score für den Wert 110 wäre: $z = (110 - 101.6)/6.5 \approx 1.29$

Mit dem Ausreißer: $\bar{x} = 901.3$, $s = 1993.7$
Der Z-Score für den Wert 110 ist nun: $z = (110 - 901.3)/1993.7 \approx -0.40$.
Der Z-Score für den Ausreißer 5000: $z = (5000 - 901.3)/1993.7 \approx 2.06$.

Der Ausreißer hat seine eigene Erkennung verhindert und normale Werte wie 110 erscheinen nun als unterdurchschnittlich.

9.1.2 Die IQR-Methode (nach Tukey)

Um den Maskierungseffekt zu umgehen und eine Ausreißer-Definition zu haben, die weitgehend Verteilungsunabhängig ist, wurde von Tukey die IQR-Methode eingeführt.

Diese robuste Methode basiert direkt auf dem Konzept des Boxplots (vgl. Anhang A.4.2) und verwendet den Interquartilsabstand (IQR), um Grenzen für „normale“ Daten zu definieren. Sie ist die am häufigsten empfohlene Methode für das erste Screening, da sie, wie der IQR selbst, unempfindlich gegenüber Extremwerten ist.

John W. Tukey führte die **Boxplots** ein und prägte die **IQR-Methode** zur Erkennung von Ausreißern. Tukey war einer der einflussreichsten Statistiker des 20. Jahrhunderts und prägte auch den Begriff „explorative Datenanalyse“. (vgl. [11] - kein schönes Buch, aber eines der ersten Bücher zur Datenanalyse.)

Die Grenzen, oft als „Tukey Fences“ bezeichnet, werden wie folgt berechnet:

$$\text{Untere Grenze} = Q1 - 1.5 \cdot \text{IQR}$$
$$\text{Obere Grenze} = Q3 + 1.5 \cdot \text{IQR}$$

Alle Datenpunkte, die unterhalb der unteren Grenze oder oberhalb der oberen Grenze liegen, werden als potenzielle Ausreißer klassifiziert.

Der Faktor 1.5 ist nicht willkürlich. Für normalverteilte Daten liegen etwa 99.3% der Werte innerhalb der $1.5 \cdot \text{IQR}$ (entspricht $\pm 2.7\sigma$). Ein Faktor von 3.0 wird manchmal verwendet, um nur „extreme“ Ausreißer zu identifizieren (entspricht $\pm 4.7\sigma$ und damit rund 99.9998% der Daten bei Normalverteilungsannahme).

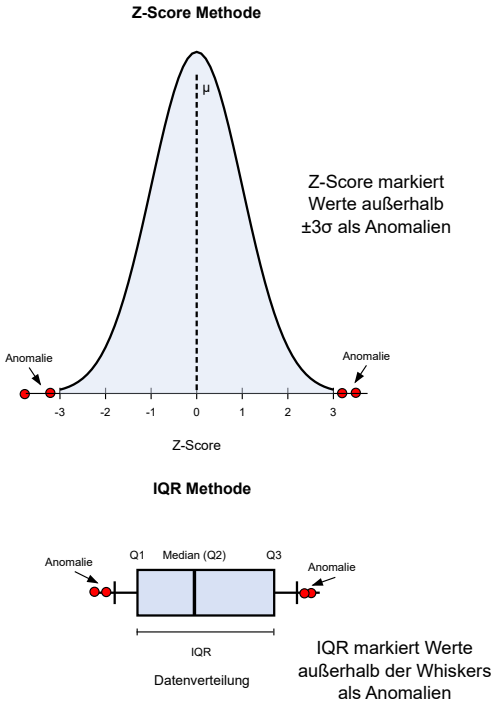


Abbildung 9.1: Outlier-Analyse mit Z-Score und IQR-Methode

Beispiel: Anwendung der IQR-Methode

Gegeben sei der Datensatz von Testergebnissen:
$\{78, 82, 85, 88, 90, 91, 92, 95, 98, 125\}$.

1. **Sortieren:** Die Daten sind bereits sortiert.
2. **Quartile berechnen:** $Q1 = 85$, $Q2(\text{Median}) = \frac{90+91}{2} = 90.5$, $Q3 = 95$.
3. **IQR berechnen:** $\text{IQR} = Q3 - Q1 = 95 - 85 = 10$.
4. **Grenzen berechnen:** Untere Grenze $= 85 - 1.5 \cdot 10 = 70$. Obere Grenze $= 95 + 1.5 \cdot 10 = 110$.
5. **Ausreißer identifizieren:** Der Wert 125 liegt oberhalb der oberen Grenze von 110 und wird daher als Ausreißer markiert. Der Wert 78 liegt innerhalb der Grenzen.

Die **IQR-Methode** erfordert eine Sortierung der Daten ($O(n \log n)$), um die Quartile zu bestimmen. Bei sehr großen Datensätzen ($n \approx 10^6$) ist sie damit etwa 6-mal rechenintensiver als der **Z-Score**, der nur Mittelwert und Standardabweichung benötigt ($O(n)$). Mit der heutigen Rechnerleistung spielt dies aber keinen große Rolle.

9.1.3 Der modifizierte Z-Score

Ein weiteres Maß, das die Probleme des klassischen Z-Scores überwindet, ist der modifizierte Z-Score. Der Vorteil hier ist, dass er bei normalverteilten Daten weitgehend dem klassischem Z-Score entspricht. Ausgangspunkt ist der Median Absolut Deviation (Median der absoluten Abweichungen):

Der **Median Absolute Deviation (MAD)** ist :

$$\text{MAD} = \text{Median}\big(|x_i - \text{Median}(x)|\big). \tag{9.2}$$

Der MAD hat einen **Breakdown Point** von 50% – das bedeutet, er bleibt robust, solange nicht mehr als die Hälfte der Daten Ausreißer sind - und robuster geht es nicht (vgl. [12]). Die Standardabweichung hat einen Breakdown Point von 0% (ein einziger extremer Wert kann sie beliebig verzerren) .

MAD ist ein robuster Streuungsmaßstab, der auf Mediane anstatt Mittelwerten beruht. Er sagt, wie stark die Werte typischerweise vom Median abweichen (= der Median der Abweichungen vom Median). Einzelne extreme Ausreißer haben fast keinen Einfluss auf ihn (im Gegensatz zur Standardabweichung, die durch Ausreißer stark aufgebläht wird).

Beispiel: Berechnung der MAD

Datensatz ohne Ausreißer: $(10, 11, 12, 13, 14)$
Median = 12
Abweichungen: $(2, 1, 0, 1, 2)$
Median der Abweichungen = 1 $\Rightarrow$ **MAD = 1**

Datensatz mit Ausreißer: $(10, 11, 12, 13, 100)$
Median = 12
Abweichungen: $(2, 1, 0, 1, 88)$
Median der Abweichungen = 1 $\Rightarrow$ **MAD = 1**

Trotz des extremen Ausreißers 100 bleibt die MAD klein und robust.

Achtung

Falls mehr als 50% der Daten eines Feldes den gleichen Wert aufweisen, dann ist der MAD 0. Es ist daher wichtig im Vorfeld ein Histogramm zu erstellen.

Praktisch kann man ihn als eine Art „robuste Standardabweichung“ interpretieren, die sich am Median und nicht am Mittelwert orientiert. Mit ihm lässt sich ein modifizierte Z-Score erstellen:

Der **modifizierte Z-Score** verwendet den Median anstelle des Mittelwerts und die Median Absolute Deviation (MAD) anstelle der Standardabweichung:

$$\text{modifizierter Z-Score}_i = \frac{0.6745 \times (x_i - \text{Median})}{\text{MAD}} \tag{9.3}$$

wobei MAD = Median($|x_i - \text{Median}|$) und der Faktor 0.6745 eine Normierungskonstante ist.

Der Faktor 0.6745 entspricht dem 75. Perzentil der Standardnormalverteilung und dem Erwartungswert des MAD bei standardnormalverteilten Daten. Er sorgt damit dafür, dass der modifizierte Z-Score bei normalverteilten Daten ähnliche Werte wie der klassische Z-Score liefert ([13], Seite 12).

Beispiel: Anwendung des modifizierten Z-Scores

Gegeben sei der Datensatz von Testergebnissen:
$\{78, 82, 85, 88, 90, 91, 92, 95, 98, 125\}$.

1. **Median berechnen:** $Q2 = \frac{90+91}{2} = 90.5$.
2. **Abweichungen vom Median:**
   $|x_i - 90.5| = \{12.5, 8.5, 5.5, 2.5, 0.5, 0.5, 1.5, 4.5, 7.5, 34.5\}$.
3. **MAD berechnen:** Median dieser Abweichungen = 5.
4. **Modifizierte Z-Scores berechnen:**

   $$M_i = \frac{0.6745 \times (x_i - 90.5)}{5}.$$

   Beispiele:
   $M_{78} = \frac{0.6745 \times (-12.5)}{5} \approx -1.69$,
   $M_{125} = \frac{0.6745 \times (34.5)}{5} \approx 4.65$.
5. **Ausreißer identifizieren:** Da $|M_{125}| = 4.65 > 3.5$, ist 125 ein Ausreißer. Alle anderen Werte haben $|M_i| < 3.5$ und gelten nicht als Ausreißer.

Der **modifizierte Z-Score** verwendet den Median und die MAD und ist damit robuster gegenüber Ausreißern als der klassische Z-Score. Werte mit Score > 3.5 gelten üblicherweise als Ausreißer ([13]).

Der MAD ist ein äußerst robustes Streuungsmaß.: Er kann bis zu 50% Ausreißer in den Daten tolerieren, ohne selbst verzerrt zu werden. Dadurch bleibt der modifizierte Z-Score auch bei starker Kontamination mit Extremwerten zuverlässig.

Der **Breakdown Point** einer Statistik gibt an, welcher Anteil der Daten beliebig verfälscht sein darf, bevor die Statistik selbst unbrauchbar wird. Für den Median und die MAD liegt dieser bei 50%, für Mittelwert und Standardabweichung bei 0%.

9.1.4 Hampel-Identifier

Ein weiteres robustes Maß um Outlier zu identifizieren ist der Hampel-Identifier. Er ist eine leichte Modifizierung des modifizierten Z-Scores, aber aufgrund seiner Konstruktion leichter verständlich.

Frank Hampel führte die beiden zentralen Begriffe **Einflussfunktion** und **Breakdown Point** ein, mit denen die Robustheitseigenschaften von statistischen Verfahren quantifiziert und optimale Verfahren konstruiert werden können ([14]).

Der Hampel-Identifier benutzt - analog zum modifizierten Z-Score - den Median als Maß der zentralen Tendenz und den MAD als Streuungsmaß:

Der **Hampel-Identifier** klassifiziert einen Datenpunkt x_i als Ausreißer, wenn:

$$|x_i - \text{Median}| > k \times \text{MAD} \tag{9.4}$$

wobei k typischerweise zwischen 2 und 3 gewählt wird.

Der Faktor $k = 3$ entspricht etwa dem 3-Sigma-Kriterium bei normalverteilten Daten, ist aber deutlich robuster.

Während sowohl der Hampel-Identifier als auch der modifizierte Z-Score robuste Verfahren zur Ausreißererkennung darstellen, bietet der Hampel-Identifier Vorteile in der praktischen Anwendung:

Der modifizierte Z-Score erfordert die Erklärung der Normierungskonstante 0.6745 und des Schwellenwerts 3.5, was oft weitere Erklärungen notwendig macht.

Der Hampel-Identifier verwendet die intuitive Formulierung:

> „Ein Wert ist ein Ausreißer, wenn er mehr als k MAD-Einheiten vom Median der Daten entfernt liegt. MAD ist dabei der Median aller absoluten Abweichungen vom Median der Daten."

Diese Erklärung ist für Fachexperten ohne statistische Expertise sofort verständlich, da sie auf dem einfachen Konzept der Distanz basiert.

Beispiel: Anwendung des Hampel-Identifiers

Gegeben sei der Datensatz von Testergebnissen:
$\{78, 82, 85, 88, 90, 91, 92, 95, 98, 125\}$.

1. **Median berechnen:** $Q2 = \frac{90+91}{2} = 90.5$.
2. **Abweichungen vom Median:**
   $|x_i - 90.5| = \{12.5, 8.5, 5.5, 2.5, 0.5, 0.5, 1.5, 4.5, 7.5, 34.5\}$.
3. **MAD berechnen:**
   Sortiert: $\{0.5, 0.5, 1.5, 2.5, 4.5, 5.5, 7.5, 8.5, 12.5, 34.5\}$.
   MAD = 5.
4. **Hampel-Grenzen berechnen:**
   Untere Grenze = $90.5 - 3 \times 5 = 75.5$.
   Obere Grenze = $90.5 + 3 \times 5 = 105.5$.
5. **Ausreißer identifizieren:**
   Der Wert 125 liegt oberhalb der oberen Grenze von 105.5 und ist ein Ausreißer.
   Der Wert 78 liegt innerhalb der Grenzen und ist *kein* Ausreißer.

Der hier beschriebene Hampel-Identifier (vgl. Seite 64, Code X84 in [15] oder als Download [16]; hier ist $k = 5.2$) ist eine gängige MAD-basierte Variante, aber nicht zu verwechseln mit dem **Hampel Filter**. Dieser definiert ein Datenenfenster auf einer Zeitreihe und ersetzt die MAD-Ausreißer jeweils mit ihrem jeweiligen Median.

9.2 Ausreißererkennung bei schiefen Verteilungen

Bei Einkommensdaten ist das Problem der Rechtsschiefe besonders ausgeprägt: Wenige sehr hohe Einkommen führen zu einem "langen Schwanz", während die meisten Werte im unteren Bereich konzentriert sind.

Ein grundlegendes Problem der bisher beschriebenen Ausreißererkennungsmethoden ist ihre symmetrische Natur. Sowohl die IQR-Methode, der Hampel-Identifier als auch der modifizierte Z-Score definieren Ausreißer über gleichmäßige Abstände nach oben und unten. Bei stark schiefen

Verteilungen führt dies zu einer systematischen Verzerrung der Erkennungsleistung.

Rechtsschiefe Verteilungen, wie sie typischerweise bei Einkommensdaten, Verkaufspreisen oder Wartezeiten auftreten, haben eine charakteristische Form mit einem „langen rechten Schwanz“. Die symmetrischen Grenzen der klassischen Verfahren führen dazu, dass:

- Zu viele Werte im rechten Schwanz fälschlicherweise als Ausreißer klassifiziert werden
- Die natürliche Asymmetrie der Datenverteilung nicht berücksichtigt wird
- Legitime hohe Werte (z.B. Führungskräfte-Gehälter) als anomal eingestuft werden

Die **Schiefe** (Skewness) einer Verteilung wird mathematisch definiert als das dritte zentralisierte Moment. Werte < -1 oder > 1 indizieren starke Asymmetrie, die eine angepasste Ausreißererkennung erfordert (vgl. Abschnitt A.3.1).

Wenn die Schiefe der Verteilung einen kritischen Schwellenwert überschreitet (empirisch: Schiefe < -1 oder Schiefe > 1), sollte vor der Ausreißererkennung eine Transformation angewendet werden, um die Verteilung zu symmetrisieren.

Das **Yeo-Johnson-Verfahren** (2000) ist eine stetige Familie von Potenz-Transformationen, die sowohl positive als auch negative Werte verarbeiten kann. Für einen gegebenen Wert x und Parameter λ ist die Transformation definiert als:

$$y = \begin{cases} \frac{(x+1)^\lambda - 1}{\lambda} & \text{falls } x \geq 0, \lambda \neq 0 \\ \ln(x+1) & \text{falls } x \geq 0, \lambda = 0 \\ -\frac{(-x+1)^{2-\lambda} - 1}{2-\lambda} & \text{falls } x < 0, \lambda \neq 2 \\ -\ln(-x+1) & \text{falls } x < 0, \lambda = 2 \end{cases}$$

Der optimale Parameter λ wird durch Maximum-Likelihood-Schätzung (vgl. Kapitel B) bestimmt, um die transformierten Daten möglichst normalverteilt zu machen.

Das Yeo-Johnson-Verfahren erweitert die Box-Cox-Transformation und kann im Gegensatz zu dieser auch mit null- und negativwertigen Daten umgehen, was bei realen Datensätzen häufig vorkommt ([17], Seite 123).

Das Yeo-Johnson-Verfahren hat sich als besonders geeignet erwiesen, da es sowohl positive als auch negative Werte verarbeiten kann und den optimalen Transformationsparameter λ bestimmt.

Workflow für schiefe Daten:

1. Berechne die Schiefe der Daten
2. Falls |Schiefe| > 1: Wende Yeo-Johnson-Transformation an
3. Führe Ausreißererkennung (IQR, Z-Score, Hampel oder mod. Z-Score) auf transformierten Daten durch
4. Transformiere identifizierte Ausreißergrenzen zurück in den Originalmaßstab
5. Berechne Outlier-Score basierend auf den angepassten Grenzen

Dieser Ansatz gewährleistet, dass der Outlier-Score auch bei stark asymmetrischen Datenverteilungen aussagekräftige Ergebnisse liefert und nicht durch die natürliche Schiefe der Daten verzerrt wird.

Die Qualität einer Ausreißererkennung lässt sich nicht pauschal beurteilen, sondern hängt stark vom spezifischen Anwendungskontext ab. Ein Verfahren, das in einem Bereich hervorragende Ergebnisse liefert, kann in einem anderen völlig ungeeignet sein. Daher ist eine systematische Evaluierung unerlässlich, die sowohl die statistischen Eigenschaften der Daten als auch die praktischen Anforderungen des Anwendungsbereichs berücksichtigt.

Hinweis: Outlier-Analyse

In der explorativen Datenanalyse ist es oft sinnvoll, mehrere Methoden parallel anzuwenden und die Ergebnisse zu vergleichen. Punkte, die von mehreren Verfahren als Ausreißer identifiziert werden, verdienen besondere Aufmerksamkeit, während Diskrepanzen zwischen den Methoden Hinweise auf die Datenstruktur geben können.

Dieser "Ensemble"-Ansatz wird auch in der modernen maschinellen Ausreißererkennung erfolgreich eingesetzt.

9.3 Praxisbeispiel: Outlier-Analyse

Das verwendete Datenset in diesem Abschnitt ist unter [18] zu finden. In der Zip Datei sind die Dateien *winequality-red.csv* und *winequality-white.csv*, die jeweils Analysedaten für unterschiedliche Rot- und Weißweine enthalten.

Für die Weißwein-Daten gibt es folgende Werte:

Tabelle 9.1: Physiochemische Datenstatistiken für Weißwein (aus [19])

Variable	Attribut (Einheit)	Min	Max	Mean
fixed acidity	Fixed acidity (g(tartaric acid)/dm^3)	3.8	14.2	6.9
volatile acidity	Volatile acidity (g(acetic acid)/dm^3)	0.1	1.1	0.3
citric acid	Citric acid (g/dm^3)	0.0	1.7	0.3
residual sugar	Residual sugar (g/dm^3)	0.6	65.8	6.4
chlorides	Chlorides (g(sodium chloride)/dm^3)	0.01	0.35	0.05
free sulfur dioxide	Free sulfur dioxide (mg/dm^3)	2	289	35
total sulfur dioxide	Total sulfur dioxide (mg/dm^3)	9	440	138
density	Density (g/cm^3)	0.987	1.039	0.994
pH	pH	2.7	3.8	3.1
sulphates	Sulphates (g(potassium sulphate)/dm^3)	0.2	1.1	0.5
alcohol	Alcohol (vol.%)	8.0	14.2	10.4

Folgender Prompt erstellt ein Python-Programm für ein Histogramm und einen einen Box-Plot mit den Outliern. Wenn die Schiefe betragsmäßig größer 1 ist, wird zusätzlich das Yeo-Johnson-Verfahren angewandt und die zusätzlichen Plots mit den transformierten Daten generiert.

Prompt für automatisierte IQR-Outlier-Analyse

Das Verzeichnis ist *C:\Daten*. In der Datei *winequality-white.csv* sind in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen {Python-Quellcode}, der nach Eingabe eines Feldnamens folgendes macht:

1. Die Schiefe der Feldname-Daten wird berechnet.
2. Berechnet die Outlier auf Basis des Z-Score (Grenzwert 3), IQR-Outlier (mit 1.5xGrenzen), modifizierter Z-Score (Grenzwert 3.5) und den Hampel-Identifer (Grenzwert 3)
3. Erstelle ein Histogramm der Daten im Farbcode #D8E1F4. Markiere den Median (durchgehend), das IQR (gepunktet) und den Erwartungswert (gestrichelt) mit einem senkrechten schwarzen Strich. Zusätzlich die Standardabweichung (als rot gepunktete Linie) und die MAD (als grün gepunktete Linie).
4. Erstelle einen Whiskers (#D8E1F4 und schwarz) und markiere die gefunden Outliers (Z-Score als kleines rotes Kreuz, IQR-Score als kleinen grüne Kreis, modifizierte Z-Score als kleines blaues Dreieck und Hampel-Identifer als kleines schwarzes Quadrat). Versetze die Outlier jeweils ein wenig, damit sie sich nicht vollständig verdecken.
5. Speichere das Histogramm unter histo_Feldname.pdf und BoxPlot_Feldname.pdf in das Verzeichnis.
6. Wenn die Schiefe der Daten betragsmäßig größer 1 ist, wende zusätzlich das Yeo-Johnson-Verfahren für die Outlier-Berechnung an. Speichere dann die zusätzlichen Grafiken mit der Transformation der Daten in histo_Feldname_tranf.pdf und BoxPlot_Feldname_trans.pdf.

Achte darauf, dass alle Outlier sichtbar sind und nicht durch andere Pakete überschrieben werden und die Legende innerhalb der Grafik ist.

Der entsprechende Python-Code wurde mit Claude Sonnet 4 als LLM erstellt.

Anstelle ab 1 kann auch schon ab 0.5 Schiefe eine Transformation sinnvoll sein.

Prompt 9.1: Prompt für IQR Outlier Analyse

In den folgenden Original-Bildern ist die Ausgabe des Pyhton-Scripts mit obigen Prompt dargestellt.

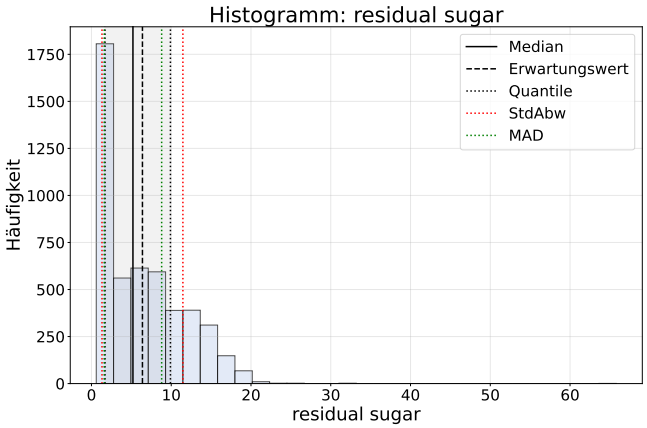


Abbildung 9.2: Histogramm des Feldes "residual sugar". Die Verteilung ist deutlich rechtsschief.

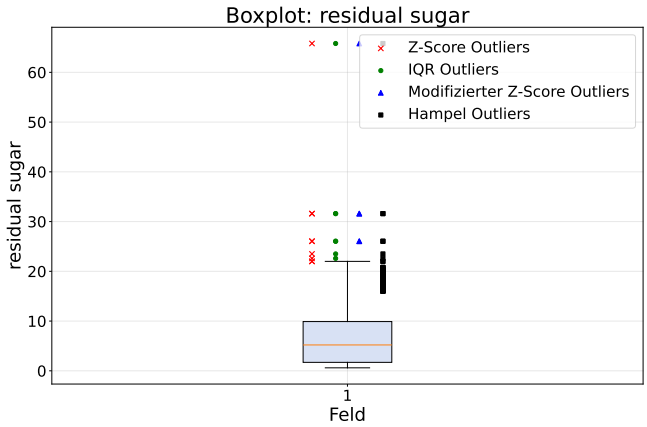


Abbildung 9.3: Outlier und Box-Plot des Feldes "residual sugar". Es gibt zahlreiche Ausreißer.

Die Berechnung der Schiefe zeigt einen Wert von 1.0768. Nach Anwendung der Yeo-Johnson-Transformation sieht das Histogramm folgendermaßen aus:

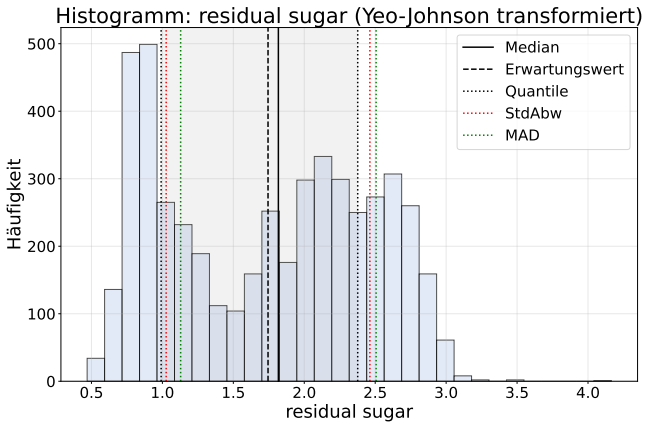


Abbildung 9.4: Histogramm des Feldes "residual sugar" nach der Yeo-Johnson-Transformation. Die Verteilungsschiefe verschwindet.

Die Ausreißer haben sich massiv reduziert, wie nachfolgender BoxPlot zeigt.

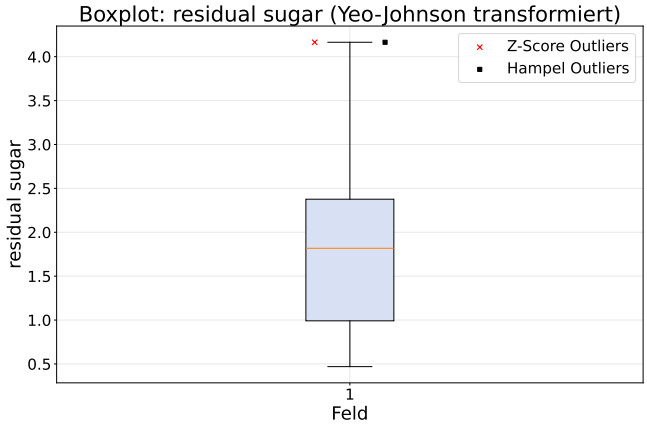


Abbildung 9.5: Outlier und BoxPlot des Feldes "residual sugar" nach der Yeo-Johnson-Transformation. Die Anzahl der Ausreißer haben sich massiv auf einen Ausreißer reduziert.

Wichtig

Es ist sehr wichtig zu Beginn einer Datenanalyse einen Einblick über die Datenstruktur (z.B. über ein Histogramm) zu bekommen.

9.4 Zusammenfassung

Dieses Kapitel hat die fundamentalen Methoden zur univariaten Ausreißererkennung systematisch behandelt und deren praktische Anwendung in der Datenqualitätsanalyse aufgezeigt. Es wurde dargelegt, dass die Wahl der geeigneten Methode stark von den Eigenschaften der zugrundeliegenden Datenverteilung und dem spezifischen Anwendungskontext abhängt.

Ein sehr gutes Buch, das über die Inhalte hier weit hinaus geht, ist das Buch *Feature Engineering and Selection: A Practical Approach for Predictive Models* von Max Kuhn and Kjell Johnson ([17]). Die freizugängliche Online-Version ist in [20].

Umgang mit identifizierten Ausreißern

10

Die bloße Identifikation von Ausreißern markiert lediglich den Beginn eines kritischen Entscheidungsprozesses. Was zunächst als statistisches Problem erscheint, entpuppt sich schnell als komplexe Abwägung zwischen methodischer Korrektheit und fachlicher Expertise.

Die Art und Weise, wie mit Ausreißern umgegangen wird, beeinflusst nicht nur die Validität der statistischen Schlussfolgerungen, sondern kann auch darüber entscheiden, ob wichtige Erkenntnisse gewonnen oder verloren werden.

Vor der Entdeckung des Ozonlochs (1985 in [21]) wurden extrem niedrige Messwerte zunächst als Outliers verworfen, was die Erkenntnis fast verhinderte ([22]).

Die zentrale Herausforderung liegt darin, dass Ausreißer sowohl Störfaktoren als auch Informationsträger sein können. Ein extremer Wert kann einerseits das Ergebnis eines Messfehlers, einer fehlerhaften Dateneingabe oder eines technischen Defekts sein.

Andererseits kann er ein seltenes, aber höchst relevantes Ereignis repräsentieren – etwa einen Durchbruch in der Forschung, einen neuen Markttrend oder ein kritisches Sicherheitsereignis. Diese Dualität erfordert einen systematischen und durchdachten Ansatz, der über rein statistische Überlegungen hinausgeht.

In der Finanzanalyse können extreme Kursbewegungen sowohl auf Datenfehler als auch auf bedeutsame Marktereignisse hinweisen – die falsche Entscheidung kann Millionen kosten.

10.1 Validierung der Ausreißer

Die Validierung identifizierter Ausreißer stellt einen mehrstufigen Prozess dar, der sowohl technische Expertise als auch Domänenwissen erfordert.

Die notwendige Rückverfolgbarkeit von Daten zeigt wie wichtig eine gut dokumentierte Datenbasis ist.

10.1.1 Datenquellen-Überprüfung

Der erste Validierungsschritt führt zurück zu den Ursprüngen der Daten. Diese forensische Herangehensweise erfordert oft die Zusammenarbeit verschiedener Abteilungen und kann zeitaufwändig sein, ist aber unerlässlich für eine fundierte Bewertung.

Berechnete Felder

Insbesondere bei Feldern, die sich aus anderen Feldern berechnen, ist die Fehleranfälligkeit groß (z.B. fehlerhafte Formel, Verwendung von standardisierten Fehlerwerten wie 9999 bei Division mit 0 etc.).

Die Kontrolle der ursprünglichen Datenquellen umfasst die Überprüfung von Logfiles, Sensordaten, Eingabeprotokollen oder manuellen Aufzeichnungen. Hierbei können systematische Probleme aufgedeckt werden, die über den einzelnen

Ausreißer hinausgehen. Beispielsweise könnte ein scheinbar isolierter extremer Temperaturwert auf einen defekten Sensor hinweisen, der auch andere, weniger offensichtliche Messfehler verursacht.

Die Überprüfung von Mess- oder Eingabefehlern erfordert oft technisches Verständnis der verwendeten Instrumente und Prozesse. Häufige Fehlerquellen sind Kalibrierfehler, Einheitenverwechslungen (etwa Celsius vs. Fahrenheit) , Kommafehler bei manueller Eingabe oder Probleme bei der Datentransmission. Diese systematische Fehleranalyse kann wertvolle Verbesserungen der Datenqualität für zukünftige Erhebungen liefern.

Mars Climate Orbiter (NASA, 1999): Die Sonde ging durch einen Umrechnungsfehler verloren – ein Team lieferte Schubdaten in Pfund-Kraft-Sekunden, das Missionssystem erwartete Newton-Sekunden. Ergebnis: falsche Bahnkorrekturen, Eintritt zu tief in die Marsatmosphäre und Totalverlust der Mission. Ein Beispiel für fatale Folgen mangelnder Datenqualität ([23]).

10.1.2 Plausibilitätsprüfung

Die Validierung mit Experten aus dem Fachbereich ist besonders wichtig, da diese die inhaltliche Plausibilität besser beurteilen können als reine Statistiker. Ein Mediziner kann beispielsweise einschätzen, ob ein extremer Blutdruckwert bei einem bestimmten Patienten physiologisch möglich ist, während ein Ingenieur beurteilen kann, ob eine außergewöhnliche Materialfestigkeit unter den gegebenen Testbedingungen erreicht werden kann.

Expert judgement ist subjektiv – dokumentieren Sie daher sowohl die Einschätzung als auch die Begründung und den Namen des Experten.

Der Vergleich mit historischen Daten oder Benchmarks bietet wichtigen Kontext für die Einordnung extremer Werte. Ein Aktienkurs, der um 50% an einem Tag steigt, mag statistisch ein Ausreißer sein, ist aber bei Übernahmeankündigungen durchaus plausibel. Die Herausforderung liegt darin, relevante Vergleichsdaten zu identifizieren und deren Übertragbarkeit auf die aktuelle Situation zu bewerten.

10.1.3 Konsistenzprüfung

Moderne multivariate Ausreißererkennungsverfahren (z.B. Isolation Forest, LOF, Autoencoder) können diese Konsistenzprüfungen automatisieren und dabei subtile Muster erkennen.

Die interne Konsistenz von Daten bietet oft den besten Hinweis auf die Validität von Ausreißern. Diese multidimensionale Analyse (vgl. Teil 13.8 in diesem Buch) kann Muster aufdecken, die bei der isolierten Betrachtung einzelner Variablen übersehen werden.

Der Abgleich mit anderen Variablen desselben Datensatzes kann wichtige Hinweise auf die Plausibilität von Ausreißern liefern. Wenn beispielsweise ein Patient ein ungewöhnlich niedriges Gewicht aufweist, sollten auch andere Parameter wie Größe, Alter und Gesundheitszustand betrachtet werden.

Inkonsistenzen zwischen verschiedenen Variablen können sowohl auf Datenfehler als auch auf interessante medizinische Fälle hinweisen.

10.1.4 Prozedur zur Validierung von Ausreißern

Folgende Prozedur zur Validierung von Ausreißern kann angewandt werden:

To Do: Prozess zur Validierung von Outlier

Entwickeln Sie eine Standardarbeitsanweisung für den Umgang mit Ausreißern in Ihrem Team oder Unternehmen.

1. **Rollen und Verantwortlichkeiten definieren:** Wer ist für die Erkennung, Validierung und Behandlung von Ausreißern zuständig (z.B. Data Analyst, Data Steward, Fachexperte)?
2. **Methodenkatalog festlegen:** Welche univariaten Methoden (z.B. IQR, mod. Z-Score) sind standardmäßig zu verwenden? Wann werden schiefe Daten transformiert?
3. **Dokumentationspflicht festlegen:** Definieren Sie ein Template, das bei jeder Datenbereinigung auszufüllen ist. Es sollte die identifizierten Ausreißer, die verwendete Methode, die Begründung für die Behandlung (Korrektur, Löschung, Beibehaltung) und den Namen des Verantwortlichen enthalten.
4. **Kommunikationswege klären:** Wie werden Erkenntnisse über Ausreißer an die Datenerzeuger (z.B. andere Abteilungen, Sensor-Betreiber) zurückgemeldet, um die Datenqualität an der Quelle zu verbessern?

Gute Datenqualität ist ein sehr großes Asset, das sich indirekt auch in einem besseren Unternehmensergebnis zeigt. Daher ist Datenqualität ein äußerst wichtiges strategisches Thema, das aktiv von der Geschäftsleitung unterstützt werden sollte.

10.2 Behandlungsstrategien

Hat man nach sorgfältiger Analyse gemäß Abschnitt 10.1 die Ausreißer validiert, stellt sich die Frage, wie man damit umgeht.

Die Wahl der angemessenen Behandlungsstrategie für identifizierte Ausreißer gehört zu den anspruchsvollsten Entscheidungen in der Datenanalyse. Sie erfordert eine sorgfältige Abwägung zwischen statistischer Korrektheit, fachlicher Plausibilität und den Zielen der Analyse.

Eine falsche Entscheidung kann sowohl zu irreführenden Ergebnissen als auch zum Verlust wichtiger Informationen führen.

Oft werden offensichtliche falsche Daten nicht in den Originaldaten korrigiert, sondern nur lokal bei den auszuwertenden Daten. Die Datenqualität bleibt damit schlecht. Es besteht die Gefahr, dass die nicht korrigierten Daten im Originalsystem in anderen nachgelagerten Systemen Fehlinterpretationen verursachen.

Wichtig: Ausreißer und fehlerhafter Datensatz

Ein **Ausreißer ist nicht zwangsläufig ein Fehler**. Er kann ein seltenes, aber korrektes und potenziell sehr aufschlussreiches Ereignis darstellen (z.B. ein Betrugsfall, ein Systemdurchbruch).

10.2.1 Korrektur

Die Korrektur von Ausreißern ist nur dann angemessen, wenn ein eindeutiger Fehler identifiziert werden kann und eine zuverlässige Methode zur Bestimmung des korrekten Wertes verfügbar ist. Diese Strategie erfordert besondere Sorgfalt, da sie eine aktive Veränderung der ursprünglichen Daten bedeutet.

Der Rückgriff auf Originaldaten oder alternative Quellen ist die bevorzugte Korrekturmethode.

Bei kritischen Anwendungen sollten Korrekturen immer durch mindestens zwei unabhängige Quellen validiert werden. Korrekturen sind immer nachvollziehbar zu dokumentieren.

Die Interpolation basierend auf Nachbarwerten bei Zeitreihen kann angemessen sein, wenn ein einzelner Messwert offensichtlich fehlerhaft ist, aber die umgebenden Werte plausibel erscheinen. Verschiedene Interpolationsmethoden – von einfacher linearer Interpolation bis hin zu komplexeren Spline-Verfahren – stehen zur Verfügung. Die Wahl der Methode sollte auf den Charakteristika der Zeitreihe und der Art des vermuteten Fehlers basieren.

Sowohl Interpolationsmethoden als auch Rekonstruktionsmethoden sollten nur auf den Analysedaten durchgeführt werden. Die Originaldaten sollten bei eindeutigen Fehlwerten als 'NULL' oder besser '#Fehler: alter Wert' ersetzt werden. Bei nichtstationären Reihen kann lineare Interpolation systematische Bias einführen (z. B. in volatilen Märkten).

Die Schätzung anhand verwandter Variablen nutzt die Korrelationsstruktur der Daten zur Rekonstruktion fehlerhafter Werte. Regressionsmodelle oder machine learning-basierte Verfahren können hierfür eingesetzt werden. Wichtig ist dabei, dass die verwendeten Prädiktorvariablen selbst nicht von Fehlern betroffen sind und dass die zugrundeliegenden Zusammenhänge stabil sind.

Dokumentation von Outlier-Entfernungen

Die Entfernung oder das Ersetzen von Ausreißern sollte immer sehr genau dokumentiert und begründet werden. Eine transparente Dokumentation aller Datenbereinigungsschritte ist für die Reproduzierbarkeit von Analysen unerlässlich.

10.2.2 Beibehaltung der Outlier

Die Beibehaltung von Ausreißern ist oft die wissenschaftlich korrekteste Entscheidung, besonders wenn sie authentische extreme Ereignisse repräsentieren. Diese Strategie erfordert

jedoch oft den Einsatz robuster Analysemethoden, um zu verhindern, dass einzelne extreme Werte die gesamte Analyse dominieren.

Extreme, aber plausible Ereignisse wie Naturkatastrophen, Finanzkrisen oder medizinische Seltenfälle können wichtige Erkenntnisse über die Grenzen und Eigenschaften des untersuchten Systems liefern. Das Entfernen solcher Datenpunkte würde die Realität verzerren und möglicherweise wichtige Risiken unterschätzen.

Innovative Leistungen oder außergewöhnliche Fälle repräsentieren oft Durchbrüche oder neue Möglichkeiten. In der Forschung können solche Ausreißer auf vielversprechende neue Ansätze hinweisen, während sie in der Wirtschaft disruptive Innovationen oder neue Marktchancen signalisieren können.

Viele wissenschaftliche Durchbrüche begannen als "Ausreißer" in den Daten – ihre Elimination hätte wichtige Entdeckungen verhindert.

Wichtige Minderheitengruppen in den Daten verdienen besondere Aufmerksamkeit aus ethischen und wissenschaftlichen Gründen. Das systematische Entfernen von Datenpunkten, die unterrepräsentierte Gruppen betreffen, kann zu verzerrten Ergebnissen und unfairen Algorithmen führen.

Insbesondere in KI-Anwendungen kann das Entfernen von Outliers zu algorithmischem Bias führen, z. B. wenn es marginalisierte Gruppen betrifft. Empfohlen wird eine Fairness-Prüfung (z. B. nach AEO-Guidelines der EU [24]). Dies ist besonders relevant bei der Entwicklung von KI-Systemen und medizinischen Behandlungsrichtlinien.

10.2.3 Robuste statistische Methoden

Wenn Ausreißer beibehalten werden, sollten robuste Methoden eingesetzt werden, um deren übermäßigen Einfluss zu minimieren. Statt des arithmetischen Mittels kann der Median oder der getrimmte Mittelwert verwendet werden. In Regressionsanalysen eignen sich M-Schätzer (z. B. Huber's Methode), die weniger sensibel auf Extremwerte reagieren. Dies gewährleistet stabile Ergebnisse, ohne Daten zu verlieren.

Huber's M-Estimator lässt den ursprünglichen Datensatz unberührt. Stattdessen wird der Einfluss der Ausreißer während des Schätzprozesses mathematisch begrenzt ([15] bzw. [16], Definition 1, Seite 104).

10.2.4 Separate Analyse

Eine differenzierte Herangehensweise, die verschiedene Szenarien parallel betrachtet, bietet oft den besten Kompromiss

zwischen statistischer Robustheit und Vollständigkeit der Information.

Die Hauptanalyse ohne Ausreißer liefert stabile, von Extremwerten unbeeinflusste Ergebnisse. Diese kann als konservative Baseline dienen und ist oft für praktische Anwendungen am besten geeignet. Sie sollte jedoch immer explizit als "bereinigte" Analyse ausgewiesen werden.

Ein Vergleich einer Analyse mit Outlier und ohne Outlier kann schnell einen Überblick schaffen, ob eine intensive Outlier-Analyse überhaupt notwendig ist.

Die Sensitivitätsanalyse mit verschiedenen Ausreißerdefinitionen zeigt, wie robust die Ergebnisse gegenüber unterschiedlichen Behandlungsstrategien sind. Wenn die Schlussfolgerungen unabhängig von der Ausreißerbehandlung ähnlich bleiben, erhöht dies das Vertrauen in die Ergebnisse. Große Unterschiede hingegen signalisieren, dass die Ausreißer einen erheblichen Einfluss haben und weitere Untersuchungen erforderlich sind.

Outlier (z.B. Wetterextreme, Stromspitzen etc.) werden meist separat modeliert. Oft sind es gerade die Extremwerte, die gut analysiert werden müssen und wichtige neue Erkenntnisse bringen.

Die spezielle Analyse der Ausreißer als eigene Gruppe kann wichtige Einblicke in seltene Phänomene oder Subpopulationen liefern. Diese Teilanalyse kann eigene wissenschaftliche Fragen beantworten und zur Hypothesengenerierung für zukünftige Studien beitragen.

10.2.5 Winsorizing

Das Winsorizing (auch als "Stutzen" bezeichnet) bietet einen Kompromiss zwischen der vollständigen Elimination von Ausreißern und ihrer unveränderten Beibehaltung. Bei diesem Verfahren werden extreme Werte auf bestimmte Perzentilwerte gesetzt, wodurch ihr Einfluss reduziert, aber nicht eliminiert wird.

Winsorizing (Stutzen) sollte nicht mit **Trimming** (Abschneiden) verwechselt werden. Beim Trimming werden die Extremwerte komplett entfernt, was die Stichprobengröße reduziert. Beim Winsorizing bleiben sie erhalten, aber ihr Wert wird angepasst.

Sei $X = (x_1, x_2, \ldots, x_n)$ eine geordnete Stichprobe mit $x_1 \leq x_2 \leq \cdots \leq x_n$ und seien $\alpha, \beta \in [0, 1]$ mit $\alpha + \beta < 1$ die unteren und oberen Winsorizing-Parameter.

Die **winsorisierte Stichprobe** $X^W = (x_1^W, x_2^W, \ldots, x_n^W)$ ist definiert durch:

$$x_i^W = \begin{cases} x_{\lceil n\alpha \rceil} & \text{falls } i \leq \lfloor n\alpha \rfloor \\ x_i & \text{falls } \lfloor n\alpha \rfloor < i < n - \lfloor n\beta \rfloor \\ x_{n-\lceil n\beta \rceil} & \text{falls } i \geq n - \lfloor n\beta \rfloor \end{cases}$$

wobei $\lfloor \cdot \rfloor$ die Gaußklammer (Abrundung) und $\lceil \cdot \rceil$ die Aufrundung bezeichnet.

Alternativ kann Winsorizing auch über Quantile definiert werden: Seien q_α und $q_{1-\beta}$ die α- bzw. $(1-\beta)$-Quantile von X. Dann ist:

$$x_i^W = \begin{cases} q_\alpha & \text{falls } x_i < q_\alpha \\ x_i & \text{falls } q_\alpha \leq x_i \leq q_{1-\beta} \\ q_{1-\beta} & \text{falls } x_i > q_{1-\beta} \end{cases}$$

Für symmetrisches Winsorizing gilt $\alpha = \beta$.

Das 95%-Winsorizing ist eine häufig verwendete Variante, bei der Werte unter dem 5. Perzentil auf diesen Wert gesetzt werden. Gleiches gilt für Werte über dem 95. Perzentil. Diese Transformation reduziert den Einfluss der extremsten 5% der Werte an beiden Enden der Verteilung, während die Stichprobengröße erhalten bleibt.

Der Vorteil des Winsorizing liegt darin, dass es die Stichprobengröße erhält und gleichzeitig den verzerrenden Einfluss extremer Werte reduziert. Dies ist besonders wertvoll bei kleinen Stichproben, wo jeder Datenpunkt wichtig ist. Allerdings verändert das Verfahren die ursprüngliche Datenverteilung und kann bei unsachgemäßer Anwendung zu künstlichen Häufungen an den Winsorizing-Grenzen führen.

Vorsicht

Winsorizing kann die Varianz unterschätzen und zu verzerrten Inferenzen führen, insbesondere bei nichtnormalverteilten Daten (z. B. in Finanzdaten mit fat tails). Testen Sie die Auswirkungen durch Sensitivitätsanalyse.

Die Wahl des Winsorizing-Levels sollte auf der Verteilung der Daten und den Zielen der Analyse basieren. Während 5% an jedem Ende ein gängiger Standard ist, können je nach Kontext auch andere Werte (1%, 10%) angemessen sein. Wichtig ist, dass die Entscheidung a priori getroffen und transparent dokumentiert wird.

10.3 Praktische Umsetzung der Outlier-Analyse

Die theoretischen Grundlagen der univariaten Ausreißererkennung sind nur dann wertvoll, wenn sie systematisch und reproduzierbar in die tägliche Analysepraxis überführt werden. Dies erfordert sowohl strukturierte Workflows als auch die geschickte Nutzung moderner Werkzeuge zur Automatisierung wiederkehrender Aufgaben. Die folgenden Anleitungen bieten konkrete Schritte für die Implementierung robuster Ausreißererkennungsprozesse in verschiedenen Arbeitsumgebungen.

Eine strukturierte Herangehensweise reduziert nicht nur Fehler, sondern macht Analyseergebnisse auch für Kollegen und Aufsichtsbehörden nachvollziehbar und reproduzierbar.

Die Automatisierung der Ausreißererkennung mittels Skripten und Programmen ermöglicht es, auch große Datensätze systematisch zu durchleuchten und dabei gleichzeitig eine konsistente Methodik zu gewährleisten. Moderne Sprachmodelle können dabei als intelligente Assistenten fungieren,

die maßgeschneiderte Analyseskripte erstellen und dabei bewährte statistische Praktiken automatisch umsetzen.

To Do: Systematische Outlier-Analyse für Ihren Datensatz

Führen Sie eine vollständige univariate Ausreißeranalyse für Ihren aktuellen Datensatz durch:

1. **Datenexploration:** Erstellen Sie für jede numerische Variable Histogramme und Boxplots, um die Verteilungsform zu verstehen. Notieren Sie sich Schiefe, mögliche Multimodalität und offensichtliche Extremwerte.
2. **Methodenauswahl:** Wählen Sie basierend auf der Verteilungsform die geeigneten Erkennungsmethoden aus. Idealerweise werden alle in Kapitel 9 vorgestellten Verfahren durchgeführt, falls nötig (bei einer signifikanten Schiefe der Daten) zusätzlich auf die mit dem Yeo-Johnson-Verfahren transformierten Daten.
3. **Grenzwerte dokumentieren:** Legen Sie die Schwellenwerte fest (z.B. $|Z| > 3$, IQR-Faktor 1.5) und begründen Sie Ihre Wahl schriftlich.
4. **Ausreißer identifizieren:** Wenden Sie mindestens zwei verschiedene Methoden an und vergleichen Sie die Ergebnisse. Erstellen Sie eine Übersichtstabelle mit allen identifizierten Ausreißern und den verwendeten Methoden.
5. **Validierung durchführen:** Überprüfen Sie jeden identifizierten Ausreißer auf Plausibilität, Messfehler und sachliche Korrektheit. Dokumentieren Sie Ihre Entscheidungen mit Begründung.
6. **Behandlungsstrategie umsetzen:** Entscheiden Sie für jeden Ausreißer über Korrektur, Entfernung, Beibehaltung oder Winsorizing. Erstellen Sie sowohl einen bereinigten als auch einen unbereinigten Datensatz.
7. **Sensitivitätsanalyse:** Führen Sie Ihre Hauptanalyse mit beiden Datensätzen durch und bewerten Sie die Stabilität Ihrer Ergebnisse gegenüber der Ausreißerbehandlung.

Die Dokumentation jedes Schritts ist nicht nur für die Nachvollziehbarkeit wichtig, sondern auch für das Lernen aus Erfahrungen und die kontinuierliche Verbesserung der Datenqualitätsprozesse.

Folgender Prompt gibt eine Sensitivitätsanaylse für die Weindaten aus [18]:

Prompt für Sensitivitätsanalyse zu Outliers

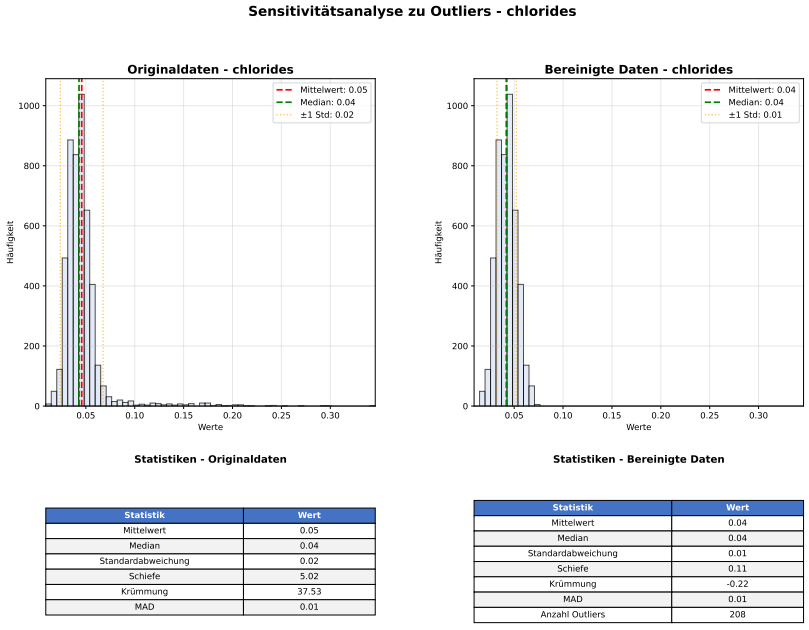
Das Verzeichnis ist *C:\Daten*. In der Datei *winequality-white.csv* sind in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen {Python-Quellcode}, der nach Eingabe eines Feldnamens folgendes macht:

1. Berechne die Statistiken Mittelwert, Median, Standardabweichung, Schiefe, Krümmung (Excess Kurtosis), MAD für die Originaldaten.
2. Identifiziere alle Outliers (IQR-Methode mit 1.5xGrenzen). Bereinige die Daten um die Outlier und berechne die analogen Statistiken auf die bereinigten Daten. Ergänze um die Anzahl der Outlier.

3. Erstelle jeweils für die Originaldaten und die bereinigten Daten eine gut strukturierte Tabelle der berechneten Statistiken (2 Nachkommastellen).
4. Erstelle jeweils für die Originaldaten und die bereinigten Daten ein Balkendiagramm (60 bins) (Farbcode #D8E1F4 für Balken), das die jeweiligen Mittelwerte, Mediane und Standardabweichungen zeigt. Benutze jeweils die gleiche Skalierung.
5. Gib folgende (2x2) Grafik aus: links - Das Balkendiagramm mit den Originaldaten und darunter die Tabelle mit den Statistiken der Originaldaten. Rechts analog für die bereinigten Daten.
6. Speichere die Grafik als *sensitivity_bar_{Feldname}.pdf*.

Prompt 10.1: Prompt für Sensitivitätsanalyse zu Outliers

Das Ergebnis für das Feld *chlorides* ist in folgender Abbildung:



Statistik	Wert
Mittelwert	0.05
Median	0.04
Standardabweichung	0.02
Schiefe	5.02
Krümmung	37.53
MAD	0.01

Statistik	Wert
Mittelwert	0.04
Median	0.04
Standardabweichung	0.01
Schiefe	0.11
Krümmung	-0.22
MAD	0.01
Anzahl Outliers	208

Die Krümmung hier ist die Excess Kurtosis - d.h. Normalverteilung hätte Excess Kurtosis 0 und Schiefe 0.

Abbildung 10.1: Sensitivitätsanaylse der Outlier für das Feld *chlorides* aus den Daten aus [18].

Die Sensitivitätsanalyse zeigt einen signifikanten Einfluss von Ausreißern auf die deskriptiven Statistiken.

Verzerrung durch Ausreißer: Die 208 identifizierten Ausreißer verzerren die klassischen Kennzahlen erheblich.

Stabilität robuster Maße: Der Median (0.04) und die Median-Absolutabweichung (MAD, 0.01) bleiben von der Datenbereinigung unberührt. Sie beschreiben die zentrale Tendenz und Streuung der Mehrheit der Daten zuverlässig.

Nach Bereinigung liegt nahezu eine Normalverteilung vor (Schiefe, Excess Kurtosis nahe 0, Median und Mean identisch).

Der Mittelwert (0.05) wird vom Median (0.04) weggezogen. Die Standardabweichung wird verdoppelt (0.02 vs. 0.01). Die Schiefe (5.02) und die Kurtosis (37.53) sind extrem hoch.

Insbesondere extreme Outlier sind immer interessant und sollten näher untersucht werden.

Es wäre daher ratsam, die 208 identifizierten Ausreißer näher zu untersuchen. Handelt es sich um Messfehler, Eingabefehler oder repräsentieren sie ein besonderes, aber seltenes Phänomen? Diese Erkenntnis könnte für das Verständnis des zugrundeliegenden Prozesses wertvoll sein.

Die Ausreißer zu ignorieren, würde zu falschen Schlussfolgerungen führen, auch wenn die bereinigte Verteilung als Normalverteilung aus Analysesicht sehr "angenehm" ist.

10.4 Zusammenfassung

Dieses Kapitel entwickelt einen systematischen Ansatz zum Umgang mit identifizierten Ausreißern, der methodische Korrektheit mit fachlicher Expertise verbindet und über reine Statistik hinausgeht. Es unterstreicht die Dualität von Ausreißern als potenzielle Störfaktoren oder wertvolle Informationsträger, was einen kontextabhängigen Entscheidungsprozess erfordert.

AIC für die optimale Verteilungsauswahl

11

Oft ist es sinnvoll, für eine gegebene Stichprobe – z.B. eine univariate Reihe von Messwerten wie Längen, Gewichten oder Zeiten – eine geeignete theoretische Wahrscheinlichkeitsverteilung zu identifizieren.

Eine gute theoretische Verteilung ermöglicht eine präzise Beschreibung der Datenstruktur und bildet darüber hinaus die Grundlage für zuverlässige statistische Inferenz.

Zudem können signifikante Abweichungen von der theoretisch besten Verteilung beispielsweise auf Häufungsanomalien (Heapings) hinweisen.

Merke:

Insbesondere wenn es bestimmte Grenzen gibt - z.B. eine steuerliche Einkommensgrenze für eine Vergünstigung - können Heaping-Effekte auftreten.

Modelle mit expliziten Verteilungen können robuster sein und sich unter Umständen besser für Prognosen eignen. Dies spart Ressourcen und reduziert Unsicherheiten in der Entscheidungsfindung.

Eine systematische Methode, um aus einer Menge von Kandidatenverteilungen die beste auszuwählen, ist die statistische Modellselektion, die im folgenden Abschnitt detailliert erläutert wird.

11.1 Statistische Modellselektion

In der modernenstatistischen Analyse und im maschinellen Lernen stehen Forschende und Anwender häufig vor einer fundamentalen Herausforderung: der Auswahl des „besten“ Modells aus einer Menge von Kandidatenmodellen.

Ein sehr gutes Buch zum Thema ist *Model Selection and Multimodel Inference* von Kenneth P. Burnham und David R. Anderson ([25]).

Jedes Modell stellt eine vereinfachte Hypothese über die komplexen, datengenerierenden Prozesse der realen Welt dar. Die Kunst besteht darin, ein Modell zu finden, das die zugrunde liegenden Muster in den Daten adäquat erfasst, ohne dabei das zufällige Rauschen zu interpretieren. Dieses Spannungsfeld zwischen Anpassungsgüte und Modellkomplexität ist als Bias-Varianz-Dilemma bekannt und bildet den Kern der Modellselektion.

Hirotugu Akaike (1927–2009) war ein japanischer Statistiker. Er entwickelte das AIC in den frühen 1970er Jahren während seiner Arbeit am Institut für Statistische Mathematik in Tokio. Seine Motivation war, eine Brücke zwischen der Welt der Statistik und der Informationstheorie zu schlagen.

Das in den frühen 1970er Jahren von Hirotugu Akaike entwickelte **Akaike-Informationskriterium (AIC)** ist ein zentrales und weit verbreitetes Werkzeug zur Lösung dieses Problems. Die primäre Motivation hinter dem AIC ist die Schaffung eines objektiven Kriteriums, das einen formalen Kompromiss

zwischen der Güte der Modellanpassung an die beobachteten Daten und der Einfachheit des Modells ermöglicht.

Ein Modell, das zu viele Parameter enthält, neigt zur Überanpassung (Overfitting). Es modelliert nicht nur das Signal, sondern auch das Rauschen in den Daten und verliert dadurch seine Fähigkeit, neue, ungesehene Daten vorherzusagen.

Der Begriff "Information" im AIC bezieht sich auf den Informationsverlust. Das beste Modell ist dasjenige, das bei der Approximation der Realität am wenigsten Information verliert.

Ein zu einfaches Modell hingegen könnte wichtige Strukturen und Zusammenhänge übersehen und leidet unter Unteranpassung (Underfitting).

11.2 Akaike-Informationskriterium

Die Anwendungsbereiche des AIC sind außerordentlich breit und erstrecken sich über zahlreiche wissenschaftliche Disziplinen. Im maschinellen Lernen stellt es eine fundamentale Methode dar, um der Gefahr der Überanpassung entgegenzuwirken und Modelle mit hoher Generalisierungsfähigkeit zu finden.

Das **Akaike-Informationskriterium (AIC)** wird formal definiert als:

$$\text{AIC} = 2k - 2\ln(\hat{L})$$

Das AIC-Kriterium folgt Einsteins Maxime *„Everything should be made as simple as possible, but not simpler"*: Der Term $-2 \cdot ln(L)$ belohnt gute Datenanpassung, während +2k Modellkomplexität bestraft.

k ist die Anzahl der im Modell geschätzten Parameter. Dies ist der **Strafterm** für die Modellkomplexität. Für jeden zusätzlichen Parameter wird das AIC um den Wert 2 erhöht.

$\hat{L}$ der maximierte Wert der Likelihood-Funktion (vgl. Kapitel B) für das Modell ist. Der Term $-2\ln(\hat{L})$ ist ein Maß für die **Anpassungsgüte** des Modells an die Daten. Ein besser angepasstes Modell hat einen höheren $\hat{L}$-Wert und somit einen kleineren $-2\ln(\hat{L})$-Wert.

AIC ist ein relatives Maß. Die absolute Höhe ist von der Anzahl der Stichprobedaten abhängig. Daher dürfen nur AIC-Werte von Modellen verglichen werden, die exakt dieselben Stichprobendaten verwenden.

Das Ziel der Modellselektion mit dem AIC ist es, dasjenige Modell aus einer Menge von Kandidatenmodellen auszuwählen, das den **kleinsten AIC-Wert** aufweist. Dieser Wert selbst hat keine absolute Bedeutung. Er ist nur für den relativen Vergleich von Modellen relevant, die auf denselben Daten angepasst wurden.

Beispiel: AIC für Verteilungsvergleich - Log-Normal vs. Normal

Angenommen, wir haben eine Stichprobe von $n = 250$ positiven Messwerten (z.B. Einkommen, Reaktionszeiten, oder Partikelgrößen) und möchten entscheiden, ob diese Daten besser durch eine Normalverteilung oder eine Log-Normalverteilung beschrieben

Die Anzahl der Datenpunkte bestimmt die Höhe des AIC. Damit ist die absolute Höhe nur bei genau denselben Daten vergleichbar.

werden.

Stichprobendaten: $x = (x_1, x_2, \ldots, x_{250})$ mit $x_i > 0$ für alle i.

Modell 1: Normalverteilung

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

Dichtefunktion: $f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$

Anzahl Parameter: $k_1 = 2$ (μ, σ^2)

Modell 2: Log-Normalverteilung

$$X \sim \text{LogNormal}(\mu_{\ln}, \sigma_{\ln}^2)$$

Dichtefunktion: $f(x|\mu_{\ln}, \sigma_{\ln}^2) = \frac{1}{x\sqrt{2\pi\sigma_{\ln}^2}} \exp\left(-\frac{(\ln x - \mu_{\ln})^2}{2\sigma_{\ln}^2}\right)$

Anzahl Parameter: $k_2 = 2$ $(\mu_{\ln}, \sigma_{\ln}^2)$

Maximum-Likelihood-Schätzer:

Für die Normalverteilung:

$$\hat{\mu} = \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Für die Log-Normalverteilung:

$$\hat{\mu}_{\ln} = \frac{1}{n}\sum_{i=1}^{n} \ln(x_i)$$

$$\hat{\sigma}_{\ln}^2 = \frac{1}{n}\sum_{i=1}^{n}(\ln(x_i) - \hat{\mu}_{\ln})^2$$

Für diese beiden Verteilungen sind die Maximum-Likelihood-Schätzer genau die Schätzer für die Momente und können explizit angegeben werden. Das ist aber eher die Ausnahme (vgl. Anhang B)

Geschätzte Parameter:

Modell	Parameter
Normalverteilung	$\hat{\mu} = 45{,}32$, $\hat{\sigma}^2 = 312{,}75$
Log-Normalverteilung	$\hat{\mu}_{\ln} = 3{,}65$, $\hat{\sigma}_{\ln}^2 = 0{,}18$

Log-Likelihood-Berechnung: Für die Normalverteilung:

$$\ell_1(\hat{\mu}, \hat{\sigma}^2) = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\hat{\sigma}^2) - \frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{n}(x_i - \hat{\mu})^2$$

Hier wird $-\ln$ auf $\prod_{i=1}^{n} f(x_i|\hat{\mu}, \hat{\sigma})$ angewandt. Die Multiplikation wird dadurch zur Addition.

Für die Log-Normalverteilung:

$$\ell_2(\hat{\mu}_{\ln}, \hat{\sigma}_{\ln}^2) = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\hat{\sigma}_{\ln}^2) - \sum_{i=1}^{n}\ln(x_i) - \frac{1}{2\hat{\sigma}_{\ln}^2}\sum_{i=1}^{n}(\ln x_i - \hat{\mu}_{\ln})^2$$

Die Log-Likelihood ist die Summe der logarithmierten Einzelwahrscheinlichkeiten aller Beobachtungen unter dem angenommenen Modell.

Log-Likelihood-Werte:

Modell	$\ell(\hat{\boldsymbol{\theta}})$
Normalverteilung	−1247,83
Log-Normalverteilung	−856,91

AIC-Berechnung:

$$\text{AIC}_{\text{Normal}} = -2 \cdot (-1247{,}83) + 2 \cdot 2 = 2495{,}66 + 4 = 2499{,}66$$
$$\text{AIC}_{\text{LogNormal}} = -2 \cdot (-856{,}91) + 2 \cdot 2 = 1713{,}82 + 4 = 1717{,}82$$

Modellvergleich:

Modell	**AIC**	**ΔAIC**
Normalverteilung	2499,66	781,84
Log-Normalverteilung	1717,82	0,00

Interpretation: Die Log-Normalverteilung zeigt einen deutlich niedrigeren AIC-Wert und ist damit das klar bevorzugte Modell.

Die Wahl der Kandidatenverteilungen sollte nicht willkürlich sein, sondern auf theoretischen Überlegungen oder explorativer Datenanalyse (z.B. Histogramme, Q-Q-Plots) basieren.

11.3 Die Wahl der Modelle

Eine der direktesten Anwendungen des AIC ist die Auswahl einer geeigneten Wahrscheinlichkeitsverteilung zur Beschreibung eines gegebenen Datensatzes. In vielen statistischen Analysen ist die Annahme einer bestimmten Verteilung grundlegend und das AIC bietet eine datengestützte Methode, um diese wichtige Entscheidung zu treffen.

Hat man Schiefe, Kurtosis und den Wertebereich der Stichprobedaten, dann kann das Testen folgender Verteilungen Sinn machen:

Tabelle 11.1: Kriterien für Verteilungstests basierend auf Schiefe (S), Kurtosis (K) und Vorzeichen (V)

Verteilung	**Schiefe (S)**	**Kurtosis (K)**	**Vorzeichen (V)**
Normalverteilung	$\lvert S\rvert < 1$	$\lvert K-3\rvert < 2$	irrelevant
Log-Normalverteilung	$S > 0.5$	$K > 3$	nur positive
Exponentialverteilung	$S \approx 2$	$K \approx 9$	nur positive
Beta-Verteilung	$-2 < S < 2$	$1.8 < K < 6$	nur positive
Gamma-Verteilung	$S > 0$	$K > 3$	nur positive
Gumbel-Verteilung	$S \approx 1.14$	$K \approx 5.4$	irrelevant
Weibull-Verteilung	$-0.5 < S < 3.6$	$1.8 < K < 25$	nur positive
Gleichverteilung	$S \approx 0$	$K \approx 1.8$	irrelevant

Die Werte in der Tabelle sind Richtwerte und können je nach Stichprobengröße variieren. Bei kleinen Stichproben sollten die Toleranzen größer gewählt werden. “irrelevant“ bei Vorzeichen bedeutet, dass sowohl positive als auch negative Werte möglich sind. “nur positive“ bedeutet entsprechend, dass die Verteilung nur für positive Werte sinnvoll ist.

11.4 Praxisbeispiel: Modellauswahl

Es wird das Datenset wie in Abschnitt 9.3 verwendet (aus [18]). Als Feld wird “free sulfur dioxide“ ausgewählt.

Die Schiefe ist 1.406, die Kurtosis ist 14.46. Die Werte sind alle positiv.

Als mögliche Verteilungen werden gemäß Tabelle 11.1 die Log-Normalverteilung, die Gamma-Verteilung und die Weibullverteilung ausgewählt.

Für das LLM wird folgender Prompt formuliert:

Prompt für AIC-optimierte Modellauswahl

Das Verzeichnis ist *C:\Daten*. In der Datei *winequality-white.csv* sind in der ersten Zeile die Feldnamen. Die Trennung ist mit “;“. Erstelle einen {Python-Quellcode}, der folgendes macht:

1. Wähle das Feld ’free sulfur dioxide’
2. Berechne für die Daten jeweils den AIC-Wert für folgende Verteilungen: Log-Normalverteilung, Gamma-Verteilung, und Weibullverteilung
3. Erstelle ein Histogramm mit 150 bins der Daten im Farbcode #D8E1F4 und lege die Verteilung mit dem niedrigsten AIC-Wert als blaue Linie über die Daten. Gib auch den Namen der Verteilung und die Parameter in der Abbildung an.
4. Speichere das Histogramm mit der Modellverteilung unter AIC_Feldname.pdf

Der Prompt kann auch allgemeiner formuliert sein und die Aussagen aus Tabelle 11.1 direkt in den Prompt einfließen. Dann ist das vom LLM generierte Python für alle Felder gültig.

Prompt 11.1: Prompt für AIC Modelauswahl

Das über das LLM generierte Python-Programm liefert als beste Verteilung eine Gamma-Verteilung:

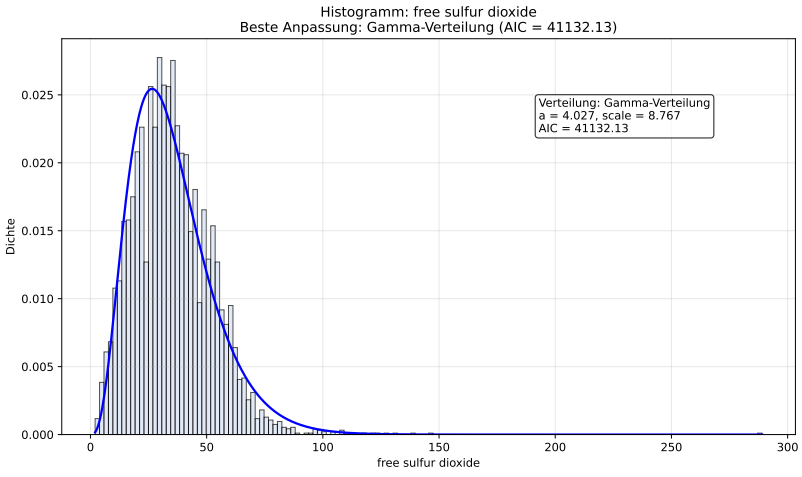


Abbildung 11.1: Bestes Modell gemäß AIC für das Feld “free sulfur dioxide“ ist eine Gamma-Verteilung mit $a = 4.027$ und Scale-Faktor $b = 8.767$

Zusätzlich soll ein Q-Q-Plot erstellt werden:

Prompt für Q-Q-Plot

Das Verzeichnis ist *C:\Daten*. In der Datei *winequality-white.csv* sind in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen {Python-Quellcode}, der folgendes macht:

1. Wähle das Feld 'free sulfur dioxide'
2. Erstelle einen Q-Q-Plot für das ausgewählte Feld (#D8E1F4 mit schwarzer Umrandung) und der Gamma-Verteilung (blau) mit den Parametern $a = 4.027$ und Scale-Faktor $b = 8.767$.
3. Erstelle die kumulierte Verteilung des ausgewählten Feldes (#D8E1F4, gestrichelt) und der Gamma-Verteilung (blau, durchgehend) in einem weiteren Bild und markiere die maximale Differenz der Abweichung. Gib die maximale Abweichung als Wert auch im Plot an.
4. Bitte benutze nur die Farbe blau und #D8E1F4 und Schwarz.
5. Speichere das den Q-Q-Plot unter QQ_Feldname.pdf und die kumulierten Verteilungen unter KS_Feldname.pdf.

Prompt 11.2: Prompt Q-Q-Plot für AIC Modell

Das über das LLM (Claude Sonnet 4) generierte Python-Programm liefert folgenden Q-Q-Plot:

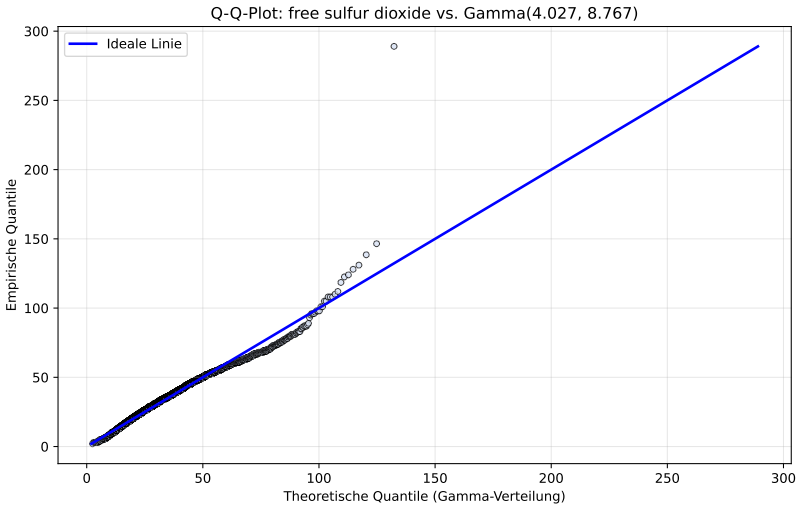


Abbildung 11.2: Q-Q-Plot für das Feld "free sulfur dioxide" mit der Gamma-Verteilung ($a = 4.027$ und Scale-Faktor $b = 8.767$)

Der Q-Q-Plot zeigt signifikante Abweichungen.

Der Vergleich aus der empirischen Verteilungsfunktion und der kumulierten Gamma-Verteilung ($a = 4.027$ und Scale-Faktor $b = 8.767$) zeigt eine maximale Abweichung von 0.0456.

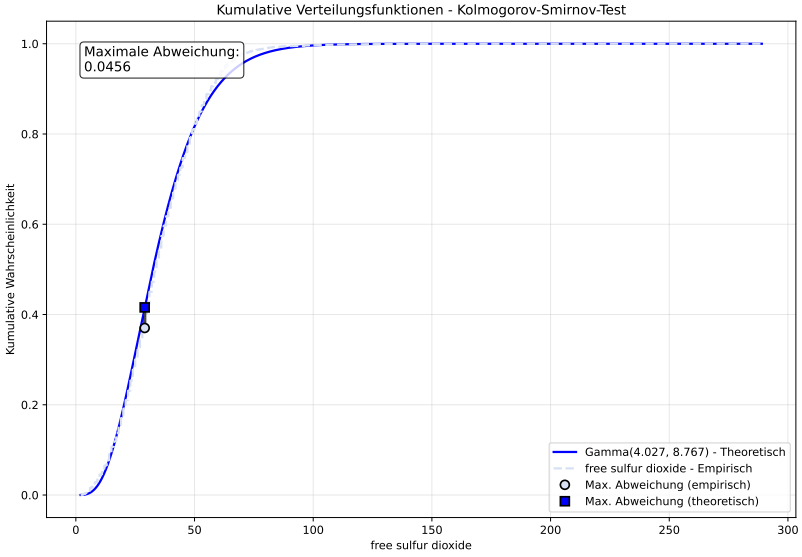


Abbildung 11.3: Empirische Verteilungsfunktion und die kumulierte Gamma-Verteilung ($a = 4.027$ und Scale-Faktor $b = 8.767$). Der maximale Abstand mit 0.0456 ist insbesondere aufgrund der Anzahl der Datenpunkte von 4898 relativ hoch.

Auf Basis des Q-Q-Plots und maximalen Abweichungen sind Zweifel berechtigt, in wie weit die gefundene Gamma-Funktion eine "gute" Approximation für die empirischen Daten aus dem Feld "free sulfur dioxide" sind. Es soll daher folgende Hypothese getestet werden:

H_0: Die Variable 'free sulfur dioxide' folgt einer Gamma-Verteilung mit geschätzten Parametern $\hat{a} = 4.027$ und $\hat{b} = 8.767$.

H_1: Die Variable folgt nicht dieser Gamma-Verteilung.

Da die Parameter $\hat{a}$ und $\hat{b}$ aus den Daten geschätzt wurden, kann die asymptotische Verteilung der KS-Teststatistik nicht direkt verwendet werden (vgl. Kapitel C). Daher wird ein Bootstrap-Verfahren eingesetzt, um die Nullverteilung der Teststatistik zu approximieren.

Das Bootstrap-Verfahren simuliert die Situation unter H_0 und berücksichtigt dabei die Unsicherheit der Parameterschätzung. Durch die Neuschätzung der Parameter in jeder Bootstrap-Iteration wird die korrekte Nullverteilung approximiert.

Der Hypothesen-Test wird über folgenden Prompt umgesetzt:

Prompt für Bootstrap-Goodness-of-Fit-Test

Das Verzeichnis ist *C:\Daten*. In der Datei *winequality-white.csv* sind in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen {Python-Quellcode}, der folgendes macht:

1. Wähle das Feld 'free sulfur dioxide'
2. Implementiere einen Bootstrap-Goodness-of-Fit-Test für die Gamma-Verteilung mit den Parametern $a = 4.027$ und Scale-Faktor $b = 8.767$ mit $B = 1000$ Bootstrap-Iterationen
3. Verwende als Teststatistik den Kolmogorov-Smirnov-Test (maximale Abweichung zwischen empirischer und theoretischer Verteilungsfunktion)
4. Berechne den empirischen p-Wert durch Vergleich der beobachteten Teststatistik mit der Bootstrap-Verteilung
5. Erstelle ein Histogramm der Bootstrap-Teststatistiken mit einer vertikalen Linie bei der beobachteten Teststatistik
6. Gib den p-Wert und die Testentscheidung bei $\alpha = 0.05$ aus
7. Bitte benutze nur die Farbe blau und #D8E1F4 und Schwarz.
8. Speichere das Histogramm unter Bootstrap_Test_{Feldname}.pdf

Prompt 11.3: Prompt für Goodness of Fit Test für AIC

Das Ergebnis (LLM ist ChatGPT 5) ist in folgender Abbildung und zeigt, dass die H_0-Hypothese nicht aufrechterhalten werden kann:

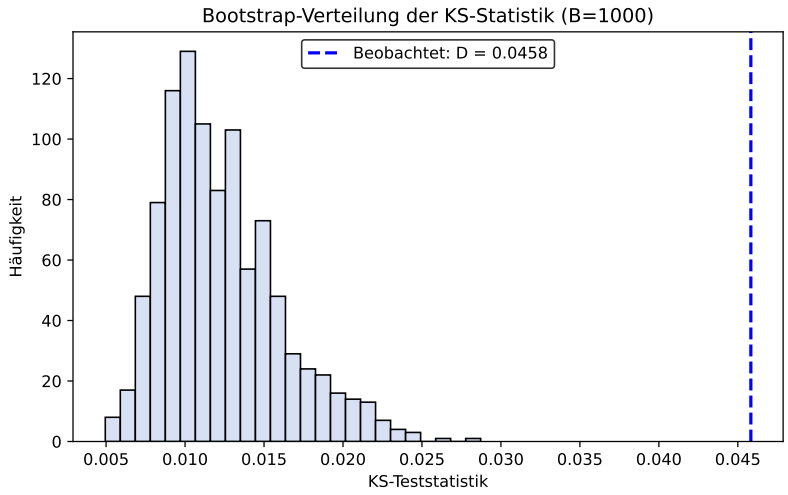


Abbildung 11.4: Bootstrap-Goodness-of-Fit-Test für die Gamma-Verteilung mit den Parametern $a = 4.027$ und Scale-Faktor $b = 8.767$. Die H0-Hypothese kann nicht aufrechterhalten werden.

11.5 Anwendung auf Heaping-Punkte

Auch wenn über die AIC-Modellsuche oft keine "Hypothesenstarke" beste Verteilung gefunden werden kann, kann sie helfen, Anomalien zu erkennen.

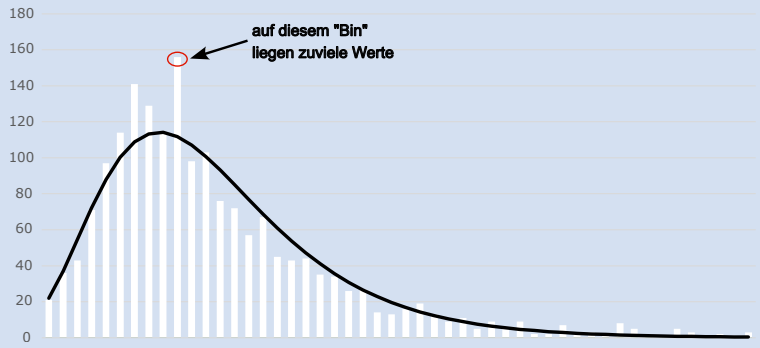
Folgendes Beispiel soll die Anwendung des besten Modells demonstrieren, um Heaping-Bereiche aufzudecken:

Beispiel: AIC für Einkommensdaten

Aus einer Kreditdatenbank liegen Einkommensdaten vor. Es wurden verschiedene Modelle getestet, die in folgender Tabelle zusammengefasst sind.

Verteilung	AIC	Parameter 1	Parameter 2
Log-Normal	30,603	7,94	0,39
Gumbel	30,681	2.441,87	1.010,95
Gamma	30,771	5,44	555,68
Normal	31,275	3.025,40	1.296,59
Weibull	31,438	3,10	3.382,78
Uniform	32,679	1.117,00	8.993,76
Exponential	32,834	0,00	0,00

Den besten Anpassungswert hat die LogNormal-Verteilung. Ein Vergleich mit den Echt-Daten zeigt eine Auffälligkeit.



Dies deutet auf eine erhöhte Häufigkeit in einer Einkommenskategorie hin. In diesem Fall wäre zu prüfen, ob dieser "Heaping"-Wert zufällig oder systembedingt ist, z.B. dann wenn dadurch knapp eine Einkommensgrenze erreicht wird, die einen günstigeren Kredit möglich macht (z.B. Sachbearbeiter bei Credit Scoring). Eine reine TopN-Betrachtung oder ein Clumpiness-Score hätte den Bereich nicht aufgedeckt, da es sich unter Umständen um viele Werte (z.B. 3000,12 etc.) handelt, die aber über der 3000 Euro-Grenze liegen.

11.6 Einschränkungen und praktische Hinweise

Obwohl das AIC ein äußerst nützliches Werkzeug ist, ist seine Anwendung mit wichtigen Einschränkungen und Überlegungen verbunden, die für eine korrekte Interpretation unerlässlich sind.

Zunächst ist es entscheidend zu verstehen, dass AIC-Werte **kein absolutes Maß für die Modellgüte** sind. Ein niedriger AIC-Wert bedeutet nicht zwangsläufig, dass ein Modell gut ist. Er bedeutet nur, dass es das beste unter den betrachteten Kandidatenmodellen ist. Wenn alle Kandidatenmodelle eine schlechte Beschreibung der Realität sind, wird das AIC lediglich das „am wenigsten schlechte" auswählen. Die Definition einer a priori sinnvollen Menge von Kandidatenmodellen,

Ein häufiger Fehler ist der Vergleich von AIC-Werten für Modelle, die mit unterschiedlichen Datensätzen angepasst wurden, z.B. nach Entfernung von fehlenden Werten für bestimmte Variablen. Dies ist unzulässig.

die auf theoretischem und Domänenwissen basiert, ist daher von größter Bedeutung.

Merke: AIC prüft nicht die Modellannahmen! Ein Modell kann einen niedrigen AIC-Wert haben und trotzdem z.B. die Annahme der normalverteilten Residuen verletzen. Die diagnostische Prüfung bleibt unerlässlich.

Ein weiterer wichtiger Punkt ist, dass das AIC nur zum Vergleich von Modellen verwendet werden kann, die auf **exakt denselben Daten** angepasst wurden. Jede Veränderung am Datensatz, wie das Entfernen von Beobachtungen mit fehlenden Werten, erfordert eine Neuanpassung aller Modelle an den reduzierten Datensatz, bevor ihre AIC-Werte verglichen werden können.

Obwohl das AIC bei großen Stichproben zur Auswahl von zu komplexen Modellen neigen kann (wo das BIC oft bevorzugt wird), bleibt es ein Standardwerkzeug für prädiktive Modellierung. In der Praxis sollte das AIC niemals isoliert verwendet werden. Die Auswahl eines Modells sollte immer durch **diagnostische Überprüfungen** ergänzt werden. Für das oder die bestplatzierten Modelle sollten die zugrunde liegenden Annahmen (z.B. Normalität und Homoskedastizität der Residuen in der linearen Regression) geprüft werden. Grafische Darstellungen wie Residuen-Plots sind hierbei unerlässlich.

AICc (korrigiertes AIC) = $AIC + \frac{2k(k+1)}{n-k-1}$ berücksichtigt kleine Stichprobengrößen. Für $n/k < 40$ empfohlen, konvergiert zu AIC für große n. Verhindert Überanpassung bei wenigen Datenpunkten relativ zur Parameteranzahl.

To Do Checkliste für die AIC-Anwendung

1. Definiere eine wissenschaftlich fundierte Menge an Kandidatenmodellen, bevor die Analyse beginnt.
2. Stelle sicher, dass alle Modelle auf exakt demselben Datensatz angepasst werden.
3. Verwende bei kleinen Stichproben ($n/k < 40$) das AICc anstelle des AIC.
4. Betrachte nicht nur das Modell mit dem niedrigsten AIC, sondern auch Modelle mit $\Delta_i < 2$ als plausibel.
5. Berechne bei Modellunsicherheit Akaike-Gewichte und erwäge eine Modellmittelung.

Akaike-Gewichte transformieren AIC-Differenzen in Wahrscheinlichkeiten:

$$w_i = \frac{\exp(-\frac{1}{2}\Delta_i)}{\sum \exp(-\frac{1}{2}\Delta_k)}.$$

11.7 Zusammenfassung

Dieses Kapitel hat das Akaike-Informationskriterium (AIC) als ein fundamentales Werkzeug der statistischen Modellselektion vorgestellt. Es wurde gezeigt, dass das AIC auf den Prinzipien der Informationstheorie und der Maximum-Likelihood-Schätzung aufbaut und einen eleganten Kompromiss zwischen der Anpassungsgüte eines Modells an die Daten und seiner Komplexität bietet.

Die zentralen Erkenntnisse sind: Die formale Definition des AIC als $\text{AIC} = 2k - 2\ln(\hat{L})$ balanciert die Anzahl der Parameter k als Strafterm gegen die maximierte Log-Likelihood

$\ln(\hat{L})$ als Maß für die Anpassung. Das Ziel ist die Auswahl des Modells mit dem geringsten AIC-Wert. Für kleine Stichproben liefert das korrigierte AIC (AICc) eine zuverlässigere Schätzung, indem es die Komplexität stärker bestraft.

Die Anwendung des AIC ist vielfältig und reicht von der Auswahl geeigneter Wahrscheinlichkeitsverteilungen bis hin zur Variablenselektion in komplexen Regressionsmodellen. Es bietet eine objektive, datengestützte Grundlage für Entscheidungen, die andernfalls subjektiv wären. Darüber hinaus ermöglichen Erweiterungen wie die Akaike-Gewichte und die Modellmittelung einen differenzierten Umgang mit Modellunsicherheit, was zu robusteren und ehrlicheren wissenschaftlichen Schlussfolgerungen führt.

AIC ist auch eine wichtige Maßzahl bei Modellen des maschinellen Lernens und bei (logistischen) Regressionen.

Trotz seiner Stärken ist das AIC kein Allheilmittel. Seine korrekte Anwendung erfordert ein Verständnis seiner theoretischen Grundlagen und Einschränkungen. Insbesondere ist das AIC ein relatives Maß und seine Ergebnisse sollten stets im Kontext von Domänenwissen und diagnostischen Modellprüfungen interpretiert werden.

In Kombination mit anderen Methoden und einem soliden wissenschaftlichen Ansatz trägt das AIC maßgeblich dazu bei, aus Daten fundiertes Wissen zu generieren.

Datenqualitätstools über Verteilungstests

12

Innerhalb des Kanons der statistischen Datenqualitätstools nimmt der Kolmogorov-Smirnov-Test (KS-Test), dessen mathematische Grundlagen in Anhang C detailliert erläutert werden, eine besondere Stellung ein.

Als nichtparametrischer Test ist er nicht auf Annahmen über die zugrundeliegende Verteilung der Daten angewiesen, was ihn besonders flexibel einsetzbar macht.

Andrey Kolmogorov (1903-1987) war ein Universalmathematiker, Nikolai Smirnov (1900-1966) ein Spezialist für mathematische Statistik. Ihr gemeinsamer Test ist ein Meilenstein der nichtparametrischen Statistik.

Seine Stärke liegt in der Fähigkeit, die Verteilungsfunktion einer Stichprobe entweder mit einer theoretischen Verteilung (Ein-Stichproben-Test) oder mit der Verteilung einer anderen Stichprobe (Zwei-Stichproben-Test) zu vergleichen.

Dieses Kapitel beleuchtet die vielfältigen und praxisnahen Anwendungsmöglichkeiten des KS-Tests als ein robustes Instrument zur Sicherung und Überwachung der Datenqualität in verschiedenen Domänen.

Obwohl der KS-Test sehr vielseitig ist, besitzt er eine geringere Teststärke (Power) in den "Rändern" (tails) der Verteilung.

12.1 Vergleich zweier Stichproben

In vielen Situationen ist es notwendig zu testen, inwiefern zwei Stichproben von der gleichen Verteilung abstammen. Ein Beispiel ist die Erweiterung eines statistischen Versuchs mit neuen Stichprobendaten. Die neue Stichprobe muss aus der gleichen Grundgesamtheit stammen wie die alte Stichprobe.

Eine weitere Anwendung ist die zeitliche Konsistenz. Hat man jährlich Daten, möchte man meist wissen inwiefern die Daten des aktuellen Jahres zu den Daten des Vorjahres passen.

Wie hier ein Zwei-Stichproben-Kolmogorov-Smirnov-Test (KS-Test) eingesetzt werden kann, soll folgendes Beispiel zeigen.

12.1.1 Praxisbeispiel: Aktuelle und Vorjahresdaten für Kreditrisikodaten

In diesem Beispiel wird der Zwei-Stichproben-Kolmogorov-Smirnov-Test (KS-Test) angewendet, um zu prüfen, ob zwei unabhängige Stichproben aus derselben Verteilung stammen.

Lending Club ist eine US-amerikanischen Online-Kreditplattform. Er war einer der Pioniere im Bereich *Peer-to-Peer-Kredite*

Als Datengrundlage dient der öffentlich verfügbare Datensatz von **Lending Club**. Der Datensatz kann von Kaggle heruntergeladen werden (vgl. [26]).

Der umfassende Datensatz ist in zwei separate Dateien aufgeteilt:

- **accepted_2007_to_2018q4.csv** Enthält Daten zu Krediten, die genehmigt und finanziert wurden. Diese Datei umfasst über **2.2** Millionen Datensätze.
- **rejected_2007_to_2018q4.csv** Enthält Daten zu Kreditanträgen, die von Lending Club abgelehnt wurden. Diese Datei ist mit über **27** Millionen Einträgen erheblich größer.

Zusammen bietet der Datensatz also einen Einblick in fast **30 Millionen** Kreditentscheidungen. Unter `https://www.handbuch-datenqualitaet.de` sind in *accepted_2008.csv* und *accepted_2009.csv* ein Auszug der akzeptierten Kredite für die Jahre 2008 und 2009. Aufgrund der frühen Jahre vom Lending Club sind die Anzahl der Daten noch moderat (2008: 2393 und 2009: 5281 Datensätze)

Ziel ist es, die Verteilung des jährlichen Einkommens von Kreditnehmern aus dem Jahr **2008** mit der entsprechenden Verteilung aus dem Jahr **2009** zu vergleichen.

Die Nullhypothese (H_0) und die Alternativhypothese (H_1) lauten wie folgt:

Das Aufstellen einer klaren Hypothese und eines entsprechenden Signifikanzniveaus ist Grundlage für jeden statistischen Test.

- H_0: Die Stichproben des Einkommens aus den Jahren 2008 und 2009 stammen aus derselben Verteilung.

$$F_{2008}(x) = F_{2009}(x) \quad \text{für alle } x$$

- H_1: Die Stichproben des Einkommens stammen nicht aus derselben Verteilung.

Die durchschnittliche nominelle Einkommenserhöhung in den USA von 2008 auf 2009 war rund 4,14%. Dies muss beim Verteilungsvergleich berücksichtigt werden.

Es soll ein Kolmogorov-Smirnov-Test angewandt werden. Dazu wird folgender Prompt formuliert:

Prompt für Zwei-Stichproben-Einkommen-Test

Das Verzeichnis ist *C:\Daten*. Die Dateien darin *accepted_2008.csv* und *accepted_2009.csv* enthalten in der ersten Zeile die Feldnamen. Die Trennung ist mit “,”(Komma). Erstelle einen {Python-

Quellcode}, der folgendes macht:

1. Lade beide Dateien und wähle das Feld 'annual_inc' (jährliches Einkommen) aus beiden Datensätzen
2. Entferne fehlende Werte und Ausreißer (z.B. annual_inc < 0 oder extreme Werte)
3. Führe einen Zwei-Stichproben-Kolmogorov-Smirnov-Test durch, um zu prüfen, ob die Einkommensverteilungen aus 2008 und 2009 aus derselben Population stammen. Erhöhe dabei die Daten aus 2008 um 4,14%
4. Erstelle ein Histogramm mit beiden Einkommensverteilungen:
   - 2008-Daten: #D8E1F4 mit schwarzer Umrandung, halbtransparent
   - 2009-Daten: Blau mit schwarzer Umrandung, halbtransparent
   - Beide Histogramme überlagert in einem Plot
5. Erstelle einen zweiten Plot mit den kumulativen Verteilungsfunktionen:
   - 2008: #D8E1F4, durchgehende Linie
   - 2009: Blau, gestrichelte Linie
   - Markiere die Stelle der maximalen Abweichung mit einer vertikalen schwarzen Linie
   - Zeige den KS-Statistik-Wert im Plot an
   - Führe den KS-Test durch und gib Teststatistik, p-Wert und Testentscheidung bei $\alpha = 0.05$ im Plot an
6. Bitte verwende nur die Farben Blau, #D8E1F4 und Schwarz
7. Speichere die Histogramme unter Einkommen_Histogramm_-Vergleich.pdf und die kumulativen Verteilungen unter Einkommen_KS_Test.pdf

Prompt 12.1: Prompt für Stichproben Test

Die Ergebnisse sind in folgenden Abbildungen (LLM Claude Sonnet 4):

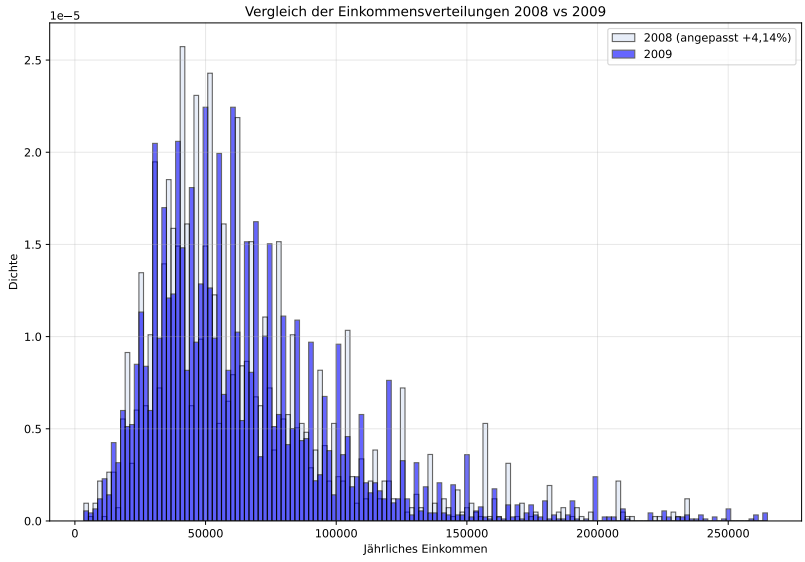


Abbildung 12.1: Histogramm der Einkommensverteilung von 2008 (um 4,14% erhöht) und der Einkommensverteilung von 2009.

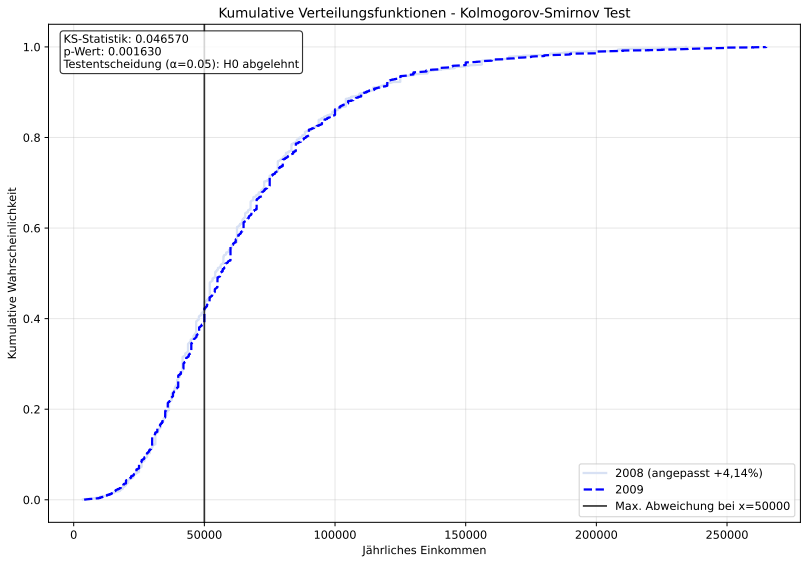


Abbildung 12.2: Der Zwei-Stichproben-Kolmogorov-Smirnov-Test lehnt die Nullhypothese mit $\alpha = 0.05$ ab.

Auf Basis des Zwei-Stichproben-Tests kann die Nullhypothese nicht aufrechterhalten werden. D.h. es kann nicht nachgewiesen werden, dass beide Stichproben von der gleichen Verteilung stammen.

Für überprüfte Daten konnte die Null-Hypothese nicht widerlegt werden, dass die Daten aus derselben Verteilung stammen.

Das Besondere an dem Datensatz ist, dass es Einkommensdaten gibt, die überprüft wurden ('verification_status' ≠ "Not Verified"). Dies sind die verlässlicheren Daten. In diesem Fall wird die H_0-Hypothese mit einem p-Wert von 0.351 nicht abgelehnt, d.h. es kann hier nicht widerlegt werden, dass die überprüften Einkommensdaten aus der gleichen Verteilung stammen.

12.2 Anomalieerkennung mit dem KS-Test

Die Fähigkeit, ungewöhnliche Muster oder Ereignisse in Daten zu erkennen, ist für viele Geschäftsprozesse von Cybersicherheit bis zur Betrugserkennung von entscheidender Bedeutung. Der KS-Test kann als effektives Werkzeug zur Anomalieerkennung dienen, indem er Abweichungen von einem etablierten "Normalzustand" identifiziert.

12.2.1 Ausreißererkennung in Datenbatches

In vielen Szenarien werden Daten in Batches verarbeitet, z.B. tägliche Verkaufstransaktionen oder stündliche Log-Dateien. Um die Qualität dieser eingehenden Daten zu sichern, kann

man die Verteilung eines neuen Batches mit einer historischen Referenzverteilung vergleichen.

Eine **Referenzverteilung** ist die empirische Verteilungsfunktion von Daten aus einem als "normal" oder "stabil" definierten historischen Zeitraum. Sie dient als Benchmark, gegen den neue Daten geprüft werden.

Die Referenzverteilung sollte aus einem langen, stabilen Zeitraum aggregiert werden, um saisonale Schwankungen oder kurzfristige Störungen herauszumitteln und eine robuste "Normalitätsbaseline" zu schaffen.

Der Zwei-Stichproben-KS-Test vergleicht die Verteilung des neuen Daten-Batches (z.B. die Umsätze des letzten Tages) mit der Referenzverteilung (z.B. die aggregierten Umsätze der letzten 90 Tage). Eine signifikante Abweichung kann auf eine Anomalie hindeuten, z.B. einen Systemfehler, eine ungewöhnliche Marktveränderung oder eine Betrugsattacke.

Dieser Ansatz ist besonders mächtig, da er nicht nur einzelne Ausreißer, sondern eine strukturelle Verschiebung der gesamten Verteilung erkennt.

Die Referenzverteilung ist nicht statisch. In sich verändernden Umgebungen (z.B. saisonales Kaufverhalten) muss die Referenzverteilung regelmäßig und vorsichtig aktualisiert werden, um nicht "alte Normalität" als neuen Fehler zu klassifizieren.

12.2.2 Monitoring von Datenströmen

Das Konzept der Anomalieerkennung lässt sich auf kontinuierliche Datenströme (Streaming Data) ausweiten, wie sie im Internet of Things (IoT) von Sensoren oder in Finanzmärkten von Handelssystemen erzeugt werden. Statt auf abgeschlossene Batches wird der KS-Test hier auf gleitenden Zeitfenstern (Sliding Windows) angewendet.

Der Prozess sieht typischerweise so aus, dass die Verteilung der Daten im aktuellsten Zeitfenster (z.B. die letzten 5 Minuten) mit der Verteilung in einem länger zurückliegenden Referenzfenster (z.B. die letzte Stunde) verglichen wird. Ein signifikanter p-Wert aus diesem kontinuierlichen Vergleich deutet auf eine plötzliche Veränderung im Datenstrom hin und kann eine automatische Alarmierung auslösen.

Die Wahl der Fenstergröße ist ein kritischer Trade-off: Zu kleine Fenster sind anfällig für Rauschen, zu große reagieren nur träge auf Veränderungen. Oft werden mehrere Fenstergrößen parallel überwacht.

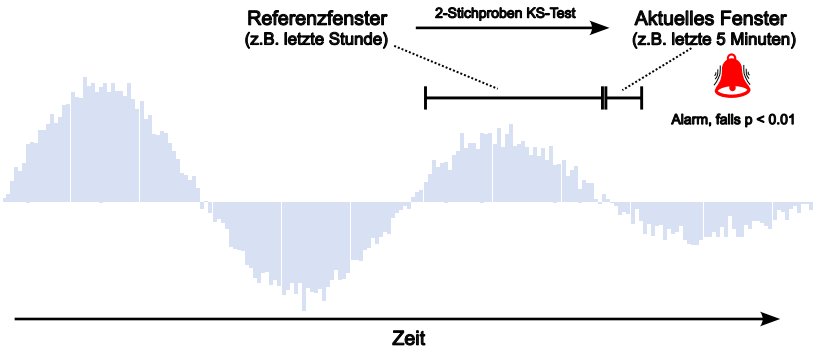


Abbildung 12.3: Konzept des Monitorings von Datenströmen mittels KS-Test auf gleitenden Zeitfenstern zur Erkennung von Verteilungsänderungen in Echtzeit.

12.2.3 Praxisanwendung: DDoS-Angriffsüberwachung

In diesem Beispiel wird der Zwei-Stichproben-Kolmogorov-Smirnov-Test (KS-Test) angewendet, um zu prüfen, ob zwei zeitlich aufeinanderfolgende Netzwerkverkehrs-Zeitfenster aus derselben Verteilung stammen.

DDoS (Distributed Denial of Service): Koordinierter Angriff vieler Systeme, um einen Server durch Überlastung unerreichbar zu machen.

Als Datengrundlage dient der öffentlich verfügbare TSD-DDoS (Time Series Dataset for DDoS) Attack Detection Datensatz vom IEEE DataPort (vgl. [27]). Dieser wurde speziell für die zeitreihenbasierte Erkennung von TCP-basierten Flooding-Angriffen entwickelt.

Die Datenstruktur umfasst folgende Kennzahlen pro Zeitfenster:

Typische DDoS-Indikatoren sind: Hohe #SYN bei niedrigen #SYN-ACK (SYN-Flood), ungewöhnliche #RST-Spitzen (Connection-Reset-Attacken), oder dramatischer Anstieg der #TCP-Pakete pro 5s-Fenster.

- `File number` Datei-Identifikator für verschiedene Erfassungsperioden
- `time period serial number` Fortlaufende Zeitperioden-Nummer (alle 5 Sekunden)
- `#SYN Packets` Anzahl der TCP-SYN-Pakete (Verbindungsanfragen)
- `#SYN-ACK Packets` Anzahl der TCP-SYN-ACK-Pakete (Verbindungsbestätigungen)
- `#ACK Packets` Anzahl der TCP-ACK-Pakete (Bestätigungen)
- `#RST Packets` Anzahl der TCP-RST-Pakete (Verbindungsabbrüche)
- `#TCP Packets` Gesamtanzahl aller TCP-Pakete pro Zeitfenster

Ziel ist es, die Verteilung der TCP-Paket-Anzahl aus einem Referenzfenster (normale Baseline-Periode) mit der entsprechenden Verteilung aus einem aktuellen Testfenster (potenzielle Anomalie-Periode) zu vergleichen.

Die Nullhypothese (H_0) und die Alternativhypothese (H_1) lauten wie folgt:

- H_0: Die Stichproben der TCP-Paket-Anzahl aus Referenz- und Testfenster stammen aus derselben Verteilung (normaler Netzwerkverkehr).

$$F_{\text{Referenz}}(x) = F_{\text{Test}}(x) \quad \text{für alle } x$$

- H_1: Die Stichproben der TCP-Paket-Anzahl stammen nicht aus derselben Verteilung (d.h. potenzielle DDoS-Anomalie).

Bei DDoS-Angriffen ist typischerweise eine dramatische Erhöhung der Paket-Anzahl zu beobachten. Während normaler Netzwerkverkehr oft 50-500 TCP-Pakete pro 5-Sekunden-Fenster aufweist, können DDoS-Angriffe zu 5.000-50.000+ Paketen pro Zeitfenster führen.

Es soll ein gleitender Fenster-Ansatz mit dem Kolmogorov-Smirnov-Test angewandt werden, um kontinuierliche Anomalie-Überwachung zu ermöglichen. Dazu wird folgender Prompt formuliert:

Ein gleitender Fenster-Ansatz ist für die Analyse von Zeitreihendaten einfach umzusetzen. Es können zudem verschiedene Fenster definiert werden, um kurz- und langfristige Anomalien zu erkennen.

Prompt für DDoS-Anomalieerkennung mit Zwei-Stichproben-KS-Test

Das Verzeichnis ist *C:\Daten*. Die Datei darin *DDoS.csv* enthält in der ersten Zeile die Feldnamen. Die Feldtrennung ist ",". Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle das Feld '#TCP Packets'
2. Entferne fehlende Werte und setze negative Werte auf 0.
3. Implementiere eine gleitende Fenster-Anomalieerkennung:
   - Referenzfenster: 100 vergangene Zeitperioden (normale Baseline)
   - Testfenster: 20 aktuelle Zeitperioden (zu prüfende Phase)
   - Führe für jeden möglichen Zeitpunkt einen Zwei-Stichproben-KS-Test zwischen Referenz- und Testfenster durch
   - Markiere Zeitpunkte mit p < 0.001 als Anomalien
4. Erstelle einen dreifach unterteilten Plot:
   - Oberer Plot: Originale Zeitreihe der TCP-Pakete in #D8E1F4, detektierte Anomalien als rote Punkte
   - Mittlerer Plot: KS-Statistik über Zeit in schwarz
   - Unterer Plot: p-Werte in logarithmischer Skalierung (schwarz), Schwellwert p=0.01 als rote gestrichelte Linie, detektierte Anomalien als rote Punkte
5. Speichere die Zeitreihen-Analyse unter DDoS_Anomalie_-Zeitreihe.pdf

Prompt 12.2: Prompt für DDoS-Anomalie

Das Ergebnis (mit LLM Gemini 2.5 pro als Code-Generator) ist in folgender Abbildung:

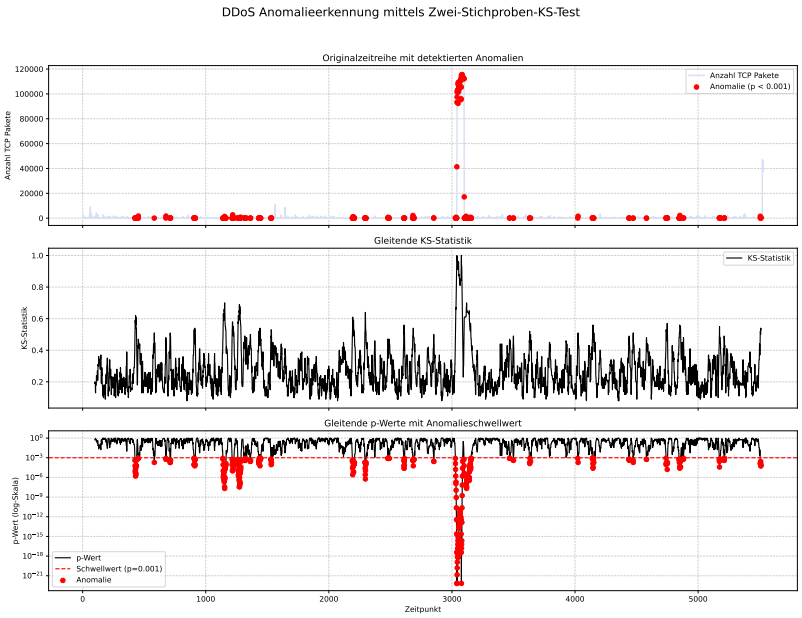


Abbildung 12.4: Gleitende Fenster-Anomalieerkennung mittels KS-Test für DDoS-Detektion. Oberer Plot: TCP-Paket-Anzahl pro 5-Sekunden-Zeitfenster mit detektierten Anomalien (rote Punkte). Mittlerer Plot: KS-Statistik zur Quantifizierung der Verteilungsabweichung zwischen Referenz- und Testfenster. Unterer Plot: p-Werte des KS-Tests (logarithmische Skalierung) mit Signifikanzschwelle $p = 0.001$ zur Anomaliedetektion in Echtzeit.

12.2.4 Fraud Detection

Der KS-Test ist ein Werkzeug der "Anomaly Detection", d.h. er findet Abweichungen vom Normalen. Dies unterscheidet sich von der Erkennung bekannter Betrugsmuster ("Misuse Detection"), für die oft regelbasierte Systeme oder überwachte Modelle verwendet werden.

In der Betrugserkennung, insbesondere im Finanz- und Versicherungssektor, geht es darum, Transaktionen zu identifizieren, die vom normalen Verhalten abweichen. Während komplexe Machine-Learning-Modelle hier oft zum Einsatz kommen, kann der KS-Test eine wertvolle ergänzende Rolle spielen oder als einfaches Baseline-Modell dienen.

Man kann z.B. die Verteilung von Transaktionsbeträgen eines einzelnen Kunden oder Händlers an einem Tag mit dessen historischer Verteilung der letzten 180 Tage vergleichen. Ein Betrüger, der versucht, ungewöhnlich hohe oder viele kleine, untypische Transaktionen durchzuführen, würde die Verteilung signifikant verändern. Dies würde vom KS-Test erkannt werden und eine genauere Prüfung der Transaktionen auslösen. Dies ist besonders nützlich, um Betrugsmuster zu erkennen, die sich nicht in einem einzelnen Ausreißer, sondern in einer veränderten Gesamtdynamik manifestieren.

12.3 Weitere Anwendungsmöglichkeiten des KS-Tests

Der Kolmogorov-Smirnov-Test hat auf Grund seiner Verteilungsunabhängigen Eigenschaften zahlreiche weitere Anwendungsmöglichkeiten. Folgende Tabelle zeigt einige Beispiele:

Anwendungsbereich	Test-Variante	Beschreibung und Zweck
Normalitätstest	Stichproben-KS	Prüfung ob Stichprobe aus Normalverteilung stammt (Parameter μ und σ müssen bekannt sein)
Modellvalidierung	Lilliefors-Test	Prüfung der Normalverteilung von Residuen bei Regressions- und Zeitreihenmodellen (Parameter werden aus Daten geschätzt)
Qualitätskontrolle	Stichproben-KS	Abgleich von Produktionsmerkmalen mit Soll-Verteilung in der Fertigung (Früherkennung von Prozessabweichungen)
ETL-Validierung	2-Stichproben-KS	Sicherstellung der Verteilungsgleichheit zwischen Quell- und Zieldaten nach Extract-Transform-Load-Prozessen
Datenbankmigration	2-Stichproben-KS	Validierung der korrekten Datenübertragung zwischen altem und neuem System (Erkennung von Konvertierungsfehlern)
Zeitreihenbrüche	2-Stichproben-KS	Validierung von Strukturbruchpunkten durch Vergleich der Verteilungen vor und nach dem vermuteten Bruchzeitpunkt
Sensordaten	1/2-Stichproben-KS	Erkennung von Sensordrift oder -defekten durch Vergleich mit Referenzverteilung oder Goldstandard-Sensor (Neukalibrierung erforderlich)
Stichproben	1/2-Stichproben-KS	Prüfung der Repräsentativität von Umfrage- und Forschungsstichproben gegenüber bekannter Grundgesamtheit oder Referenzstichprobe

Der KS-Test ist ein "Schweizer Taschenmesser" der Statistik: Testet Gleichheit ganzer Verteilungen (nicht nur Mittelwerte), funktioniert ohne Verteilungsannahmen, bei kleinen/großen Stichproben, und erkennt subtile Muster, die parametrische Tests übersehen.

12.4 Zusammenfassung

Dieses Kapitel hat die breite Anwendbarkeit des Kolmogorov-Smirnov-Tests als vielseitiges Werkzeug im Arsenal des Datenqualitätsmanagements aufgezeigt. Die Einsatzmöglichkeiten reichen von der fundamentalen Validierung von Verteilungsannahmen, über die Sicherstellung der Konsistenz bei Datenbewegungen zwischen Systemen bis hin zu fortgeschrittenen Anwendungen in der Anomalieerkennung und der Wahrung der Datenintegrität.

Die Stärke des Tests liegt in seiner nichtparametrischen Natur und seiner Fähigkeit, die gesamte Verteilungsform zu vergleichen, was ihn empfindlich für eine Vielzahl von Datenqualitätsproblemen macht.

Trotz seiner Vielseitigkeit besitzt der KS-Test auch Grenzen. Seine Sensitivität ist in der Mitte der Verteilung am höchsten und an den Rändern geringer.

KS-Test-Limitationen: Höchste Sensitivität in Verteilungsmitte, schwächere Erkennung von Tail-Anomalien.

Ferner ist die direkte Anwendung des Ein-Stichproben-Tests ungültig, wenn die Parameter der theoretischen Verteilung

Big-Data-Paradox: Bei $n \to \infty$ wird der KS-Test hypersensitiv und erkennt statistisch signifikante, aber praktisch irrelevante Unterschiede.

aus den Daten geschätzt werden, was den Einsatz von Modifikationen wie dem Lilliefors-Test (vgl. Abschnitt C.3.2) erforderlich macht.

Bei sehr großen Datensätzen kann der Test zudem übermäßig empfindlich werden und statistisch signifikante, aber praktisch irrelevante Abweichungen aufzeigen.

Der Benford-Test: Erkennung manipulierter Daten

13

13.1 Grundlagen und historische Entwicklung

Das Benfordsche Gesetz, auch als Gesetz der ersten Ziffer bekannt, ist eine der faszinierendsten Entdeckungen in der angewandten Mathematik und Statistik. Es beschreibt die natürliche Häufigkeitsverteilung der ersten signifikanten Ziffer in vielen realen Datensätzen und hat sich zu einem Werkzeug für die Aufdeckung von Datenmanipulationen entwickelt.

Historische Entwicklung: 1881 entdeckte der Astronom Simon Newcomb das Phänomen durch abgenutzte Logarithmentabellen-Seiten mit niedrigen Anfangsziffern. Nach 57 Jahren Vergessenheit wiederentdeckte Physiker Frank Benford 1938 das Gesetz und bestätigte es empirisch an über 20.000 Datensätzen aus diversen Bereichen.

Das **Benfordsche Gesetz** besagt, dass in vielen natürlich vorkommenden Zahlendatensätzen die erste signifikante Ziffer d mit der Wahrscheinlichkeit

$$P(d) = \log_{10}\left(1 + \frac{1}{d}\right)$$

auftritt, wobei $d \in \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Diese scheinbar kontraintuitive Verteilung bedeutet, dass die Ziffer 1 mit etwa 30,1% deutlich häufiger als erste Ziffer auftritt als die Ziffer 9 mit nur 4,6%. Die Wahrscheinlichkeiten nehmen logarithmisch ab, was zunächst paradox erscheint, da man intuitiv eine Gleichverteilung erwarten würde.

Tabelle 13.1: Benford-Verteilung der ersten Ziffern

Erste Ziffer d	Wahrscheinlichkeit $P(d)$	Prozent	Erwartete Häufigkeit (bei $n = 1000$)
1	$\log_{10}(2) = 0.301$	30.1%	301
2	$\log_{10}(1.5) = 0.176$	17.6%	176
3	$\log_{10}(1.333) = 0.125$	12.5%	125
4	$\log_{10}(1.25) = 0.097$	9.7%	97
5	$\log_{10}(1.2) = 0.079$	7.9%	79
6	$\log_{10}(1.167) = 0.067$	6.7%	67
7	$\log_{10}(1.143) = 0.058$	5.8%	58
8	$\log_{10}(1.125) = 0.051$	5.1%	51
9	$\log_{10}(1.111) = 0.046$	4.6%	46

Die Ziffer 1 ist sechsmal häufiger als die Ziffer 9. Dieses Verhältnis bleibt konstant, egal welche Einheit man betrachtet - ein Beispiel für die Skaleninvarianz des Gesetzes.

13.2 Theoretische Grundlage

13.2.1 Mathematische Herleitung

Die mathematische Begründung für das Benfordsche Gesetz basiert auf der Annahme der Skaleninvarianz. Wenn ein Datensatz dem Benfordschen Gesetz folgt, sollte er auch nach

einer Multiplikation mit einem konstanten Faktor weiterhin diesem Gesetz folgen.

Ein einfaches Beispiel: Die Bevölkerungszahlen deutscher Städte folgen dem Benford-Gesetz sowohl in der Anzahl der Einwohner als auch wenn man sie in Tausender umrechnet. Das Gesetz ist "skaleninvariant".

Skaleninvarianz bedeutet, dass die Häufigkeitsverteilung der ersten Ziffern unabhängig von der verwendeten Maßeinheit ist. Ein Datensatz, der dem Benfordschen Gesetz in Euro folgt, sollte es auch in Dollar oder Yen tun.

Diese Eigenschaft führt zur logarithmischen Verteilung, da die Logarithmusfunktion die einzige Funktion ist, die diese Skaleninvarianz-Eigenschaft erfüllt. Mathematisch lässt sich zeigen, dass für einen über mehrere Größenordnungen verteilten Datensatz die Wahrscheinlichkeit, dass eine Zahl mit der Ziffer d beginnt, proportional zur Länge des Intervalls $[d, d+1)$ auf der logarithmischen Skala ist.

Die logarithmische Skala erklärt, warum das Benford-Gesetz bei exponentiellen Wachstumsprozessen besonders gut funktioniert - wie bei Bakterienkulturen, Börsenkursen oder Bevölkerungsentwicklungen.

13.2.2 Bedingungen für die Gültigkeit

Das Benfordsche Gesetz gilt nicht universell, sondern nur unter bestimmten Voraussetzungen. Der Datensatz muss über mehrere Größenordnungen verteilt sein, was bedeutet, dass die kleinsten und größten Werte sich um mindestens den Faktor 10 unterscheiden sollten. Die Daten sollten natürlich entstanden und nicht künstlich konstruiert oder begrenzt sein. Besonders wichtig ist, dass die Daten keinen spezifischen Unter- oder Obergrenzen unterliegen, die die natürliche Verteilung verzerren könnten.

Klassische Gegenbeispiele: Körpergrößen von Erwachsenen (zu eng begrenzt), Telefonnummern (künstlich konstruiert) oder IQ-Werte (normiert auf Mittelwert 100).

Zusätzlich muss der Datensatz eine ausreichende Vielfalt aufweisen und darf nicht durch bewusste menschliche Auswahl oder Rundung stark beeinflusst sein. Datensätze mit einer starken zentralen Tendenz oder solche, die auf feste Werte normiert sind (wie Prozentangaben zwischen 0 und 100), folgen typischerweise nicht dem Benfordschen Gesetz.

Folgende Tabelle fasst die Voraussetzungen zusammen:

Tabelle 13.2: Benford-Gesetz Voraussetzungen

Kriterium	Schwellenwert	Bewertung
Stichprobengröße	≥ 1000	Erforderlich
Größenordnungen	≥ 3	Erforderlich
Max/Min Verhältnis	≥ 1000	Empfohlen
Schiefe (betragsmäßig)	> 0.3	Empfohlen
Datentyp	natürlich entstanden	Erforderlich
Künstliche Grenzen	Keine	Erforderlich

13.3 Der Benford-Test

13.3.1 Testdurchführung

Der Benford-Test prüft, ob ein gegebener Datensatz der erwarteten Benford-Verteilung folgt. Dies geschieht typischerweise durch einen Chi-Quadrat-Anpassungstest, der die beobachteten Häufigkeiten der ersten Ziffern mit den theoretisch erwarteten Häufigkeiten vergleicht.

Der erste dokumentierte forensische Einsatz erfolgte 1992 durch Mark Nigrini zur Analyse verdächtiger Steuererklärungen. Seitdem wird der Test routinemäßig von Finanzaufsichtsbehörden eingesetzt.

Die **Benford-Teststatistik** ist definiert als:

$$\chi^2 = \sum_{d=1}^{9} \frac{(O_d - E_d)^2}{E_d}$$

wobei O_d die beobachtete Häufigkeit der ersten Ziffer d und $E_d = n \cdot P(d)$ die erwartete Häufigkeit unter der Benford-Verteilung darstellt.

Unter der Nullhypothese, dass die Daten der Benford-Verteilung folgen, ist diese Teststatistik asymptotisch chi-quadrat-verteilt mit $\nu = 8$ Freiheitsgraden (da es 9 Kategorien gibt, aber die Gesamtsumme fixiert ist, bleiben 8 freie Parameter).

13.3.2 Hypothesen und Interpretation

Bei sehr großen Datensätzen (n > 100.000) kann der Chi-Quadrat-Test überempfindlich werden und auch praktisch unbedeutende Abweichungen als statistisch signifikant identifizieren.

Beim **Benford-Test** werden folgende Hypothesen geprüft:

H_0 : Die ersten Ziffern folgen der Benford-Verteilung (13.1)

H_1 : Die ersten Ziffern folgen nicht der Benford-Verteilung (13.2)

Eine Ablehnung der Nullhypothese kann verschiedene Ursachen haben. Sie kann auf natürliche Abweichungen hindeuten, wenn der Datensatz die Voraussetzungen für das Benfordsche Gesetz nicht erfüllt. Häufiger jedoch deutet eine signifikante Abweichung auf menschliche Manipulation, bewusste Fälschung oder systematische Verzerrungen in der Datensammlung hin.

Berühmter Fall: 2009 wurde das Benford-Gesetz zur Analyse der iranischen Wahlergebnisse eingesetzt. Die Daten zeigten signifikante Abweichungen, was Manipulationsverdacht stützte.

13.3.3 Anwendungsbeispiel

Die nachfolgend verwendeten Daten stammen aus dem sehr empfehlenswerten Buch [28]. Die Daten können unter `https://github.com/carloscinelli/benford.analysis/blob/master/data/corporate.payment.rda` heruntergeladen werden. Sie wurden im vorliegenden Beispiel über R in ein csv konvertiert (zu finden auf `www.handbuch-datenqualitaet.de`). Es sind

189.470 Datensätze von Unternehmenszahlungen, die auf das Benford-Gesetz getestet werden sollen.

Diese Daten stammen aus einem realen Betrugsfall eines US-amerikanischen Unternehmens. Sie enthalten sowohl legitime als auch manipulierte Zahlungen - ein idealer Testfall für Benford-Analysen.

Zunächst sollen die Voraussetzungen überprüft werden:

Prompt für Benford-Gesetz-Voraussetzungs-Prüfung

Das Verzeichnis ist *C:\Daten*. Die Datei *corporate_payment.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ",". Erstelle einen {Python-Quellcode}, der folgendes macht:

1. Lade die Datei und wähle die Spalte 'Amount' aus
2. Entferne fehlende Werte, Nullwerte und negative Zahlen
3. Berechne folgende Kennzahlen:
   - Stichprobengröße nach Bereinigung
   - Anzahl der Größenordnungen: $\log_{10}(\text{max}) - \log_{10}(\text{min})$
   - Schiefe der Verteilung (betragsmäßig)
   - Verhältnis Maximum zu Minimum
4. Gib in der Konsole eine formatierte Tabelle aus

Prompt 13.1: Prompt für Test der Benford Voraussetzungen

Das Ergebnis der Konsoleausgabe (LLM ist Claude Sonnet 4) ist:

Tabelle 13.3: Benford-Gesetz Voraussetzungsprüfung

Kennzahl	Wert
Stichprobengröße	185 083
Anzahl Größenordnungen	9.43
Schiefe (betragsmäßig)	223.2
Verhältnis Max/Min	2 676 347 578

Mit 9,43 Größenordnungen übertrifft dieser Datensatz die Mindestanforderung von 3 Größenordnungen deutlich. Die extreme Schiefe von 223,2 ist typisch für Finanzdaten mit wenigen sehr hohen Ausreißern.

Die Voraussetzungen sind damit erfüllt.

Nun kann der Benford-Test auf die erste Ziffer mit Konfidenz α vorgenommen werden. Dazu benutzt man folgenden Prompt:

Prompt für Benford-Gesetz-Analyse

Das Verzeichnis ist *C:\Daten*. Die Datei *corporate_payment.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ",". Erstelle einen {Python-Quellcode}, der folgendes macht:

1. Lade die Datei und wähle die Spalte 'Amount'
2. Entferne fehlende Werte, Nullwerte und negative Beträge aus den Zahlungsdaten
3. Führe einen Chi-Quadrat-Anpassungstest ($\alpha = 0.05$)durch, um zu prüfen, ob die beobachteten Häufigkeiten der ersten Ziffer der Benford-Verteilung folgen
4. Erstelle ein Balkendiagramm mit:
   - Beobachtete Häufigkeiten: #D8E1F4 mit schwarzer

Umrandung als Balken

- Theoretische Benford-Häufigkeiten: Blaue durchgezogene Linie

5. Füge eine Textbox in der oberen rechten Ecke des Diagramms ein mit den Ergebnissen des Chi-Quadrat-Anpassungstest durch.
6. Speichere die Benford-Analyse unter Benford_Analyse.pdf

Prompt 13.2: Prompt für DBenford-Analyse

Das Python-Script aus dem Prompt über ChatGPT 5 liefert folgende Darstellung:

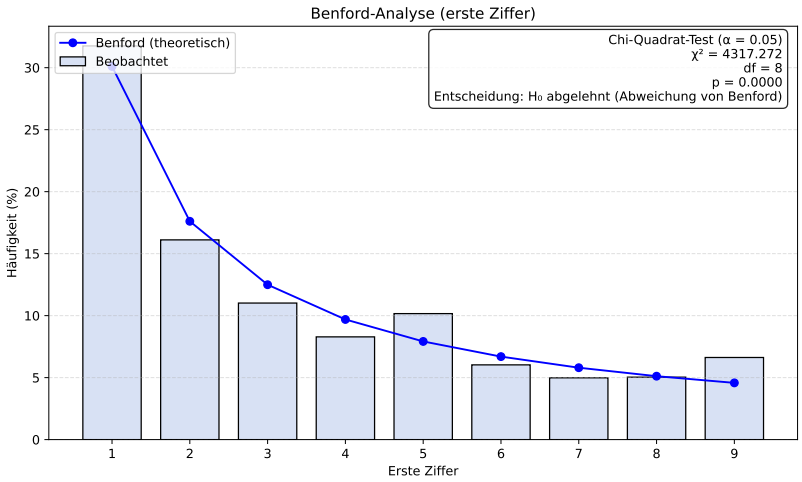


Abbildung 13.1: Histogramm der ersten Ziffer aller 189.470 Zahlungsdaten. Die Chi-Quadrat-Test zeigt eine Abweichung vom Benford-Gesetz.

13.4 Erweiterte Benford-Tests

13.4.1 Test der zweiten Ziffer

Neben dem klassischen Test der ersten Ziffer kann auch die Verteilung der zweiten signifikanten Ziffer analysiert werden. Diese folgt einer anderen, aber ebenfalls vorhersagbaren Verteilung.

Der Test der zweiten Ziffer ist oft sensitiver für Manipulationen, da Betrüger typischerweise nur auf die erste Ziffer achten und die zweite vernachlässigen.

Die **Wahrscheinlichkeit der zweiten Ziffer** s bei gegebener erster Ziffer d ist:

$$P(s|d) = \log_{10}\left(1 + \frac{1}{10d + s}\right)$$

wobei $s \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ und $d \in \{1, 2, \ldots, 9\}$.

Die Analyse der zweiten Ziffer kann zusätzliche Einblicke liefern, da Manipulateure oft nur die erste Ziffer im Blick haben und die zweite Ziffer vernachlässigen. Dies kann zu charakteristischen Mustern führen, die durch den erweiterten Test aufgedeckt werden.

Praxis-Tipp: Bei der zweiten Ziffer ist die Ziffer 0 am häufigsten (etwa 12%), während 9 am seltensten auftritt (etwa 8,5%). Diese Verteilung ist flacher als bei der ersten Ziffer.

13.4.2 Kombination mehrerer Ziffernpositionen

Für eine umfassende Analyse können die ersten beiden Ziffern gemeinsam betrachtet werden. Die Wahrscheinlichkeit für eine Zweistellenkombination d_1d_2 ist gegeben durch:

$$P(d_1d_2) = \log_{10}\left(1 + \frac{1}{10d_1 + d_2}\right)$$

Diese Erweiterung erhöht die Sensitivität des Tests, da nun 90 verschiedene Kombinationen (10 bis 99) analysiert werden können, was eine feinere Differenzierung zwischen natürlichen und manipulierten Daten ermöglicht.

Die häufigste Zweistellenkombination ist "10"mit 4,14%, die seltenste ist "99"mit nur 0,44% - ein Verhältnis von fast 10:1.

13.4.3 Test der letzten beiden Ziffern

Ein besonders mächtiges Werkzeug zur Aufdeckung von Manipulationen ist der **Last-Two-Digits Test**, der die Verteilung der letzten beiden Ziffern analysiert. Im Gegensatz zu den ersten Ziffern, die dem Benfordschen Gesetz folgen, sollten die letzten beiden Ziffern bei natürlichen Daten gleichverteilt sein.

Dieser Test ist besonders effektiv bei der Aufdeckung von "psychologischen"Preisen wie 9,99€ oder runden Beträgen wie 50,00€, die Menschen unbewusst bevorzugen.

Der **Last-Two-Digits Test** prüft die Nullhypothese, dass die letzten beiden Ziffern einer gleichmäßigen Verteilung folgen. Bei natürlichen Daten sollte jede Kombination von 00 bis 99 mit der Wahrscheinlichkeit $P = \frac{1}{100} = 0.01$ auftreten.

Die Teststatistik folgt der gleichen Chi-Quadrat-Struktur:

$$\chi^2 = \sum_{i=0}^{99} \frac{(O_i - E_i)^2}{E_i}$$

wobei $E_i = \frac{n}{100}$ für alle i und die Teststatistik asymptotisch χ^2_{99}-verteilt ist.

Berühmtes Beispiel: Bei der Analyse der griechischen Staatsdefizit-Zahlen 2010 zeigten sich verdächtige Häufungen bei runden Endziffern, was den Manipulationsverdacht untermauerte.

Wichtig: Besondere Sensitivität des Last-Two-Digits Tests

Der Test der letzten beiden Ziffern ist besonders sensitiv für menschliche Manipulationen, da Menschen dazu neigen, "rundeSZahlen zu verwenden (Endungen auf 00, 50) oder bestimmte Ziffernkombinationen zu bevorzugen bzw. zu vermeiden. Natürliche Prozesse zeigen hingegen typischerweise eine sehr gleichmäßige Verteilung der Endziffern.

Typische Manipulationsmuster zeigen sich durch: - Überrepräsentation von runden Zahlen (00, 25, 50, 75) - Vermeidung

bestimmter ünschöner"Kombinationen - Clustering um bestimmte bevorzugte Endungen - Auffällige Lücken in der Verteilung

Kulturelle Besonderheit: In manchen asiatischen Ländern wird die Zahl 4 vermieden (klingt wie "Tod"), was sich in Preisstrukturen und damit in Datenanalysen niederschlägt.

13.4.4 Kombinierter Ansatz: Vollständige Ziffernanalyse

Die höchste Sensitivität erreicht man durch die Kombination aller Zifferntests. Ein vollständiger Benford-Test umfasst:

1. Test der ersten Ziffer (Benford-Verteilung)
2. Test der zweiten Ziffer (bedingte Benford-Verteilung)
3. Test der ersten beiden Ziffern (kombinierte Benford-Verteilung)
4. Test der letzten beiden Ziffern (Gleichverteilung)

Der gesamte Benford-Test kann so umgesetzt werden

Forensische Praxis: Professionelle Wirtschaftsprüfer verwenden meist alle vier Tests gleichzeitig. Erst wenn alle Tests unauffällig sind, gilt ein Datensatz als ünverdächtig".

Prompt für Benford-Gesamtanalyse

Das Verzeichnis ist *C:\Daten*. Die Datei *corporate_payment.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ",". Erstelle einen {Python-Quellcode}, der folgendes macht:

1. Lade die Datei und wähle die Spalte 'Amount'
2. Entferne fehlende Werte, Nullwerte und negative Beträge aus den Zahlungsdaten
3. Führe jeweils einen Chi-Quadrat-Anpassungstest ($\alpha = 0.05$) durch, um zu prüfen, ob die beobachteten Häufigkeiten der ersten Ziffer, der zweiten Ziffer und der beiden ersten Ziffern der Benford-Verteilung folgen
4. Teste, ob die beiden letzten Ziffern einer Gleichverteilung folgen.
5. Erstelle jeweils ein Balkendiagramm mit:
   - Beobachtete Häufigkeiten: #D8E1F4 mit schwarzer Umrandung als Balken
   - Theoretische Benford-Häufigkeiten: Blaue durchgezogene Linie
6. Füge jeweils eine Textbox in der oberen rechten Ecke der Diagramme ein mit den Ergebnissen der statistischen Tests. Ansonsten keine Legende.
7. Speichere die Grafiken als 2x2 Grafik in Benford_Analyse_-Gesamt.pdf

Prompt 13.3: Prompt für Benford-Gesamtanalyse

Das Ergebnis ist in folgender Abbildung zusammengefasst:

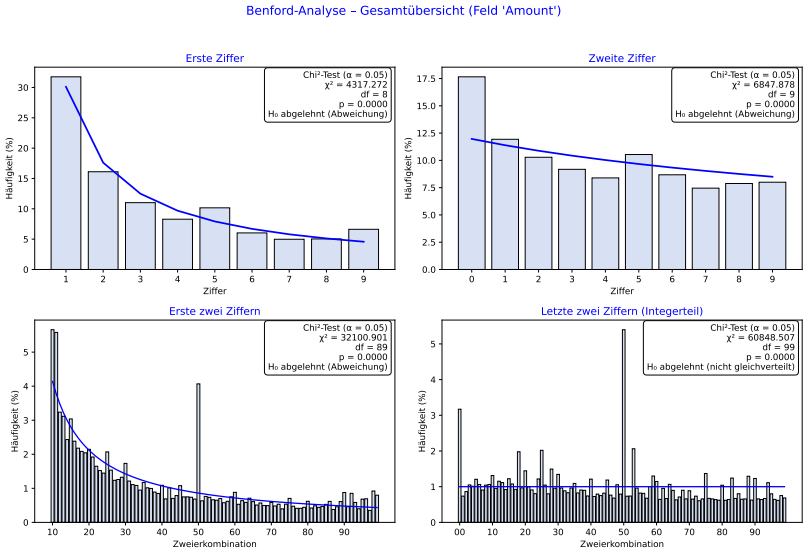


Abbildung 13.2: Testergebnis der 189.470 Zahlungsdaten aus 1. Ziffer, 2. Ziffer, erste beiden Ziffern und den letzten beiden Ziffern.

Die Analyse zeigt, dass es einen Manipulationsverdacht für die Daten gibt. Insbesondere die zweier Kombination “50“ bei den ersten beiden Ziffern ist ein großer Ausreißer, den man sich näher betrachten sollte.

Der Ausreißer bei "50"deutet auf gefälschte Rechnungen hin, die bewusst mit runden 50er-Beträgen erstellt wurden - ein typisches Muster bei Abrechnungsbetrug.

13.5 Grenzen und kritische Betrachtung

13.5.1 Falsch-positive Ergebnisse

Ein wesentliches Problem des Benford-Tests ist die Möglichkeit falsch-positiver Ergebnisse. Nicht alle Abweichungen von der Benford-Verteilung sind auf Manipulation zurückzuführen. Natürliche Prozesse können durchaus zu Datensätzen führen, die nicht dem Benfordschen Gesetz folgen.

Beispiele für legitime Abweichungen sind Datensätze mit natürlichen Grenzen (wie Körpergrößen oder Temperaturen), Daten aus kontrollierten Prozessen (wie Qualitätsmessungen in der Fertigung), oder Zahlen, die durch menschliche Konventionen geprägt sind (wie Preise, die häufig auf 99 Cent enden).

13.5.2 Sophistizierte Manipulationen

Paradox der Forensik: Je bekannter der Benford-Test wird, desto weniger effektiv wird er gegen informierte Betrüger. Daher setzen Forensiker heute auf Kombination mit anderen, weniger bekannten Methoden.

Ein weiteres Problem ergibt sich, wenn Manipulateure über Kenntnisse des Benfordschen Gesetzes verfügen. Erfahrene Betrüger können ihre gefälschten Zahlen so gestalten, dass sie der Benford-Verteilung folgen, wodurch der Test seine Wirksamkeit verliert.

Diese Limitation macht deutlich, dass der Benford-Test nie als alleiniges Beweismittel dienen sollte, sondern immer als Teil eines umfassenderen Analyseprozesses. Er ist am effektivsten als Screening-Instrument, das verdächtige Datensätze identifiziert, die dann einer detaillierteren Untersuchung unterzogen werden.

Forensische Strategie: Moderne Betrugsjäger verwenden den Benford-Test primär als ersten Filter. Verdächtige Datensätze werden dann mit Methoden aus der künstlichen Intelligenz und Machine Learning weiter analysiert.

13.6 Statistische Verfeinerungen

13.6.1 Alternative Teststatistiken

Neben dem Chi-Quadrat-Test können andere statistische Verfahren für den Benford-Test verwendet werden. Der Kolmogorov-Smirnov-Test prüft die Übereinstimmung der kumulativen Verteilungsfunktionen und ist weniger sensitiv gegenüber Ausreißern in einzelnen Kategorien.

Der Kolmogorov-Smirnov-Test ist besonders nützlich bei kleineren Stichproben (n < 1000), wo der Chi-Quadrat-Test unzuverlässig wird.

Für den **Kolmogorov-Smirnov-Benford-Test** wird die maximale Abweichung zwischen der empirischen und theoretischen kumulativen Verteilungsfunktion berechnet:

$$D = \max_{d \in \{1,\ldots,9\}} |F_{emp}(d) - F_{Benford}(d)|$$

Der Mean Absolute Deviation (MAD) Test bietet eine weitere Alternative, die robust gegenüber extremen Abweichungen ist und eine intuitive Interpretation der Ergebnisse ermöglicht.

13.6.2 Bayesianische Ansätze

MAD-Werte über 0,015 gelten als verdächtig, über 0,022 als höchst verdächtig. Diese Schwellenwerte wurden empirisch aus Tausenden von Datensätzen abgeleitet.

Moderne statistische Verfahren nutzen bayesianische Methoden, um A-priori-Wissen über die Wahrscheinlichkeit von Manipulationen in die Analyse einzubeziehen. Diese Ansätze können die Teststärke erheblich verbessern, insbesondere wenn zusätzliche Informationen über den Kontext der Datenerhebung verfügbar sind.

Bayesianische Verfahren können branchenspezifisches Wissen integrieren: Ein Datensatz aus dem Baugewerbe (hohe Betrugsrate) wird anders bewertet als einer aus dem Gesundheitswesen.

Bayesianische Verfahren ermöglichen es auch, die Unsicherheit in den Schätzungen explizit zu modellieren und probabilistische Aussagen über die Wahrscheinlichkeit einer Manipulation zu treffen, anstatt nur binäre Entscheidungen (Manipulation ja/nein) zu liefern.

13.7 Internationale Anwendungen und rechtliche Aspekte

13.7.1 Verwendung in Gerichtsverfahren

Historischer Meilenstein: Mark Nigrini führte 1992 den ersten forensischen Einsatz des Benford-Tests zur Analyse verdächtiger Steuererklärungen durch. Sein wegweisender Artikel von 1999 etablierte das Verfahren in der forensischen Buchhaltung.

Der Benford-Test hat in verschiedenen Ländern Eingang in Gerichtsverfahren gefunden, wobei seine rechtliche Anerkennung unterschiedlich ist. In den USA wird er regelmäßig als Beweismittel in Betrugs- und Steuerhinterziehungsverfahren akzeptiert, allerdings nicht als alleiniger Beweis, sondern als unterstützendes Indiz.

Die rechtliche Bewertung hängt stark von der sachgerechten Anwendung und der angemessenen Interpretation der Ergebnisse ab. Gerichte verlangen typischerweise, dass die Anwendungsvoraussetzungen des Tests erfüllt sind und dass alternative Erklärungen für die beobachteten Abweichungen ausgeschlossen wurden.

13.7.2 Standardisierung und Richtlinien

Deutsche Anwendung: Das Benford-Gesetz wird seit Jahren von der deutschen Finanzverwaltung, Steuerfahndern und Wirtschaftsprüfern als Screening-Instrument zur Aufdeckung von Unregelmäßigkeiten im Rechnungswesen eingesetzt.

Verschiedene Berufsverbände und Aufsichtsbehörden haben Richtlinien für die Anwendung des Benford-Tests entwickelt. Diese Standards definieren Mindestanforderungen für Stichprobengröße, Datenqualität und Interpretationsverfahren.

Die Association of Certified Fraud Examiners (ACFE) empfiehlt den Benford-Test als Standard-Tool in ihrer "Fraud Examiner's Manualünd bietet spezielle Zertifizierungskurse an [29].

Die Association of Certified Fraud Examiners (ACFE) und das American Institute of Certified Public Accountants (AICPA) haben detaillierte Leitfäden veröffentlicht, die praktische Empfehlungen für die forensische Anwendung des Tests enthalten [29, 30].

13.8 Zusammenfassung

Internationale Standards: Die ISO 37001 (Anti-Bribery Management Systems) fordert robuste finanzielle Kontrollen und Überwachungsverfahren, die digitale forensische Methoden wie den Benford-Test einschließen können [31].

Dieses Kapitel hat die vielfältigen Einsatzmöglichkeiten des Benford-Tests als mächtiges Instrument zur Erkennung manipulierter Daten aufgezeigt. Die Anwendungen reichen von der grundlegenden Validierung natürlicher Zahlenverteilungen (siehe Abschnitt 13.1) über die systematische Testdurchführung mit verschiedenen statistischen Verfahren (Abschnitt 13.3) bis hin zu erweiterten Analysen mehrerer Ziffernpositionen (Abschnitt 13.4) und forensischen Anwendungen (Abschnitt 13.7). Die Stärke des Tests liegt in seiner

mathematisch fundierten Basis auf dem Prinzip der Skaleninvarianz und seiner Fähigkeit, große Datensätze effizient zu screenen und dabei charakteristische Manipulationsmuster aufzudecken.

Zukunftstrend: Maschinelles Lernen wird zunehmend mit Benford-Tests kombiniert. Algorithmen lernen aus historischen Betrugsmustern und können so noch subtilere Manipulationen erkennen.

Trotz seiner bewährten Wirksamkeit besitzt der Benford-Test auch deutliche Grenzen. Seine Anwendbarkeit ist an spezifische Voraussetzungen geknüpft - Datensätze müssen über mehrere Größenordnungen verteilt sein und natürlichen Ursprungs sein, was seine Einsetzbarkeit einschränkt (siehe Tabelle 13.2). Ferner können falsch-positive Ergebnisse auftreten, wenn die Datenstruktur natürlicherweise von der Benford-Verteilung abweicht, und sophistizierte Manipulateure mit Kenntnissen des Gesetzes können den Test durch gezielte Anpassung ihrer gefälschten Daten umgehen (Abschnitt 13.5). Bei sehr großen Datensätzen kann der Test zudem übermäßig empfindlich werden und statistisch signifikante, aber praktisch irrelevante Abweichungen identifizieren.

Praktischer Tipp: Bei Datensätzen mit mehr als 1 Million Einträgen sollten strengere Signifikanzniveaus ($\alpha = 0.001$ statt 0.05) verwendet werden, um falsch-positive Ergebnisse zu vermeiden.

Die Zukunft der forensischen Datenanalyse liegt in der intelligenten Kombination des Benford-Tests mit anderen statistischen und inhaltlichen Prüfverfahren. Der Test sollte nie als alleiniges Beweismittel dienen, sondern als erstes Screening-Instrument in einem mehrstufigen Analyseprozess verstanden werden. Seine Ergebnisse können durch domänenspezifische Plausibilitätsprüfungen, detaillierte Ausreißeranalysen und alternative Teststatistiken wie den Kolmogorov-Smirnov-Test oder bayesianische Ansätze ergänzt werden (Abschnitt 13.6). Die Automatisierung von Benford-Tests in Überwachungssystemen ermöglicht eine kontinuierliche Kontrolle großer Datenströme und die frühzeitige Erkennung potenzieller Manipulationen.

Teil III: Outlier-Analyse multivariater Daten

14 Mahalanobis-Distanz für multivariate Ausreißer

Was für univariate Daten der Z-Score ist, ist für multivariate Daten die Mahalanobis-Distanz. Sie baut auf dem Mittelwertvektor und der Kovarianzmatrix der Daten auf.

Der **Mittelwertvektor** $\mu = (\mu_1, \mu_2, \ldots, \mu_p)^T$ definiert das Zentrum oder den Schwerpunkt der multivariaten Datenverteilung. Jeder Eintrag μ_i ist der Mittelwert der i-ten Variable x_i.

Die **Kovarianzmatrix** Σ ist eine $p \times p$ Matrix, die die Varianzen und Kovarianzen der Variablen beschreibt.

$$\Sigma = \begin{pmatrix} \mathrm{Var}(x_1) & \mathrm{Cov}(x_1, x_2) & \cdots & \mathrm{Cov}(x_1, x_p) \\ \mathrm{Cov}(x_2, x_1) & \mathrm{Var}(x_2) & \cdots & \mathrm{Cov}(x_2, x_p) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}(x_p, x_1) & \mathrm{Cov}(x_p, x_2) & \cdots & \mathrm{Var}(x_p) \end{pmatrix}$$

Wegen $\mathrm{Cov}(x_i, x_j) = \mathrm{Cov}(x_j, x_i)$ ist Σ symmetrisch. Falls keine perfekte lineare Abhängigkeit zwischen Variablen bestehen ist sie damit invertierbar.

Auf der Diagonalen stehen die Varianzen der einzelnen Variablen, die deren Streuung beschreiben. Die Nicht-Diagonal-Elemente sind die Kovarianzen, die die Richtung und Stärke des linearen Zusammenhangs zwischen zwei Variablen angeben.

14.1 Definition der Mahalanobis-Distanz

Die Mahalanobis-Distanz ist wie folgt definiert:

Für einen p-dimensionalen Datenvektor $x = (x_1, x_2, \ldots, x_p)^T$ wird die **Mahalanobis-Distanz** von x zum Zentrum der Daten, das durch den Mittelwertvektor μ repräsentiert wird, wie folgt definiert:

$$D_M(x) = \sqrt{(x - \mu)^T \Sigma^{-1} (x - \mu)} \tag{14.1}$$

Dabei ist Σ die Kovarianzmatrix der Daten und Σ^{-1} ihre Inverse.

Der Ausdruck $(x - \mu)$ ist der Vektor der Abweichungen jeder Variable von ihrem Mittelwert. Das Herzstück ist die inverse Kovarianzmatrix Σ^{-1}. Sie transformiert den Datenraum so, dass die Korrelationen zwischen den Variablen eliminiert und die Varianzen auf 1 normiert werden. In diesem

Im eindimensionalen Fall ($p = 1$) reduziert sich die Kovarianzmatrix auf die Varianz σ^2 und die Mahalanobis-Distanz wird zum absoluten Z-Score: $D_M(x) = \sqrt{\frac{(x-\mu)^2}{\sigma^2}} = \frac{|x-\mu|}{\sigma}$.

transformierten Raum wird dann die euklidische Distanz berechnet.

Die Mahalanobis-Distanz ist also im Grunde eine euklidische Distanz in einem "entzerrten" Datenraum.

Sind zwei Variablen stark positiv korreliert, führt eine Abweichung vom Mittelwert in die gleiche Richtung bei beiden Variablen zu einer geringeren Distanz, als wenn sie unkorreliert wären.

Die Kovarianzmatrix muss invertierbar sein. Dies ist nicht der Fall, wenn eine Variable eine perfekte lineare Kombination anderer Variablen ist (Multikollinearität) oder der Stichprobenumfang kleiner als die Anzahl der Variablen ist.

14.1.1 Vergleich zur euklidischen Distanz

Der fundamentale Unterschied zwischen der Mahalanobis- und der euklidischen Distanz lässt sich am besten visuell und anhand eines Beispiels mit korrelierten Variablen verdeutlichen. Die euklidische Distanz behandelt alle Dimensionen als unabhängig und gleich gewichtet, während die Mahalanobis-Distanz die Geometrie der Datenverteilung erfasst.

Die euklidische Distanz ist ein Spezialfall der Mahalanobis-Distanz. Wenn die Kovarianzmatrix die Identitätsmatrix ist ($\Sigma = I$), was bedeutet, dass alle Variablen unkorreliert sind und eine Varianz von 1 haben, dann sind beide Distanzen identisch.

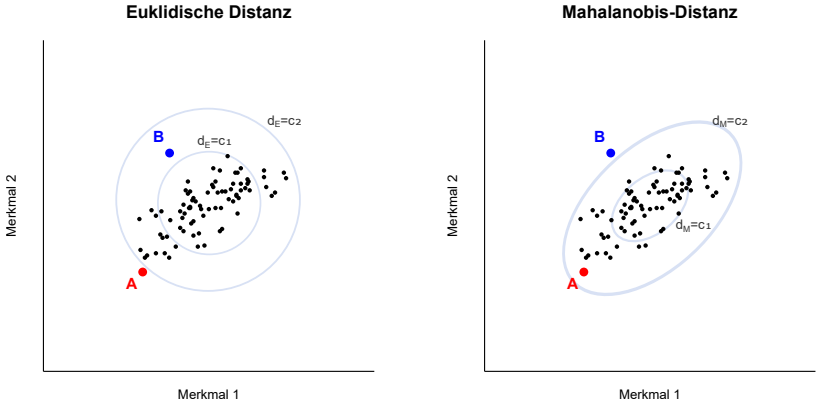


Abbildung 14.1: Vergleich von Linien gleicher euklidischer Distanz (links, Kreise) und gleicher Mahalanobis-Distanz (rechts, Ellipsen) für korrelierte Daten. Punkt A wird von der euklidischen Distanz als Ausreißer bewertet, nicht aber von der Mahalanobis-Distanz. Für Punkt B gilt das Gegenteil. Die Mahalanobis-Distanz passt sich der Form der Daten an.

Die Mahalanobis-Distanz ist daher immer dann sinnvoller als die euklidische Distanz, wenn mindestens eine der folgenden Bedingungen erfüllt ist:

- Die Variablen sind miteinander korreliert.
- Die Variablen haben unterschiedliche Varianzen (unterschiedliche Skalen).

Sie liefert eine robustere und statistisch aussagekräftigere Messung der Abweichung eines Datenpunktes in einem multivariaten Kontext.

14.1.2 Chi-Quadrat-Verteilung

Falls die Daten einer p-dimensionalen Normalverteilung folgen, dann folgt die **quadrierte Mahalanobis-Distanz** zusätzlich einer Chi-Quadrat-Verteilung (χ^2) mit p Freiheitsgraden. Dabei ist p die Dimension des Datenvektors

$$D_M^2(x) \sim \chi_p^2$$

Die Mahalanobis-Distanz kann für alle Verteilungstypen angewandt werden.Die statistische Testbarkeit über χ^2 gilt aber nur bei multivariat normalverteilten Daten.

Damit kann ein statistisch fundierter Schwellenwert für die Klassifizierung von Ausreißern festgelegt werden. Beispielsweise können alle Datenpunkte, deren quadrierte Mahalanobis-Distanz größer ist als das 99%-Quantil der χ_p^2-Verteilung, als Ausreißer betrachtet werden.

Streng genommen gilt dieser Zusammenhang nur, wenn Mittelwert μ und Kovarianzmatrix Σ die wahren Parameter der Grundgesamtheit sind. In der Praxis werden sie aus der Stichprobe geschätzt, was eine zusätzliche Unsicherheit einbringt. Bei großen Stichproben ist dieser Effekt vernachlässigbar.

Normalverteilungsannahme eher die Ausnahme

Im Allgemeinen ist eine p-dimensionalen Normalverteilung von realen Daten eher die Ausnahme als die Regel. In der Praxis werden daher oft nur die Mahalanobis-Distanzen berechnet und die Outlier geordnet von der größten zur kleinsten Distanz betrachtet (z.B. die ersten 100).

14.1.3 Praxisbeispiel: Kardiodaten

Es wird der bereinigte Datensatz des Cardiovascular Disease Dataset von *Kaggle* [9] benutzt. Er ist unter *cardio_train_bereinigt.csv* auf `www.handbuch-datenqualitaet.de` zu finden. Zur vorgenommen Bereinigung vergleiche Seite 92 im Abschnitt 8.9.1.

Prompt für Outlier-Erkennung mit Mahalanobis-Distanz

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten 'height', 'weight' und 'ap_hi' für die Outlier-Analyse aus
2. Berechne die Mahalanobis-Distanz für jeden Datenpunkt basierend auf den drei ausgewählten Variablen
3. Erstelle eine dreidimensionale Grafik mit den Datenpunkten im Farbcode #D8E1F4 und markiere 100 Werte mit der größten Mahalanobis-Distanz blau.
4. Erstelle zudem für jede Spalten 'height', 'weight' und 'ap_hi' eine Dichte der Verteilung und lege ein Histogramm der 100 Outlier-Daten darüber.
5. speichere diese 4 Grafiken in einer 2x2-Grafik unter "cardio_-Mahalanobis.pdf"

Prompt 14.1: Prompt für Outlierfindung mit Mahalanobis-Distanz

Das Ausführen des mit Claude Sonnet 4 erstellten Python-Script gibt folgendes Bild aus:

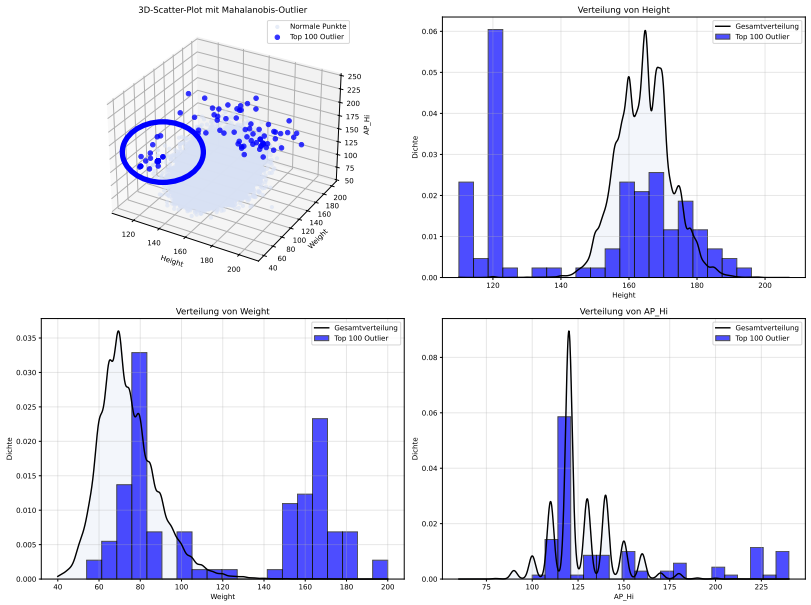


Abbildung 14.2: Die Outliers bilden nicht einfach die Extremwerte der einzelnen Achsen ab, sondern repräsentieren Datenpunkte, die in ihrer Gesamtkombination ungewöhnlich sind - das ist der große Vorteil der multivariaten Betrachtung. Die markierten Outlier sind von Menschen mit kleiner Körpergröße und relativ hohem Gewicht. Dies sieht man auch bei der Histogrammdarstellung der Körpergröße. Hier sind überproportional viele Ausreißer mit kleiner Körpergröße.

14.2 Praktische Anwendung in der Datenqualität

Die theoretischen Vorteile der Mahalanobis-Distanz führen zu vielfältigen Anwendungsmöglichkeiten im Bereich der Datenqualität, die über eine einfache Ausreißererkennung hinausgehen.

14.2.1 Ausreißererkennung

Dies ist die primäre Anwendung. Anstatt jede Variable einzeln auf Ausreißer zu prüfen (z.B. mittels Z-Score wie in Kapitel 9 beschrieben), ermöglicht die Mahalanobis-Distanz eine ganzheitliche Betrachtung. Ein Datensatz, dessen Werte einzeln im plausiblen Bereich liegen, kann in seiner Kombination unplausibel sein.

Es soll das Beispiel aus Abschnitt 12.2.3 nochmals aufgegriffen werden.

Im Praxisbeispiel 12.2.3 wurde nur das Feld '#TCP Packets' zur Outlier-Analyse benutzt. Eine multivariate Anomalieerkennung durch die Mahalanobis-Distanz könnte so aussehen:

Die Mahalanobis-Distanz berücksichtigt im Gegensatz zu univariaten Ansätzen die Korrelationsstruktur zwischen den TCP-Paket-Typen. So ist beispielsweise bei normalem Netzwerkverkehr eine ausgewogene Verteilung zwischen SYN-, ACK- und RST-Paketen zu erwarten, während DDoS-Angriffe charakteristische multivariate Signaturen aufweisen.

Der Ansatz modelliert das normale TCP-Verkehrsverhalten durch einen Mittelwertvektor $\boldsymbol{\mu}$ und eine Kovarianzmatrix $\boldsymbol{\Sigma}$ der vier TCP-Paket-Kategorien auf Basis von Baseline-Daten, d.h. historischen Daten mit normalen Netzwerkverkehr.

Für jeden neuen Zeitpunkt t wird ein Vektor

$$\boldsymbol{x}_t = [\text{SYN}, \text{SYN-ACK}, \text{ACK}, \text{RST}]_t$$

gebildet und die Mahalanobis-Distanz zur Baseline berechnet:

$$D_M(\boldsymbol{x}_t) = \sqrt{(\boldsymbol{x}_t - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}_t - \boldsymbol{\mu})}$$

Überschreitet $D_M(\boldsymbol{x}_t)$ einen definierten Schwellwert, wird eine Anomalie signalisiert.

Eine darauf aufbauende kontinuierliche multivariate Anomalie-Überwachung könnte so aussehen:

Prompt für DDoS-Anomalieerkennung mit Mahalanobis-Distanz

Das Verzeichnis ist *C:\Daten*. Die Datei darin *DDos.csv* enthält in der ersten Zeile die Feldnamen. Die Feldtrennung ist ",". Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Felder '#SYN Packets ', '#SYN-ACK Packets', '#ACK Packets', '#RST Packets' - Achtte auf das Leerzeichen in '#SYN Packets '.
2. Entferne Zeilen mit fehlenden Werten und setze negative Werte auf 0
3. Implementiere eine Anomalieerkennung mit fester Baseline:
   - Bestimme eine feste Referenz-Baseline aus den ersten 1000 "normalenSZeitperioden für $\boldsymbol{\mu}$ und $\boldsymbol{\Sigma}$
   - Berechne für alle nachfolgenden Zeitpunkte die Mahalano -bis-Distanz des aktuellen 4-Variablen-Vektors zur festen Baseline. Die höchste Mahalanobis-Distanz aus der Baseline definiert die MD-Grenze.
   - Markiere Zeitpunkte, die eine größere Abweichung der Mahalanobis-Distanz als die MD-Grenze der Baseline haben
4. Erstelle einen zweifach unterteilten Plot:
   - Oberer Plot: Originale Zeitreihe der Gesamt-TCP-

Pakete (Feld '#TCP Packtes')in #D8E1F4, detektierte Anomalien als rote Punkte

- Unterer Plot: Mahalanobis-Distanz über Zeit in schwarz mit MD-Grenze als als rote gestrichelte Linie, detektierte Anomalien als rote Punkte über der Schwelle. Mahalanobis-Distanz-Skalierung logarithmisch.

5. Speichere die Grafik in 'DDoS_Mahalanobis_Zeitreihe.pdf'

Prompt 14.2: Prompt für DDoS und Mahalanobis-Distanz

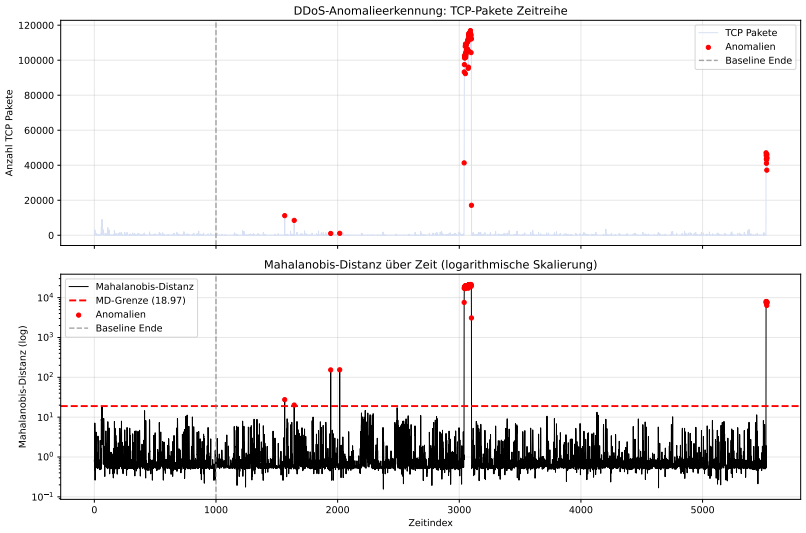


Abbildung 14.3: DDoS-Anomalieerkennung mittels Mahalanobis-Distanz. Bemerkenswert sind zwei Anomalien um Zeitindex 1900 und 2000, die trotz normaler TCP-Paketanzahl detektiert wurden – diese deuten auf subtilere Angriffsmuster hin, bei denen die Paketkomposition (SYN/ACK/RST-Verhältnisse) anomal ist, ohne dass die Gesamtpaketanzahl signifikant erhöht wird.

Das Verfahren bietet gegenüber dem univariaten KS-Test-Ansatz aus Abschnitt 12.2.3 erhebliche Vorteile: Während ein isolierter Anstieg der Gesamt-TCP-Pakete auch durch legitimen Verkehr verursacht werden kann, identifiziert die Mahalanobis-Distanz "ungewöhnliche Kombinationen" von TCP-Flags als multivariate Anomalien. Die feste Baseline ermöglicht dabei eine konsistente und nachvollziehbare Bewertung aller Zeitpunkte gegen dasselbe Referenzprofil, was die Interpretierbarkeit der Ergebnisse erhöht und eine klare Demonstration der Verfahrenseffektivität ermöglicht.

14.2.2 Plausibilitätsprüfung

Die Mahalanobis-Distanz kann auch proaktiv zur Validierung von Dateneingaben eingesetzt werden, insbesondere in Echtzeit-Systemen oder bei der Erfassung von Stammdaten. Anstatt zahlreiche, komplexe WENN-DANN-Regeln zu definieren, kann ein auf historischen, validen Daten trainiertes Mahalanobis-Modell als allgemeiner Plausibilitätscheck dienen.

Wird beispielsweise ein neues Produkt in einem Warenwirtschaftssystem angelegt, könnten die Dimensionen Länge,

Breite, Höhe und Gewicht geprüft werden. Das System berechnet die Mahalanobis-Distanz der neuen Wertekombination zum "normalen" Produktportfolio. Eine sehr hohe Distanz deutet auf eine unplausible Kombination hin (z.B. ein sehr großes, aber extrem leichtes Produkt) und kann eine Warnung auslösen, die eine Bestätigung durch den Benutzer erfordert. Dies verhindert die Eingabe von Tippfehlern oder unsinnigen Daten bereits an der Quelle.

14.2.3 Risikobewertung

Im Finanz- und Versicherungswesen ist die Identifikation von ungewöhnlichem Verhalten entscheidend für das Risikomanagement.

14.3 Grenzen und Herausforderungen

Trotz ihrer Stärken ist die Mahalanobis-Distanz kein Allheilmittel und unterliegt bestimmten Annahmen und Einschränkungen, die bei ihrer Anwendung berücksichtigt werden müssen.

Erstens ist die Methode **sensitiv bei kleinen Stichproben**. Wenn die Anzahl der Datenpunkte n nicht wesentlich größer ist als die Anzahl der Dimensionen p, kann die Schätzung der Kovarianzmatrix instabil oder ungenau sein. Ist $n \leq p$, ist die Kovarianzmatrix singulär und nicht invertierbar, was die Berechnung unmöglich macht.

Als Faustregel sollte der Stichprobenumfang n mindestens $10 \cdot p$ betragen, damit die Kovarianzmatrix hinreichend stabil geschätzt werden kann.

Zweitens basiert die statistische Interpretation der Distanz über die χ^2-Verteilung auf der Annahme, dass die Daten **multivariat normalverteilt** sind. Bei stark schiefen Verteilungen oder Daten mit mehreren Clustern (multimodale Verteilungen) kann die Mahalanobis-Distanz irreführende Ergebnisse liefern, da das Konzept eines einzigen Zentrums (μ) und einer einzigen elliptischen Form (Σ) die Datenstruktur nicht adäquat beschreibt. Box-Cox-Transformationen oder andere Datentransformationen können hier Abhilfe schaffen.

Die Mahalanobis-Distanz erfasst nur lineare Korrelationen. Bei nichtlinearen Zusammenhängen (z.B. U-Form) können Ausreißer unerkannt bleiben. Hierfür sind Kernel-Methoden oder nicht-lineare Modelle besser geeignet.

Drittens leidet die Methode unter dem **"Fluch der Dimensionalität"** (Curse of Dimensionality). In sehr hochdimensionalen Räumen neigen die Abstände zwischen den Punkten dazu, sich anzugleichen, was die Unterscheidung zwischen normalen Punkten und Ausreißern erschwert. Dadurch verliert die Distanzmetrik an Diskriminierungsfähigkeit, und die Schätzung der Kovarianzmatrix wird numerisch instabil.

Die Kovarianzmatrix wird zudem sehr groß ($p \times p$), was die Berechnung und Invertierung aufwendig macht.

Die vielleicht größte Herausforderung ist jedoch, dass die Berechnung von Mittelwert und Kovarianzmatrix selbst **stark durch Ausreißer beeinflusst** wird. Extreme Ausreißer können das berechnete Zentrum verzerren und die Varianzen aufblähen. Dies führt dazu, dass der Ausreißer selbst eine geringere Distanz erhält als erwartet (Maskierungseffekt) - vgl. das Beispiel auf Seite 100 im Kapitel 9 (Univariate Ausreißer-Analyse), während andere, normale Punkte fälschlicherweise als weiter entfernt erscheinen.

Eine Lösung für dieses Problem ist analog zu den univariaten Analysetools eine **robuste Mahalanobis-Distanz**, die auf einer robusten Schätzung der Kovarianzmatrix basiert.

Ein populärer Ansatz ist der **Minimum Covariance Determinant (MCD)**-Algorithmus. Er sucht nach jener Teilmenge von Datenpunkten, deren Kovarianzmatrix die geringste Determinante aufweist. Mittelwert und Kovarianz werden dann nur aus dieser "sauberen" Teilmenge berechnet, was die Schätzung unempfindlich gegenüber den Ausreißern macht, die außerhalb dieser Teilmenge liegen.

14.4 Zusammenfassung

Dieses Kapitel hat die Mahalanobis-Distanz als ein zentrales Werkzeug der multivariaten Datenanalyse vorgestellt. Ausgehend von der Unzulänglichkeit der euklidischen Distanz bei korrelierten Daten wurden die mathematischen Grundlagen, einschließlich Mittelwertvektor und Kovarianzmatrix, erläutert. Ein Schlüsselaspekt ist die Transformation der Daten in einen Raum, in dem Korrelationen aufgehoben und Varianzen normiert sind, was eine statistisch fundierte Abstandsmessung ermöglicht. Die Verbindung zur Chi-Quadrat-Verteilung erlaubt es, objektive Schwellenwerte für die Ausreißererkennung zu definieren.

Die praktischen Anwendungen in der Datenqualität, von der Identifikation unplausibler Kombinationen in Stammdaten über die Plausibilisierung von Eingaben bis hin zur Risikobewertung, wurden anhand von Beispielen illustriert. Trotz ihrer Mächtigkeit wurden auch die Grenzen der Methode aufgezeigt: die Annahme der Normalverteilung, die Sensitivität gegenüber Ausreißern (Maskierungseffekt) und die Herausforderungen in hochdimensionalen Daten. Als Lösung für

die Anfälligkeit gegenüber Ausreißern wurde die robuste Mahalanobis-Distanz auf Basis des MCD-Algorithmus eingeführt.

Zusammenfassend lässt sich festhalten, dass die Mahalanobis-Distanz ein unverzichtbares, interpretierbares Verfahren für jeden Datenanalysten ist, der über die univariate Betrachtung hinausgehen und die komplexen Beziehungen in seinen Daten verstehen möchte.

Hauptkomponentenanalyse und Ausreißer 15

Die Hauptkomponentenanalyse, weithin bekannt als PCA (Principal Component Analysis), ist eine der fundamentalsten und am weitesten verbreiteten Techniken in der multivariaten Datenanalyse und im maschinellen Lernen. Sie dient primär der Dimensionsreduktion und der Merkmalsextraktion, indem sie die Struktur von hochdimensionalen Daten vereinfacht, ohne dabei einen signifikanten Informationsverlust zu erleiden.

Für den theoretischen Hintergrund wird auf Anhang D verwiesen.

15.1 Die Hauptkomponentenanalyse

In vielen realen Datensätzen, wie z. B. bei Finanzmarktdaten, Sensordaten aus der Industrie oder Genomdaten, sind die beobachteten Variablen oft stark miteinander korreliert. Dies bedeutet, dass sie redundante Informationen enthalten.

Die Zielsetzung der PCA ist es, diese Redundanz zu eliminieren, indem sie die Daten in ein neues Koordinatensystem transformiert.

Das Grundprinzip besteht darin, einen Satz von möglicherweise korrelierten Variablen in einen neuen Satz von unkorrelierten Variablen umzuwandeln. Diese neuen, unkorrelierten Variablen werden als Hauptkomponenten (Principal Components, PCs) bezeichnet.

Eine der leistungsfähigsten Anwendungen der PCA jenseits der reinen Dimensionsreduktion ist die Erkennung von Ausreißern (Outliers).

Die Grundidee ist, dass PCA die „normalen“ Korrelationsmuster und die Hauptvariationsrichtungen der Daten erfasst. Datenpunkte, die diesen Mustern gut entsprechen, sind „normal“, während Ausreißer von diesen Mustern abweichen.

Die PCA wurde 1901 von Karl Pearson als eine Methode der geometrischen Anpassung von Datenpunkten an eine Linie oder Ebene eingeführt. Unabhängig davon wurde sie in den 1930er Jahren von Harold Hotelling im Kontext der statistischen Faktoranalyse weiterentwickelt.

Die Ausreißererkennung mit PCA ist eine unüberwachte Methode. Sie benötigt keine vorab gelabelten Daten, was sie besonders nützlich macht, wenn eine manuelle Kennzeichnung von Anomalien unpraktikabel ist.

15.2 Methode des Rekonstruktionsfehlers

Die gebräuchlichste Methode zur Ausreißererkennung mit PCA basiert auf dem Konzept des Rekonstruktionsfehlers.

15.2.1 Beschreibung

Der Prozess ist wie folgt:

1. **Projektion:** Ein Original-Datenpunkt $\mathbf{x}$ aus dem p-dimensionalen Raum wird auf den durch die ersten k Hauptkomponenten aufgespannten Unterraum projiziert.

2. **Rekonstruktion:** Der Datenpunkt wird von diesem k-dimensionalen Unterraum zurück in den ursprünglichen p-dimensionalen Raum projiziert. Dieser rekonstruierte Punkt $\hat{\mathbf{x}}$ ist die bestmögliche Annäherung an den Originalpunkt unter Verwendung von nur k Hauptkomponenten.

3. **Fehlerberechnung:** Der Rekonstruktionsfehler ist der Abstand zwischen dem Originalpunkt $\mathbf{x}$ und seinem rekonstruierten Gegenstück $\hat{\mathbf{x}}$. Typischerweise wird der quadrierte euklidische Abstand verwendet.

Der Raum, der von den verworfenen Hauptkomponenten (denen mit den kleinsten Eigenwerten) aufgespannt wird, heißt "Residual Subspace". Der Rekonstruktionsfehler ist die quadrierte Länge der Projektion eines Datenpunkts in diesen Residualraum.

Der **Rekonstruktionsfehler** $RE(\mathbf{x})$ eines Datenpunktes $\mathbf{x}$ ist ein Maß dafür, wie gut der Punkt durch den durch die ersten k Hauptkomponenten definierten Unterraum repräsentiert wird. Er ist definiert als:

$$RE(\mathbf{x}) = \|\mathbf{x} - \hat{\mathbf{x}}\|^2$$

Der Rekonstruktionsfehler ist in der statistischen Prozesskontrolle auch als squared prediction error (SPE) oder Q-Statistik bekannt. Eine zweite Metrik, die oft verwendet wird, ist die Hotelling's T^2-Statistik, die den Abstand eines Punktes vom Ursprung im Hauptkomponenten-Raum misst.

Normale Datenpunkte, die den in den Hauptkomponenten erfassten Korrelationen folgen, liegen nahe am k-dimensionalen Unterraum und haben daher einen kleinen Rekonstruktionsfehler.

Ausreißer hingegen weichen von diesen Mustern ab und liegen weiter vom Unterraum entfernt, was zu einem großen Rekonstruktionsfehler führt. Dieser Fehler kann direkt als Anomalie-Score verwendet werden: Je höher der Fehler, desto wahrscheinlicher ist der Punkt ein Ausreißer.

15.2.2 Praxisbeispiel: Kardiodaten

Die nachfolgend verwendeten Daten stammen aus dem gleichen Cardiovascular Disease Dataset wie in Abschnitt 14.1.3 beschrieben.

Prompt für Outlier-Erkennung mit PCA-Rekonstruktionsfehler

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten 'height', 'weight' und 'ap_hi' für die Outlier-Analyse aus
2. Führe eine PCA mit 2 Hauptkomponenten auf die drei ausgewählten Variablen durch und berechne für jeden Datenpunkt den Rekonstruktionsfehler
3. Erstelle eine dreidimensionale Grafik mit den Datenpunkten im Farbcode #D8E1F4 und markiere 100 Werte mit dem größten Rekonstruktionsfehler blau.
4. Erstelle zudem für jede Spalte 'height', 'weight' und 'ap_hi' eine Dichte der Verteilung und lege ein Histogramm der 100 Outlier-Daten darüber.
5. Speichere diese 4 Grafiken in einer 2x2-Grafik unter "cardio_-PCA_Rekonstruktion.pdf"

Prompt 15.1: Prompt für PCA-Outlier

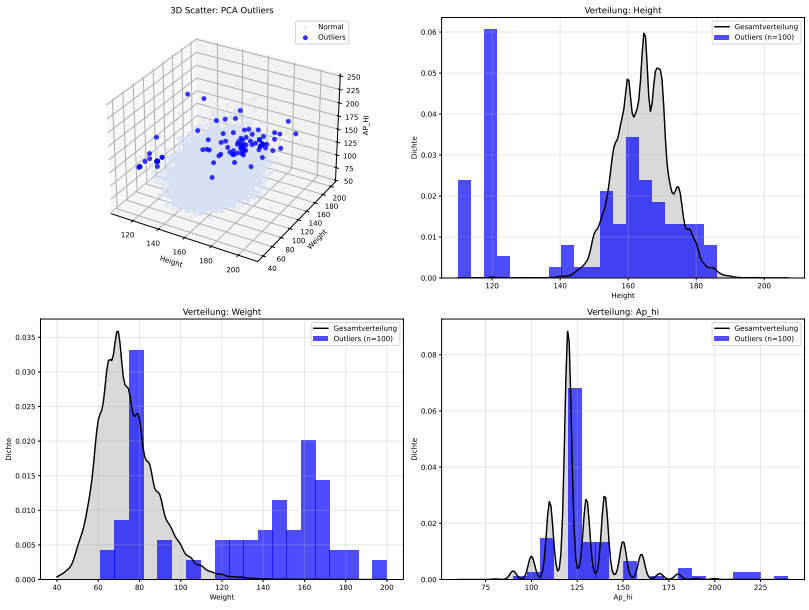


Abbildung 15.1: Ein Vergleich mit den Outliern der aus Abbildung 14.2 (Mahalanobis-Distanz) zeigt, dass ähnliche Ausreißer gefunden werden.

15.3 Vergleich PCA- und Mahalanobis-Distanz-Outlier

PCA-Rekonstruktionsfehler und Mahalanobis-Distanz liefern oft ähnliche Outlier, weil beide Methoden fundamentale statistische Eigenschaften der Datenverteilung nutzen. Der Hauptgrund liegt darin, dass beide Ansätze die Korrelationsstrukturen der Daten berücksichtigen.

Die Mahalanobis-Distanz misst Abstände unter Berücksichtigung der Kovarianzmatrix, wobei Punkte als Outlier erkannt werden, wenn sie weit vom multivariaten Zentrum entfernt sind und dabei die natürlichen Korrelationen zwischen Variablen berücksichtigt werden. Der PCA-Rekonstruktionsfehler erfasst hingegen Punkte, die schlecht durch die Hauptkomponenten repräsentiert werden. Punkte mit hohem Rekonstruktionsfehler liegen oft abseits der Hauptvariationsrichtungen, welche den stärksten Korrelationsmustern entsprechen.

Beide Methoden weisen eine mathematische Verwandtschaft auf, da sie auf der Kovarianzmatrix basieren. Die Mahalanobis-Distanz verwendet die inverse Kovarianzmatrix direkt, während PCA dieselbe Kovarianzmatrix durch Eigenvektoren diagonalisiert. Bei normalverteilten Daten existiert sogar ein direkter mathematischer Zusammenhang zwischen beiden Distanzen.

Darüber hinaus funktionieren beide Methoden optimal unter ähnlichen Bedingungen. Sie setzen voraus, dass die Daten annähernd normalverteilt sind, erfassen bevorzugt lineare Zusammenhänge und reagieren auf extreme Werte in ähnlicher Weise. Die Unterschiede zwischen den Methoden werden erst bei komplexeren Strukturen deutlich, insbesondere wenn nichtlineare Strukturen vorliegen, verschiedene Dimensionalitäten verwendet werden oder lokale von globalen Outliern unterschieden werden sollen.

Praktisch betrachtet ist es ein gutes Zeichen für robuste Ergebnisse, wenn beide Methoden ähnliche Outlier identifizieren. PCA kann bei hochdimensionalen Daten effizienter sein, während die Mahalanobis-Distanz oft direkter interpretierbar ist.

Im Fall des cardio-Datensatzes mit den Variablen height, weight und ap_hi sind lineare Korrelationen vorhanden, beispielsweise zwischen Körpergröße und Gewicht, weshalb beide Methoden ähnliche ungewöhnliche Kombinationen dieser Werte als Outlier identifizieren.

15.4 Minor Component Analysis (MCA)

Eine alternative Methode zur Ausreißererkennung mit PCA konzentriert sich auf die **Minor Components** – die Eigenvektoren mit den kleinsten Eigenwerten. Diese Komponenten repräsentieren die Richtungen mit der geringsten Varianz in den Daten und sind oft mit Rauschen oder seltenen Abweichungen verknüpft.

Die **Minor Components** sind die Eigenvektoren $\mathbf{v}_{p-k+1}, \mathbf{v}_{p-k+2}, \ldots, \mathbf{v}_p$ mit den kleinsten Eigenwerten $\lambda_{p-k+1} \leq \lambda_{p-k+2} \leq \ldots \leq \lambda_p$. Der von ihnen aufgespannte Unterraum wird als **Minor Subspace** oder **Noise Subspace** bezeichnet.

Die Minor Components entsprechen dem "Residual Subspace" oder "Noise Subspace". Während die Hauptkomponenten die dominanten Muster erfassen, zeigen die Minor Components, in welche Richtungen die Daten nur wenig variieren.

Die Grundidee ist, dass normale Datenpunkte hauptsächlich in den Richtungen der Major Components (große Eigenwerte) variieren und nur wenig Variation in den Minor Components zeigen. Ausreißer hingegen können signifikante Projektionen auf die Minor Components haben.

Der Prozess ist wie folgt:

1. **Auswahl der Minor Components:** Wähle die letzten m Eigenvektoren (mit den kleinsten Eigenwerten) aus der PCA.
2. **Projektion:** Projiziere jeden Datenpunkt $\mathbf{x}$ auf den von den Minor Components aufgespannten Unterraum.
3. **Anomalie-Score:** Berechne die Summe der quadrierten Projektionen als Anomalie-Score.

Der **Minor Component Score** $MCS(\mathbf{x})$ eines Datenpunktes $\mathbf{x}$ ist definiert als:

$$MCS(\mathbf{x}) = \sum_{j=p-m+1}^{p} (\mathbf{x}^T \mathbf{v}_j)^2 \tag{15.1}$$

wobei $\mathbf{v}_j$ die j-te Minor Component ist und m die Anzahl der verwendeten Minor Components.

Ein großer MCS-Wert deutet darauf hin, dass ein Datenpunkt in Richtungen variiert, die für die meisten anderen Datenpunkte untypisch sind. Dies ist ein starker Indikator für anomales Verhalten.

15.4.1 Zusammenhang von Rekonstruktions- und Minor Component Methode

Die Rekonstruktionsmethode und die Minor Component Methode sind mathematisch **äquivalent**, wenn die Anzahl der verwendeten Komponenten vollständig ist. Das sieht man so:

Jeder Datenpunkt $\mathbf{x} \in \mathbb{R}^n$ lässt sich über die Hauptkomponenten-Basis exakt darstellen:

$$\mathbf{x} = \sum_{i=1}^{n} a_i \mathbf{PC}_i$$

Die Rekonstruktion von $\mathbf{x}$ mit den ersten k Hauptkomponenten liefert:

$$\hat{\mathbf{x}} = \sum_{i=1}^{k} a_i \mathbf{PC}_i$$

Der Rekonstruktionsfehler ist

$$\|\mathbf{x} - \hat{\mathbf{x}}\| = \left\| \sum_{i=k+1}^{d} a_i \mathbf{PC}_i \right\| = \sqrt{\sum_{i=k+1}^{d} a_i^2}.$$

Dies ist genau der Minor Component Score für die letzten $(n-k)$-Hauptkomponenten.

Hinweis

Outlier aus der Rekonstruktionsmethode entsprechen den Outlier der Minormethode, wenn in Summe alle Hauptkomponenten verwendet werden. Ansonsten werden im Allgemeinen nicht die gleichen Outlier gefunden.

Speziell bei hochdimensionalen Daten mit vielen Hauptkomponenten, können beide Methoden eingesetzt werden und auch kombiniert werden.

$$\text{Combined Score} = \alpha \cdot RE(\mathbf{x}) + (1 - \alpha) \cdot MCS(\mathbf{x})$$

wobei $\alpha \in [0, 1]$ ein Gewichtungsparameter ist. Alternativ können beide Scores separat als Kriterien verwendet werden, wobei ein Punkt als Ausreißer gilt, wenn er bei mindestens einem der beiden Scores einen Schwellenwert überschreitet.

Die Minor Component Analysis ist besonders effektiv bei der Erkennung von Ausreißern, die nur in wenigen Dimensionen anomal sind, während sie in den Hauptvariationsrichtungen normal erscheinen.

Prompt für PCA Minor-Methode

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten 'height', 'weight' und 'ap_hi', 'age' für die Outlier-Analyse aus
2. Führe eine PCA mit 2 Hauptkomponenten und berechne für jeden Datenpunkt den Rekonstruktionsfehler

3. Führe zudem auf die letzte Hauptkomponente die Minor Komponenten Methode durch

Prompt 15.2: Prompt für PCA Minor-Methode

Das Ergebnis ist folgende Tabelle:

Tabelle 15.1: Vergleich der extremsten Outlier mit Durchschnittswerten

Variable	Durchschnitt	Rekonstruktion-Outlier		Minor-Score-Outlier	
		Abweichung	%	Abweichung	%
Height [cm]	164.41	−52.41	−31.9	−42.41	−25.8
Weight [kg]	74.14	+92.86	+125.2	+86.86	+117.2
AP_hi [mmHg]	126.68	+53.32	+42.1	−6.68	−5.3
Age [days]	19464.26	+312.74	+1.6	+1525.74	+7.8

15.5 Grenzen der PCA in der Ausreißeranalyse

Obwohl die PCA eine der am weitesten verbreiteten Methoden zur Dimensionsreduktion und Ausreißeranalyse ist, besitzt sie inhärente Grenzen, die in praktischen Anwendungen beachtet werden müssen:

Linearitätsannahme: Die PCA erfasst ausschließlich lineare Korrelationen zwischen Variablen. Nichtlineare Strukturen oder gekrümmte Mannigfaltigkeiten in den Daten (z.,B. bei komplexen Sensordaten oder Bilddaten) können nicht adäquat dargestellt werden. In solchen Fällen sind Erweiterungen wie *Autoencoder* geeigneter (siehe Kapitel 16).

Varianz als Informationsmaß: PCA basiert auf der Annahme, dass die Richtungen mit der größten Varianz auch die wichtigsten Informationen enthalten. Für viele Anwendungen trifft dies zu, doch Anomalien können sich auch in Richtungen mit geringer Varianz verbergen. Deshalb müssen Rekonstruktionsfehler- und Minor-Component-Methoden bewusst kombiniert oder ergänzt werden.

Sensitivität gegenüber Ausreißern: Da die Kovarianzmatrix die Grundlage der PCA bildet, kann ein einzelner starker Ausreißer die Eigenvektoren erheblich verzerren. Dadurch werden die Hauptkomponenten nicht mehr repräsentativ für die Mehrheit der Daten.

Zusammenfassung

PCA ist eine effiziente Methode für lineare Muster und moderate Dimensionen, stößt jedoch an Grenzen, wenn Daten stark nichtlinear, hochdimensional oder durch Ausreißer verzerrt sind.

15.6 Zusammenfassung

Dieses Kapitel hat die Hauptkomponentenanalyse (PCA) als eine leistungsstarke multivariate Methode vorgestellt. Der grundlegende Zweck der PCA ist die Dimensionsreduktion durch die Transformation korrelierter Originalvariablen in einen neuen Satz unkorrelierter Variablen, der Hauptkomponenten. Diese Komponenten werden so konstruiert, dass sie sukzessive die maximale verbleibende Varianz im Datensatz erfassen. Der Berechnungsprozess umfasst die Standardisierung der Daten, die Berechnung der Kovarianz- oder Korrelationsmatrix und deren Eigenwertzerlegung. Die resultierenden Eigenvektoren definieren die Hauptkomponenten, während die Eigenwerte den von jeder Komponente erklärten Varianzanteil angeben.

Die Interpretation der Ergebnisse erfolgt oft visuell mittels eines Scree-Plots, der bei der Wahl der relevanten Komponentenanzahl hilft, sowie durch die Analyse der Loadings, die den Einfluss jeder Originalvariable auf die Hauptkomponenten beziffern. Eine wesentliche Anwendung der PCA im Bereich der Datenqualität ist die Ausreißeranalyse. Die Methode des Rekonstruktionsfehlers nutzt die Tatsache, dass Ausreißer von den dominanten Mustern der Daten abweichen. Sie haben daher nach einer Projektion auf einen Unterraum geringerer Dimension und anschließender Rückprojektion einen hohen Rekonstruktionsfehler. Dieser Fehler dient als effektiver Anomalie-Score, besonders in hochdimensionalen Datensätzen. Trotz ihrer Stärken ist die PCA auf die Erfassung linearer Zusammenhänge beschränkt und kann durch Ausreißer in den Trainingsdaten selbst beeinflusst werden.

Autoencoder zur Datenanomalie-Erkennung

16

Während klassische Methoden der Datenqualitätssicherung auf vordefinierten Regeln oder statistischen Annahmen basieren, stoßen diese bei hochdimensionalen, komplexen oder impliziten Datenstrukturen an ihre Grenzen. Insbesondere die Erkennung subtiler, multidimensionaler Anomalien stellt eine erhebliche Herausforderung dar. An dieser Stelle bieten Methoden des maschinellen Lernens, insbesondere neuronale Netze, leistungsstarke Alternativen.

16.1 Autoencoder

Die Grundidee von Autoencodern für die Anomalieerkennung ist elegant und intuitiv:

Ein **Autoencoder** ist ein neuronales Netzwerk, das darauf trainiert wird, seine Eingabedaten über eine komprimierte interne Repräsentation (Bottleneck) zu rekonstruieren. Ziel ist es, durch die Minimierung des Rekonstruktionsfehlers eine informative kompakte Darstellung der Daten zu erlernen.

Das hier beschriebene Verfahren gehört zum **unüberwachten Lernen**, da das Modell die Struktur der Daten ohne vordefinierte Labels (wie "normal" oder "anomal"), nur aus den Daten selbst lernt.

Die Architektur eines Autoencoders lässt sich in drei Hauptteile gliedern:

1. **Encoder:** Der Encoder-Teil des Netzwerks nimmt die hochdimensionalen Eingabedaten X entgegen und transformiert sie durch eine oder mehrere verdeckte Schichten (Hidden Layers) in eine niedrigdimensionale Repräsentation z. Die Anzahl der Neuronen in den Schichten des Encoders nimmt typischerweise sukzessive ab. Mathematisch lässt sich der Encoder als eine Funktion $z = f(X)$ beschreiben.

2. **Latenter Raum (Bottleneck):** Dies ist die zentrale, schmalste Schicht des Netzwerks, welche die komprimierte Repräsentation z der Eingabedaten enthält. Die Dimensionalität des latenten Raums ist ein entscheidender Hyperparameter, der die Stärke der Kompression bestimmt. Der latente Raum wird auch als „Bottleneck"(Flaschenhals) bezeichnet.

   Die Wahl der Größe des latenten Raums ist ein Kompromiss: Zu groß, und das Netz lernt nur, die Daten zu kopieren. Zu klein, und es kann normale Variationen nicht mehr korrekt rekonstruieren, was zu vielen Fehlalarmen führt.

3. **Decoder:** Der Decoder empfängt die komprimierte Repräsentation z aus dem latenten Raum und versucht,

Die Stärke von Autoencodern liegt darin, dass sie keine expliziten Annahmen über die zugrunde liegende Datenverteilung treffen, wie es beispielsweise der in Anhang C beschriebene Kolmogorov-Smirnov-Test tut. Sie lernen die Verteilung implizit aus den Daten.

daraus die ursprünglichen Eingabedaten zu rekonstruieren. Seine Architektur ist typischerweise spiegelbildlich zum Encoder aufgebaut, d.h. die Anzahl der Neuronen in den verdeckten Schichten nimmt sukzessive zu bis die Ausgabeschicht die gleiche Dimensionalität wie die Eingabeschicht erreicht hat. Der Decoder kann als Funktion $X' = g(z)$ beschrieben werden.

Diese Vorgehensweise grenzt sich fundamental von klassischen statistischen Tests ab.

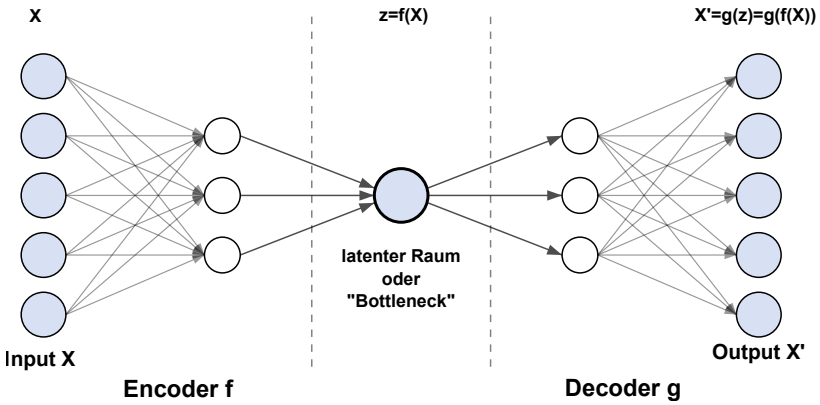


Abbildung 16.1: Grundlegende Architektur eines Autoencoders, bestehend aus Encoder, latentem Raum (Bottleneck) und Decoder. Der Encoder komprimiert die Eingabe X, und der Decoder versucht, sie aus der komprimierten Repräsentation z zu rekonstruieren, um die Ausgabe X' zu erzeugen.

16.2 Das unüberwachte Trainingsprinzip

"Lernen" in einem Autoencoder ist die Minimierung des Rekonstruktionsfehler.

Der **Rekonstruktionsfehler** ist ein Maß für die Abweichung zwischen den ursprünglichen Eingabedaten X und den vom Autoencoder rekonstruierten Daten X'. Er wird typischerweise als die Summe der quadrierten Abweichungen (Sum of Squared Errors, SSE) oder als mittlerer quadratischer Fehler (Mean Squared Error, MSE) berechnet.

Die Struktur eines Autoencoders erzwingt eine Kompression der Information. Würde man die mittlere Schicht nicht verengen, könnte das Netzwerk einfach lernen, die Eingabe durchzureichen ohne die zugrunde liegenden Muster zu verstehen.

Für einen einzelnen Datenpunkt X_i mit d Merkmalen kann der MSE wie folgt berechnet werden:

$$\text{MSE}(X_i, X_i') = \frac{1}{d} \sum_{j=1}^{d} (X_{ij} - X_{ij}')^2 \tag{16.1}$$

Das Training eines Autoencoders ist ein klassisches Beispiel für unüberwachtes Lernen, da keine gelabelten Daten (d.h. keine vordefinierten Anomalien) für den Lernprozess benötigt werden. Stattdessen dienen die Eingabedaten selbst als Ziel-Label.

Das zentrale Prinzip besteht darin, die Gewichte des Netzwerks so anzupassen, dass eine definierte Verlustfunktion (Loss Function), welche den Rekonstruktionsfehler quantifiziert, minimiert wird.

Die am häufigsten verwendete Verlustfunktion ist der bereits in Gleichung 16.1 gezeigte mittlere quadratische Fehler (MSE). Die Verlustfunktion L für einen gesamten Trainingsdatensatz mit N Datenpunkten ist dann der Durchschnitt der individuellen Rekonstruktionsfehler:

$$L(X, X') = \frac{1}{N} \sum_{i=1}^{N} \text{MSE}(X_i, X'_i) = \frac{1}{N \cdot d} \sum_{i=1}^{N} \sum_{j=1}^{d} (X_{ij} - X'_{ij})^2 \tag{16.2}$$

Dieser Wert wird während des Trainings durch Algorithmen der Gradientenabstiegsmethode minimiert, indem die Gewichte und Biases des Netzwerks iterativ angepasst werden (Backpropagation).

Die Wahl der richtigen **Aktivierungsfunktionen** ist ebenfalls von Bedeutung. In den verdeckten Schichten des Encoders und Decoders wird häufig die Rectified Linear Unit (ReLU) verwendet, da sie recheneffizient ist und das Problem des verschwindenden Gradienten (Vanishing Gradient Problem) mildert.

Für die Ausgabeschicht hängt die Wahl vom Wertebereich der Eingabedaten ab. Werden die Daten vor dem Training auf das Intervall $[0, 1]$ normalisiert, eignet sich die Sigmoid-Funktion gut als Aktivierungsfunktion der Ausgabeschicht, da sie ebenfalls Werte zwischen 0 und 1 ausgibt.

Die Skalierung der Eingabedaten (z.B. auf $[0, 1]$) ist entscheidend, da sonst Merkmale mit großen Wertebereichen die Verlustfunktion dominieren und das Training verzerren würden.

Die Qualität des Trainingsdatensatzes ist entscheidend. Enthält der „Normaldatensatz“bereits viele unentdeckte Anomalien, lernt der Autoencoder, auch diese zu rekonstruieren, was seine Fähigkeit zur Anomalieerkennung schwächt.

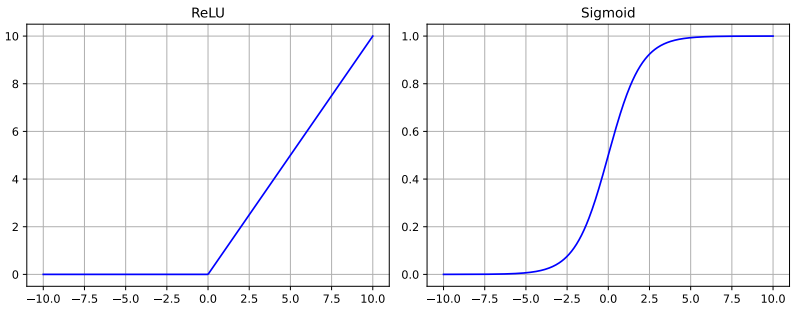


Abbildung 16.2: Visualisierung zweier Aktivierungsfunktionen: links die ReLU-Funktion $f(x) = \max(0, x)$, rechts die Sigmoid-Funktion $f(x) = \frac{1}{1+e^{-x}}$.

Beispiel: Trainingsprozess

Ein Unternehmen möchte die Qualität seiner Kundendaten überwachen. Es besitzt einen historischen Datensatz von 100.000 Kunden, der als „normal“und qualitativ hochwertig gilt.

1. **Datenvorbereitung:** Der Datensatz umfasst $p = 20$ numeri-

Um **Overfitting** zu vermeiden, bei dem das Modell die Trainingsdaten auswendig lernt, werden Techniken wie *Early Stopping* eingesetzt. Dabei wird das Training beendet, wenn der Fehler auf einem separaten Validierungsdatensatz nicht mehr sinkt.

sche Merkmale (z. B. Alter, Umsatz, Kundenbindungsdauer, Vertragsdauer). Alle Merkmale werden auf einen Bereich von $[0, 1]$ skaliert.

2. **Modellarchitektur:** Es wird ein Autoencoder mit folgender Struktur definiert:

$$20 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 8 \rightarrow 16 \rightarrow 20$$

also eine Eingabeschicht mit 20 Neuronen, zwei Encoder-Schichten mit 16 und 8 Neuronen, eine Bottleneck-Schicht mit 4 Neuronen, zwei Decoder-Schichten mit 8 und 16 Neuronen sowie eine Ausgabeschicht mit 20 Neuronen.
3. **Training:** Das Modell wird über mehrere Epochen (z. B. 50) mit Minibatches von Größe 256 auf den 100.000 normalen Kundendatensätzen trainiert. In jeder Epoche wird die Verlustfunktion (MSE) berechnet und die Gewichte werden mittels Backpropagation angepasst, um den Rekonstruktionsfehler zu minimieren.
4. **Ergebnis:** Nach dem Training ist der Autoencoder in der Lage, die Muster und Zusammenhänge normaler Kundendaten zu verstehen und diese mit einem geringen Fehler zu rekonstruieren. Neue Datenpunkte mit hohen Rekonstruktionsfehlern können als potenziell fehlerhaft oder anomal markiert werden.

16.3 Autoencoder und Encoder-Anwendungen

Autoencoder lernen durch die Minimierung des Rekonstruktionsfehlers zwischen Eingabe und Ausgabe. Je nach Anwendungsfall werden **nach** dem Training unterschiedliche Teile des trainierten Netzwerks verwendet.

16.3.1 Autoencoder Anwendungen

Bei der Nutzung des vollständigen Autoencoders (Encoder und Decoder) stehen zwei Hauptanwendungen im Vordergrund:

Die **Anomalieerkennung** nutzt den Rekonstruktionsfehler als Anomalie-Score – normale Daten werden gut rekonstruiert, während anomale Daten einen hohen Rekonstruktionsfehler aufweisen.

Der latente Raum ist im Allgemeinen „glatter“ als der Originalraum. Interpolationen zwischen Punkten führen daher häufiger zu realistischen und konsistenten neuen Beispielen, während direkte Interpolationen im Eingaberaum oft unplausible Artefakte erzeugen.

Die **Datenaugmentierung** verwendet den latenten Raum zur Generierung neuer Datenpunkte durch Interpolation zwischen existierenden latenten Repräsentationen und anschließende Dekodierung zu neuen synthetischen Daten.

16.3.2 Encoder Anwendungen

Wird hingegen nur der Encoder verwendet, erhält man als Output den Vektor im latenten Raum, der eine dimensionsreduzierte Repräsentation der ursprünglichen Eingabedaten darstellt.

Diese kompakte Repräsentation findet Anwendung in der **Datenkompression**, bei der die latenten Vektoren anstelle der ursprünglichen hochdimensionalen Daten gespeichert werden.

Die Dekompression erfolgt anschließend durch den Decoder, der die ursprünglichen Daten rekonstruiert.

Auch in der **Merkmalsextraktion**, wo die gelernten latenten Repräsentationen als informative Features für nachgelagerte maschinelle Lernverfahren dienen, werden Encoder angewandt.

16.4 Autoencoder und Anomalieerkennung

Nachdem der Autoencoder erfolgreich auf Normaldaten trainiert wurde, kann er zur Erkennung von Anomalien in neuen, ungesehenen Daten eingesetzt werden. Die einfachsten Methoden basieren auf der Analyse des Rekonstruktionsfehlers.

16.4.1 Schwellenwertbasierte Ansätze

Die direkteste Methode zur Klassifizierung eines Datenpunkts als Anomalie ist der Vergleich seines Rekonstruktionsfehlers mit einem vordefinierten Schwellenwert θ.

$$\text{Datenpunkt } X_i \text{ ist eine Anomalie, wenn } \text{MSE}(X_i, X_i') > \theta$$

Die entscheidende Frage ist, wie dieser Schwellenwert θ sinnvoll festgelegt werden kann. Eine manuelle Festlegung ist oft subjektiv und nicht robust. Ein besserer Ansatz ist die statistische Analyse der Rekonstruktionsfehler auf einem separaten Validierungsdatensatz, der ebenfalls nur aus Normaldaten besteht.

Die Wahl des Schwellenwerts ist immer ein Kompromiss zwischen der Erkennung von echten Anomalien (True Positives) und der fälschlichen Klassifizierung von normalen Daten als Anomalien (False Positives).

Eine gängige Methode besteht darin, den Schwellenwert auf Basis des Mittelwerts μ_{err} und der Standardabweichung σ_{err}

der Rekonstruktionsfehler des Validierungsdatensatzes zu definieren:

$$\theta = \mu_{err} + k \cdot \sigma_{err}$$

Hierbei ist k ein Faktor, der die Sensitivität des Detektors steuert. Ein typischer Wert für k ist 3, was bedeutet, dass jeder Datenpunkt, dessen Rekonstruktionsfehler mehr als drei Standardabweichungen über dem mittleren Fehler liegt, als Anomalie betrachtet wird. Dies entspricht der Annahme einer annähernd normalverteilten Fehlerverteilung für Normaldaten.

Die 3-Sigma-Regel ist eine Heuristik. Für Verteilungen, die nicht normal sind, gibt die **Chebyshev-Ungleichung** eine untere Schranke an, z.B. liegen mindestens 88,8% der Daten innerhalb von 3 Standardabweichungen.

16.4.2 Praxisbeispiel: DDoS-Anomalien

Als Beispiel zur Anwendung dient das Beispiel aus 14.2.1. Es sollen Anaomalien in einem Netzwerk erkannt werden. Der Prompt zur Generierung des Codes sieht folgendermaßen aus:

Prompt für DDoS-Anomalieerkennung mit Autoencoder

Das Verzeichnis ist *C:\Daten*. Die Datei darin *DDos.csv* enthält in der ersten Zeile die Feldnamen. Die Feldtrennung ist ",". Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Felder '#SYN Packets ', '#SYN-ACK Packets', '#ACK Packets', '#RST Packets' - Achte auf das Leerzeichen in '#SYN Packets '.
2. Entferne Zeilen mit fehlenden Werten und setze negative Werte auf 0
3. Implementiere eine Anomalieerkennung mit Autoencoder:
   - Verwende die ersten 2000 "normalen" Zeitperioden als Trainingsdaten für den Autoencoder
   - Trainiere einen Autoencoder mit geeigneter Architektur (4-3-2-3-4 Neuronen für Encoder-Decoder)
   - Berechne für alle Datenpunkte den Rekonstruktionsfehler (MSE zwischen Input und Output)
   - Bestimme die Anomalie-Schwelle als das 99.9 Perzentil der Rekonstruktionsfehler aus den Trainingsdaten
   - Markiere Zeitpunkte mit Rekonstruktionsfehlern über der Schwelle als Anomalien
4. Erstelle einen zweifach unterteilten Plot:
   - Oberer Plot: Originale Zeitreihe der Gesamt-TCP-Pakete (Feld '#TCP Packtes') in #D8E1F4, detektierte Anomalien als rote Punkte
   - Unterer Plot: Rekonstruktionsfehler über Zeit in schwarz mit Anomalie-Schwelle als rote gestrichelte Linie, detektierte Anomalien als rote Punkte über der Schwelle. Rekonstruktionsfehler-Skalierung logarithmisch.

5. Speichere die Grafik in 'DDoS_Autoencoder_Zeitreihe.pdf'

Prompt 16.1: Prompt für Autoencoder auf Anomalien in Zeitreihen

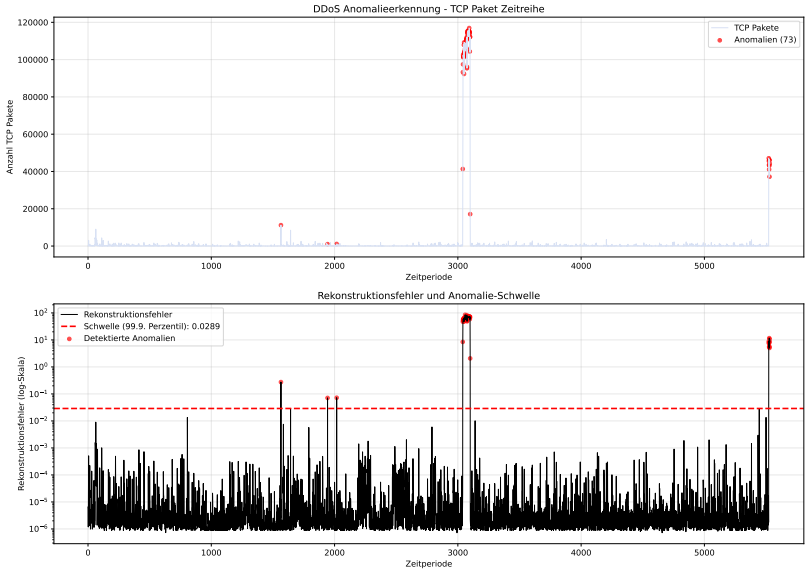


Abbildung 16.3: DDoS-Anomalieerkennung mittels Autoencoder. Ausreißer sind ähnlich zur Mahalanobis-Distanz in Abbildung 14.3.

16.5 Zusammenfassung

Dieses Kapitel hat gezeigt, dass Autoencoder ein leistungsstarkes, flexibles und datengetriebenes Werkzeug zur Anomalieerkennung und damit zur Verbesserung der Datenqualität darstellen. Ihre Stärke liegt in der Fähigkeit, komplexe, nichtlineare Muster in hochdimensionalen Daten ohne explizite Regeln oder statistische Annahmen zu lernen.

Zusammenfassend wurden folgende Kernpunkte behandelt:

- Die **Grundidee** basiert auf der Rekonstruktion von Daten. Das Modell lernt, was „normal“ ist, und identifiziert Anomalien durch einen hohen Rekonstruktionsfehler.
- **Einfache Ansätze** nutzen einen festen oder statistisch bestimmten Schwellenwert auf dem Rekonstruktionsfehler zur Klassifizierung von Anomalien. Visualisierungen wie Histogramme unterstützen diesen Prozess.
- Die **Anwendungsbereiche** sind vielfältig und reichen von der Überwachung von IT-Systemen und IoT-Geräten bis hin zur Betrugserkennung im Finanzsektor.

Trotz ihrer Stärken haben Autoencoder auch Grenzen. Ihr Erfolg hängt kritisch von der Verfügbarkeit eines repräsentativen und sauberen Datensatzes von Normaldaten für das

Training ab. Zudem kann die Interpretation, warum ein bestimmter Datenpunkt als anomal eingestuft wurde (Stichwort: Explainable AI), eine Herausforderung darstellen. Die Wahl der Architektur und der Hyperparameter erfordert Erfahrung und Experimentierfreude.

Selbstorganisierenden Karten (SOM) 17

Selbstorganisierende Karten, häufig als SOM oder Kohonen-Karten bezeichnet, stellen ein leistungsstarkes Paradigma des unüberwachten Lernens innerhalb der Familie der künstlichen neuronalen Netze dar. Entwickelt vom finnischen Wissenschaftler Teuvo Kohonen in den frühen 1980er Jahren, besteht ihr Hauptzweck darin, hochdimensionale Daten auf eine niedrigdimensionale, meist zweidimensionale, diskrete Gitterstruktur (die „Karte“) abzubilden.

Teuvo Kohonen (1934-2021) wird oft als Pionier der Neuroinformatik in Europa angesehen.

SOMs erlauben es, hochdimensionale Daten auf der Ebene darzustellen. So können Besonderheiten der Datenstruktur erkannt werden. Anders als bei PCA werden hier auch nichtlineare Muster darstellbar.

17.1 Grundlagen und Funktionsweise von SOMs

Das Kernprinzip der SOM basiert auf dem *unüberwachten, kompetitiven Lernen*.

“Unüberwacht“ bedeutet, dass der Algorithmus keine vorab gelabelten Daten benötigt. Er lernt die inhärenten Strukturen und Muster allein aus den Eingabedaten.

Das eigenständige Lernen ohne Aufsicht gibt dem Namen “Selbstorganisierende Netze“ seine Berechtigung.

“Kompetitiv“ bedeutet, dass die Neuronen der Karte in einen Wettbewerb treten, um einen bestimmten Eingabedatenpunkt zu repräsentieren.

Eine **Selbstorganisierende Karte (SOM)** ist ein künstliches neuronales Netz, das durch unüberwachtes Lernen eine 2-dimensionale, diskretisierte Repräsentation des Eingaberaums erzeugt. Ihr Hauptziel ist die Transformation eines hochdimensionalen Datensatzes in eine topologieerhaltende Karte, auf der die geometrischen Beziehungen der ursprünglichen Datenpunkte erhalten bleiben.

Die biologische Inspiration für SOMs stammt von den kortikalen Karten im Gehirn, bei denen verschiedene sensorische Informationen (z. B. visuelle oder auditive Reize) auf spezifischen, geordneten Bereichen der Großhirnrinde abgebildet werden.

Die topologische Erhaltung ist das entscheidende Merkmal. Im Gegensatz zu Techniken wie der Hauptkomponentenanalyse (PCA), die primär auf die Erhaltung der Varianz abzielt, legt die SOM einen Schwerpunkt auf die globale topologische Struktur. Dies wird durch einen kooperativen Lernprozess erreicht, bei dem nicht nur das “gewinnende“ Neuron, sondern auch seine Nachbarn auf der Karte angepasst werden.

Topologie-erhaltend bedeutet in diesem Zusammenhang, dass benachbarte Eingabevektoren (im Sinne des eukididischen Abstands) auch auf auf der Karte benachbart sind.

17.2 Architektur und Lernprozess

Die Architektur einer SOM ist konzeptuell einfach und besteht aus zwei Hauptkomponenten: einer Eingabeschicht und einer Ausgabeschicht, der eigentlichen Karte.

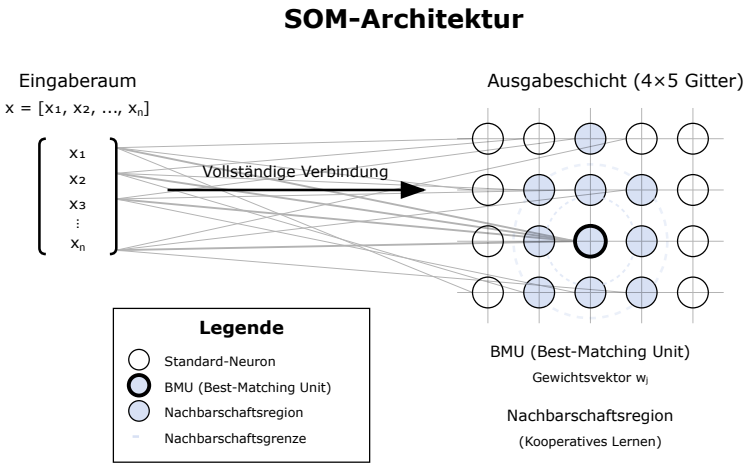


Abbildung 17.1: Grundlegende Architektur einer Selbstorganisierenden Karte. Ein n-dimensionaler Eingabevektor **x** (Eingabeschicht) wird auf ein 2D-Gitter von Neuronen projiziert (Ausgabeschicht). Jedes Neuron besitzt einen Gewichtsvektor $\mathbf{w}_j$ derselben Dimension n.

Dabei wird ein n-dimensionaler Eingabevektor **x** (Eingabeschicht) auf ein Neuron in der Topologie (Ausgabeschicht) abgebildet. Die Abbildung entsteht durch die Auswahl desjenigen Neurons, dessen Gewichte den kleinsten euklidischen Abstand zum Eingabevektor hat.

Ziel des Lernens ist es, n-dimensionale Gewichte zu finden, die ähnliche und damit gemäß euklidischer Distanz benachbarte Datensätze, auf benachbarte zweidimensionale Neuronen abzubilden. Die so entstehende Dimensionsreduktion nennt man topologische Karte:

Der große Vorteil einer topologischen Karte ist, dass sich hochdimensionale Datenstrukturen dadurch in ein- oder zweidimensionaler Form visualisieren lassen.

Eine **topologische Karte** ist eine Abbildung, die Vektoren aus einem n-dimensionalen Eingaberaum auf eine meist ein- oder zweidimensionale Anordnung von Neuronen (Ausgabeschicht) projiziert, wobei die topologische Nachbarschaft der Eingabevektoren möglichst erhalten bleibt.

17.2.1 Die Topologie

Für die Ausgabesicht einer topologische Karten, d.h. die Neuronen, gibt es unterschiedliche Formen oder Topologien:

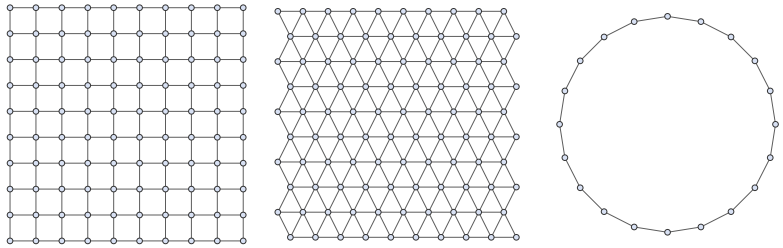


Abbildung 17.2: Topologien unterscheiden sich je nachdem wie die Neuronen in einem SOM verbunden sind. Hier die üblichsten: rechteckige, hexagonale Toplogie mit 100 Neuronen und Ring-Topologie mit 20 Neuronen.

Der Zweck der Topologie ist es, die Nachbarschaftsbeziehungen der Neuronen untereinander festzulegen.

Jedes Neuron besitzt dabei einen festen Abstand zu den anderen Neuronen. Dieser Abstand ist unabhängig von der Lernmenge und der zu lernenden Dimension und wird auch während des Lernvorgangs nicht verändert. Er hängt ausschließlich von der *gewählten Topologie* im Vorfeld des Lernens ab.

Es ist wichtig zu verstehen, dass beim Lernen nicht die Topolgie der Neuronen verändert wird, sondern die den Neuronen zugeordneten Gewichte. Die Topologie bleibt wie sie zum Beginn festgelegt wurde.

Um den Abstand zweier Neuronen zu bestimmen, gibt es Abstandsmaße, die von der Topologie abhängen.Bezeichnet man mit k_s die Koordinaten des Neurons s in der topologischen Karte (z. B. $k_s = (x_s, y_s) = (1, 3)$), dann ergibt sich der Abstand zu einem Neuron i abhängig von der gewählten Topologie.

Die gängigsten Abstandsdefinition sind:

1. **Rechteckige Topologie:**

$$d_A(k_s, k_i) = |x_s - x_i| + |y_s - y_i|$$

2. **Hexagonale Topologie:**

$$d_A(k_s, k_i) = \max\left(|x_s - x_i|, |y_s - y_i|, |x_s - x_i + y_s - y_i|\right)$$

x_s, y_s sind natürliche Zahlen. Damit ist auch der Abstand $d_A(x_s, y_s)$ ganzzahlig.

3. **Ringförmige Topologie (mit N Neuronen):**

$$d_A(k_s, k_i) = \min\left(|k_s - k_i|, N - |k_s - k_i|\right)$$

Für die rechteckige Topologie sind in Abbildung 17.3 die Neuronen mit Abstand 1 (grün) und Abstand 2 (gelb) zum Neuron $(2, 3)$ dargestellt.

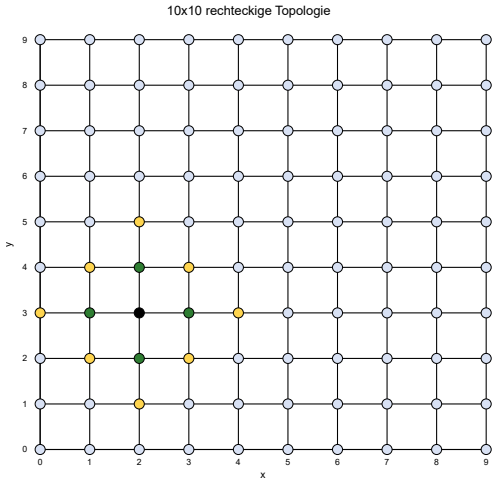


Abbildung 17.3: Nachbarschaften in einer rechteckigen SOM-Topologie um das Neuron (2, 3).

Der fundamentale Unterschied zu vielen anderen Algorithmen liegt in der Fähigkeit der SOM, die topologischen Nachbarschaftsbeziehungen der ursprünglichen Daten zu bewahren.

Das bedeutet, dass Datenpunkte, die im hochdimensionalen Eingaberaum nahe beieinander liegen auch auf der resultierenden Karte auf benachbarten Neuronen abgebildet werden.

Wichtig:

Randneuronen = am Rand des SOM-Gitters, nicht am Rand der Daten! Durch Randeffekt können Gitter-Randneuronen sogar zentrale Datenbereiche repräsentieren

Diese Eigenschaft macht SOMs nicht nur zu einem effektiven Werkzeug für die Dimensionsreduktion und Cluster-Analyse, sondern auch zu einer intuitiven Methode für die Visualisierung und Identifikation von Datenstrukturen, einschließlich der Erkennung von Anomalien.

17.2.2 Datenstandardisierung

Vor dem Training einer SOM müssen die Eingabedaten standardisiert werden, da der Lernalgorithmus distanzbasiert arbeitet. Ohne Standardisierung dominieren Features mit größeren Wertebereichen (z.B. Gewicht in kg: 50-100) gegenüber solchen mit kleineren Bereichen (z.B. Geschlecht: 1-2) die Distanzberechnung.

Die Standardisierung erfolgt typischerweise über den *StandardScaler*, der die Daten auf Mittelwert 0 und Standardabweichung 1 transformiert:

$$x_{\text{standardisiert}} = \frac{x - \mu}{\sigma}$$

wobei μ der Mittelwert und σ die Standardabweichung des jeweiligen Features ist. Dadurch erhalten alle Features das

gleiche Gewicht bei der Distanzberechnung und der Clusterbildung.

Die Skalierungsparameter (μ und σ) werden nur aus den Trainingsdaten berechnet und müssen für spätere Analysen (z.B. neue Datenpunkte, Validierung) gespeichert und wiederverwendet werden.

17.2.3 Lerprozess

Der Lernprozess ist iterativ und lässt sich in vier konzeptionelle Schritte für jeden präsentierten Eingabedatenpunkt $\mathbf{x}$ unterteilen:

Kohonen, T. (2001). *Self-Organizing Maps* (vgl. [32]) ist das maßgebliche Standardwerk, geschrieben vom Erfinder der SOMs selbst. Es bietet eine tiefgehende theoretische und mathematische Behandlung des Themas, einschließlich vieler Varianten und Anwendungsbeispiele. Es ist die primäre Quelle für ein vollständiges Verständnis der Materie.

Schritt 1: Initialisierung

Zu Beginn werden die Gewichtsvektoren $\mathbf{w}_j$ aller Neuronen j auf der Karte initialisiert. Jeder Gewichtsvektor hat dieselbe Dimension wie die Eingabedaten.

Eine gängige Methode ist die zufällige Initialisierung mit kleinen Werten aus dem Wertebereich der Eingabedaten.

Eine intelligentere Methode, die die Konvergenz beschleunigen kann, ist die Initialisierung mit Werten, die aus den Hauptkomponenten der Daten abgeleitet sind.

Eine schlechte Initialisierung kann dazu führen, dass die Karte sich „verdreht“oder topologische Defekte aufweist, bei denen die Nachbarschaftsbeziehungen nicht korrekt abgebildet werden. Daher ist die Initialisierungsphase kritischer als bei vielen anderen Netztypen.

Schritt 2: Kompetitiver Prozess

Für jeden einzelnen Eingabedatenpunkt $\mathbf{x}$ aus dem Trainingsdatensatz wird das Neuron auf der Karte gesucht, dessen Gewichtsvektor $\mathbf{w}_j$ dem Eingabedatenpunkt am ähnlichsten ist.

Diese Ähnlichkeit wird typischerweise durch den euklidischen Abstand gemessen. Das Neuron mit dem minimalen Abstand wird zur Best-Matching Unit (BMU) oder zum „Gewinnerneuron“für diesen Datenpunkt.

Die **Best-Matching Unit (BMU)** für einen gegebenen Eingabevektor $\mathbf{x}$ ist das Neuron c auf der Karte, dessen Gewichtsvektor $\mathbf{w}_c$ den geringsten euklidischen Abstand zu $\mathbf{x}$ aufweist. Formal gilt:

$$c(\mathbf{x}) = \arg\min_j \|\mathbf{x} - \mathbf{w}_j\|_2$$

wobei j über alle Neuronen der Karte iteriert und $\|\cdot\|_2$ die euklidische Norm bezeichnet.

Der kompetitive Prozess wird auch als „Winner-Takes-All“-Prinzip bezeichnet, obwohl bei SOMs durch die Nachbarschaftsfunktion auch die „Verlierer“in der Nähe des Gewinners noch lernen dürfen.

Dieser Schritt ist rein kompetitiv: Nur ein Neuron kann der Gewinner für einen bestimmten Eingabedatenpunkt sein. Die BMU ist die Repräsentation des Eingabedatenpunktes auf der niedrigdimensionalen Karte.

Schritt 3: Anpassung der Gewichte

Dies ist der entscheidende Schritt, der die Topologieerhaltung ermöglicht. Nicht nur der Gewichtsvektor der BMU, sondern auch die Gewichtsvektoren ihrer Nachbarn auf der Karte werden angepasst und in Richtung des Eingabevektors $\mathbf{x}$ „gezogen". Nachbarneuron ist man gemäß der topologischen Abstandfunktion wie auf Seite 185 beschrieben.

Die Stärke dieser Anpassung nimmt mit wachsendem Abstand zur BMU auf der Karte ab, d.h. die Gewichte des BMU-Neurons bekommen die größte Veränderung, zur BMU benachbarte Neuronen eine kleinere Anpassung der Gewichte.

Die Idee, dass benachbarte Neuronen gemeinsam lernen, ist ein Schlüsselkonzept. Es sorgt dafür, dass die Karte eine glatte, kontinuierliche Abbildung des Eingaberaums wird, anstatt nur eine einfache Vektorquantisierung durchzuführen.

Die formale Update-Regel für den Gewichtsvektor eines Neurons j zum Zeitpunkt $t+1$ lautet:

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \alpha(t) \cdot h_{cj}(t) \cdot (\mathbf{x} - \mathbf{w}_j(t))$$

Hierbei sind:

- $\alpha(t)$ die **Lernrate**, ein Faktor zwischen 0 und 1, der über die Zeit abnimmt. Sie steuert die allgemeine Größe der Anpassungsschritte.
- $h_{cj}(t)$ die **Nachbarschaftsfunktion**, die die Stärke der Anpassung für ein Neuron j in Abhängigkeit von seiner Entfernung zur BMU c auf dem Kartengitter steuert. Auch sie ist zeitabhängig.

Das Training wird oft in zwei Phasen unterteilt: eine erste, kurze „Ordnungsphase"mit hoher Lernrate und großem Radius, gefolgt von einer langen „Feinabstimmungsphase"mit kleinen Werten für $\alpha(0)$ und $\sigma(0)$.

Die Nachbarschaftsfunktion $h_{cj}(t)$ ist typischerweise eine glockenförmige Funktion, meist in Form einer Gauß-Funktion:

$$h_{cj}(t) = \exp\left(-\frac{d_A(k_c, k_j)^2}{2\sigma(t)^2}\right),$$

wobei $d_A(k_c, k_j)$ den topologischen Abstand zwischen den Neuronen c und j auf dem Ausgabegitter bezeichnet (vgl. Seite 185) und $\sigma(t)$ der **Nachbarschaftsradius** ist, der ebenfalls über die Zeit abnimmt.

Zu Beginn des Trainings ist $\sigma(t)$ groß, sodass weite Teile der Karte lernen. Im Laufe der Zeit wird $\sigma(t)$ kleiner, was zu

einer feineren Anpassung und Spezialisierung der Neuronen führt.

Schritt 4: Konvergenz

Der Prozess aus kompetitivem und kooperativem Lernen wird für viele Iterationen wiederholt, wobei der gesamte Datensatz mehrfach durchlaufen wird.

Die Lernrate $\alpha(t)$ und der Nachbarschaftsradius $\sigma(t)$ werden im Laufe der Zeit monoton verringert. Dies stellt sicher, dass zu Beginn grobe, globale Ordnungsstrukturen gebildet werden (große Lernrate, großer Radius) und gegen Ende des Trainings eine Feinabstimmung stattfindet, bei der sich die Karte stabilisiert und konvergiert. nach dem Training ist jedes Neuron auf einen bestimmten Bereich oder ein prototypisches Muster im Eingaberaum spezialisiert.

Haykin, S. (2008). *Neural Networks: A Comprehensive Foundation* ([33]) ist ein umfassendes Lehrbuch über neuronale Netze. Es enthält eine detaillierte Behandlung selbstorganisierender Karten mit einer ausführlichen Darstellung der theoretischen Grundlagen, Algorithmen und praktischen Anwendungen von Kohonen-Netzen (Seite 425 bis 474).

17.2.4 Quantisierungs- und Topologiefehler

Die Bewertung der Qualität eines trainierten Self-Organizing Maps erfolgt primär über zwei komplementäre Fehlermaße: den Quantisierungsfehler und den Topologiefehler.

Diese Metriken erfassen unterschiedliche Aspekte der Kartenqualität und sind entscheidend für die Beurteilung der Güte der gelernten Repräsentation.

Der **Quantisierungsfehler** misst die durchschnittliche Distanz zwischen jedem Eingabevektor und seinem Best Matching Unit (BMU). Er quantifiziert somit, wie gut die SOM-Prototypen die ursprünglichen Daten approximieren. Ein niedriger Quantisierungsfehler zeigt an, dass die Gewichtsvektoren der Neuronen nahe bei den Eingabedaten liegen und somit eine gute Codierung der Datenverteilung erreicht wurde.

Quantisierungsfehler: Bei standardisierten Daten sind Werte unter 1.0 gut, 1.0-1.5 akzeptabel. Werte über 2.0 erfordern Parameteranpassung – Balance zu topologischer Qualität wichtig

Der **Quantisierungsfehler** Q misst die durchschnittliche euklidische Distanz zwischen jedem Eingabevektor und seinem Best Matching Unit (BMU):

$$Q = \frac{1}{N} \sum_{n=1}^{N} \min_i \|\mathbf{x}_n - \mathbf{w}_i\|_2$$

wobei N die Anzahl der Eingabevektoren, $\mathbf{x}_n$ der n-te Eingabevektor und $\mathbf{w}_i$ der Gewichtsvektor des Neurons i ist.

Beispiel

Ein Quantisierungsfehler von 0.2 bei standardisierten Daten bedeutet, dass die BMUs im Durchschnitt 0.2 Standardabweichungen von den ursprünglichen Datenpunkten entfernt liegen.

Maßnahmen bei hohem Quantisierungsfehler

Kartengröße erhöhen - Mehr Neuronen für bessere Datenabdeckung

Training verlängern - längeres Training für bessere Konvergenz

Lernrate anpassen - Niedrigere Startlernrate für stabilere Anpassung

Initialisierung verbessern - PCA-basierte statt zufällige Gewichte
Datenvorverarbeitung - Ausreißer entfernen

Topologiefehler: Idealwert nahe 0 (unter 0.1 gut). Hoher Wert bedeutet "topologische Risse" – ähnliche Daten landen auf entfernten Kartenregionen, verschlechtert Interpretierbarkeit erheblich

Der **Topologiefehler** hingegen bewertet, inwieweit die topologische Struktur der ursprünglichen Daten in der zweidimensionalen Karte erhalten bleibt.

Er misst den Anteil der Eingabevektoren, für die das zweitbeste Matching Unit nicht in der unmittelbaren Nachbarschaft des BMU liegt. Ein niedriger Topologiefehler signalisiert, dass ähnliche Eingabedaten auf benachbarte Regionen der Karte abgebildet werden, was eine wichtige Eigenschaft für die Interpretierbarkeit und Visualisierung darstellt.

Der Topologiefehler liegt damit immer zwischen 0 (bester Wert) und 1 (schlechteseter Wert).

Der **Topologiefehler** T misst den Anteil der Eingabevektoren, für die das zweitbeste Matching Unit nicht in der unmittelbaren Nachbarschaft des BMU liegt:

$$T = \frac{1}{N} \sum_{n=1}^{N} u(\mathbf{x}_n)$$

mit $u(\mathbf{x}_n) = 1$, falls das zweitbeste Matching Unit für $\mathbf{x}_n$ nicht in der direkten Nachbarschaft des BMU liegt, sonst $u(\mathbf{x}_n) = 0$. Er bewertet die Erhaltung der topologischen Struktur der Eingabedaten.

Beispiel

Bei 1000 Datenpunkten und einem Topologiefehler von 0.12 haben 120 Eingabevektoren ihr zweitbestes Matching Unit nicht in der direkten Nachbarschaft – diese Punkte verletzen die topologische Ordnung.

Maßnahmen bei hohem Topologiefehler

Nachbarschaftsradius anpassen - Mit größerem Wert starten

Kartentopologie ändern - Hexagonale statt rechteckige Anordnung

Trainingsdauer erhöhen

Kartengröße optimieren - Höhere Kartenauflösung

Mehrere Läufe - Verschiedene Initialisierungen testen und beste wählen

2-stufiges Lernen - z.B. erste Phase mit großem Nachbarschaftsradius und Lernrate bei, zweite Phase bei kleinerem Nachbarshcaftsradius und kleinerer Lernrate

Für das Lernen von SOM-Netzen gibt es zwar Heuristiken, aber es ist auch viel "Try-and-Error" dabei.

Die Minimierung des Quantifizierungsfehler und es topologischen Fehlers stehen oft in einem Spannungsverhältnis zueinander.

Eine Verringerung des Quantisierungsfehlers durch kleinere Kartengröße oder längeres Training kann zu einem Anstieg des Topologiefehlers führen, da die begrenzte Anzahl von Neuronen möglicherweise nicht ausreicht, um sowohl eine gute Approximation als auch eine korrekte topologische Ordnung zu gewährleisten.

Die optimale SOM-Konfiguration erfordert daher eine sorgfältige Abwägung zwischen diesen beiden Qualitätskriterien, abhängig von der spezifischen Anwendung und den gewünschten Eigenschaften der resultierenden Karte.

Eine schlechte Karte bringt keine Vorteile bei der Datenanalyse.

Nachfolgende Tabelle gibt eine Orientierung, wann ein guter Lernerfolg erzeugt wurde:

SOM-Qualität	Quantisierung	Topologie	Interpretation
Sehr gut	< 0.5	< 0.05	Perfekte Balance beider Fehlermaße
Gut	$0.5 - 1.0$	$0.05 - 0.15$	Gute Balance beider Fehlermaße
Akzeptabel	$1.0 - 1.5$	$0.15 - 0.25$	Brauchbare Kartenqualität
Mittelmäßig	$1.5 - 2.0$	$0.25 - 0.35$	Verbesserung empfohlen
Schlecht	> 2.0	> 0.35	Parameteranpassung

Hinweis: Werte gelten für z-standardisierte Daten und typische SOM-Größen von 10×10 *bis* 50×50 *Neuronen.*

Tabelle 17.1: Richtwerte für Quantisierungs- und Topologiefehler bei standardisierten Daten

17.2.5 Häufige Missverständnisse

Häufige Missverständnisse bei Selbstorganisierenden Karten (SOM) sind:

SOM ist ein Clustering-Algorithmus. SOM ist primär ein Dimensionalitätsreduktions- und Visualisierungsverfahren; Clustering ist nur ein sekundärer Effekt.

Randneuronen liegen am Datenrand. Randneuronen befinden sich am Gitterrand, nicht am Rand der Datenverteilung. Jedoch sind hier oft Outlier zu finden. Auch ist der Rand

meist "dichter" (d.h. mehr zugeordnete Datensätze) als das Innere der Karte.

SOM erhält alle topologischen Beziehungen. SOM kann Nachbarschaftsbeziehungen verzerren; die topologische Treue muss daher separat gemessen werden.

Die topologische Treue sollte bei mindestens 90% der Neuronen eingehalten werden, d.h. ein topologischer Fehler von < 0.1

Größere Karten sind immer besser. Zu große Karten führen zu Überanpassung und schlechter Generalisierung.

SOM funktioniert mit beliebigen Daten. Datenskalierung ist essentiell; unterschiedliche Wertebereiche können das Ergebnis stark verzerren.

SOM ist deterministisch. Das Ergebnis hängt von der Initialisierung ab; mehrere Läufe mit verschiedenen Seeds sind empfehlenswert.

Trainingszeit ist unwichtig. Zu kurzes Training führt zu schlechter Topologie, zu langes zu Überanpassung.

17.2.6 Empfehlungen für SOM-Parameter

Die Wahl geeigneter Parameter für das Training einer Self-Organizing Map ist ein iterativer Prozess, der auf bewährten Heuristiken basiert und durch systematisches Experimentieren verfeinert werden muss.

Die nachfolgenden Empfehlungen dienen als Ausgangspunkt für die Parametersuche, wobei das Ziel stets die Erreichung mindestens guter Werte für Quantisierungs- und Topologiefehler gemäß der in Tabelle 17.1 enthaltenen Richtwerte ist.

Beispiel: Bei 100 000 Datensätzen ist $5\sqrt{100000} = 1581$, d.h. ein 40×40 Netz sollte eine ausreichende Auflösung haben.

Die **Netzgröße** sollte in einem angemessenen Verhältnis zur Anzahl der Trainingsdaten stehen. Eine bewährte Heuristik besagt, dass die Anzahl der Neuronen etwa $5\sqrt{N}$ betragen sollte, wobei N die Anzahl der Eingabevektoren darstellt.

Für kleinere Datensätze ($N < 1000$) sind Karten der Größe 10×10 bis 15×15 oft ausreichend, während größere Datensätze Karten von 30×30 bis 50×50 Neuronen erfordern können. Zu kleine Karten führen zu hohen Quantisierungsfehlern, während überdimensionierte Karten die Gefahr von Topologiefehlern erhöhen.

Das **zweistufige Lernen** hat sich als besonders effektiv erwiesen.

In der ersten Phase, der Organisationsphase, wird mit einer hohen anfänglichen Lernrate von $\alpha_0 = 0.5$ bis 0.9 und einem großen Nachbarschaftsradius von

$$\sigma_0 = \max(\text{Kartenlänge}, \text{Kartenbreite})/2$$

Bei einem 40×40-Netz ist beispielsweise $\sigma_0 = 20$ oder bei halber Diagonale $\sigma_0 = 28$.

oder

$$\sigma_0 = \frac{\sqrt{\text{Kartenlänge}^2 + \text{Kartenbreite}^2}}{2} \qquad \text{(halbe Diagonale)}$$

begonnen. Diese Phase sollte etwa das 10- bis 50-fache des Datensatzes als Iterationen umfassen und dient der groben topologischen Ordnung.

Die zweite Phase, die Feinabstimmungsphase, arbeitet mit deutlich niedrigerer Lernrate von $\alpha_0 = 0.05$ bis 0.3 und kleinerem Nachbarschaftsradius von $\sigma_0 = 1$ bis 2, um die Details der Datenverteilung zu erfassen.

Die **Nachbarschaftsfunktion** sollte typischerweise als Gaußsche Funktion implementiert werden, da diese zu stabileren Ergebnissen führt. Der Nachbarschaftsradius σ sollte exponentiell von seinem Anfangswert auf etwa 0.1 bis 0.5 am Ende des Trainings abnehmen. Eine zu schnelle Reduktion kann zu suboptimaler Topologie führen, während eine zu langsame Reduktion die Konvergenz verzögert.

Auf `https://brainboard.de/html_password/Casestudy_TSP.html` ist die Konvergenz eines SOMs am Beispiel des Traveling Salesman Problem grafisch dargestellt.

Die **Lernrate** sollte ebenfalls exponentiell abnehmen, wobei die Endlernrate etwa 1% bis maximal 10% der Anfangslernrate betragen sollte. Eine zu hohe Lernrate kann zu instabilem Verhalten führen, während eine zu niedrige Lernrate das Training unnötig verlängert ohne merkliche Qualitätssteigerung.

SOM Programm-Pakete

In der Praxis übernehmen moderne Implementierungen wie das Python-Paket `MiniSom` die exponentielle Reduktion von Lernrate und Nachbarschaftsradius automatisch. Dabei genügt es, die Anfangswerte und die Anzahl der Trainingsiterationen zu spezifizieren, während die zeitliche Anpassung der Parameter nach bewährten Algorithmen erfolgt.

Weitere Pakete: **C/C++:** Somoclu (hochperformant, GPU-Unterstützung), SOM-C-library. **Python:** SOMPY, scikit-som, Somoclu-Interface. **R:** kohonen (Standard), som, aweSOM (interaktiv).

Die **Anzahl der Trainingsiterationen** hängt stark von der Datenkomplexität ab. Erfahrungsgemäß muss jeder Datenpunkt zwischen 100 und 1000 Mal dem Netzwerk gezeigt werden, um eine zufriedenstellende Konvergenz zu erreichen. Für einen Datensatz mit N Beispielen entspricht dies

Insbesondere bei großen Datensätzen (>1 Mio.) sollten GPUs und die dazu passenden Programm-Pakete (z.B. Somoclu in C++ oder MiniSom-GPU in Python) eingesetzt werden

Tipp: erst wenig Datensätze testen, um grob die Leistungsfähigkeit des Computers zu kennen (Iterationen/Sekunde).

$100 \cdot N$ bis $1000 \cdot N$ Iterationen. Bei der zweistufigen Strategie entfallen typischerweise 20-50 Wiederholungen auf die Organisationsphase und 100-500 Wiederholungen auf die Feinabstimmung. Bei sehr großen Datensätzen reichen oft weniger Wiederholungen (10 bis 50).

Wichtig

Die Heuristiken dienen lediglich als Startpunkt. Die optimalen Parameter müssen durch systematisches Experimentieren mit verschiedenen Kombinationen ermittelt werden, wobei die Qualität anhand der Fehlermaße kontinuierlich überwacht wird. Erst wenn Quantisierungs- und Topologiefehler mindestens die Kategorie „gut“ erreichen, kann die SOM als zufriedenstellend trainiert betrachtet werden.

Die anzustrebenden Fehlerraten sind in Tabelle 17.1.

17.3 Praxisbeispiel: Kardiodaten

Nachfolgend wird ein SOM trainiert. Dabei wird im Prompt auf das Python-Paket *minisom* verwiesen. Hat man eine leistungsfähige GPU kann man hier auch *minisom-gpu* verwenden.

Der Vorteil an `minisom` ist, dass sich eigene Decay-Funktionen für α und σ leicht implementieren lassen. Die Ring-Topologie fehlt, aber kann einfach im Paket ergänzt werden.

Man sollte sich vom Paketnamen “mini“ nicht täuschen lassen. Dies bezieht sich nicht auf die Leistungsfähigkeit, sondern auf die minimale und effiziente Umsetzung.

Die nachfolgend verwendeten Daten stammen aus dem *Cardiovascular Disease Dataset* von *Kaggle* [9].

Der Datensatz wurde bereinigt (vgl. Abbildung Tabelle 8.1 im Kapitel 8) und ist als *cardio_train_bereinigt.csv* unter `www.handbuch-datenqualitaet.de` zu finden. Es sind 68 608 Patientendatensätzen mit 11 Merkmalen plus Zielvariable Herz-Kreislauf-Erkrankungen (CVD).

Prompt für SOM-Training mit MiniSom

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit “;“. Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'` und `'ap_hi'` für die Analyse aus.
2. Skaliere die fünf Merkmale für das SOM-Training.
3. Setze den Zufalls-Seed 47 für Reproduzierbarkeit
4. Initialisiere die Gewichte zufällig
5. Erzeuge ein SOM mit Gittergröße 35×35.
6. Führe ein zweistufiges Lernverfahren durch mit:

Das setzen eines Seeds ist aus didaktischen Gründen, um die Karten reproduzierbar zu machen. Wenn eigenes Karten nicht konvergieren, kann ein anderer Seed hlefen.

a) 1. Phase: $\sigma = 25$ als Startwert für den Nachbarschaftsradius und Lernrate $\alpha = 0.9$ mit 500 000 Iterationen.
b) 2. Phase: $\sigma = 5$ als Startwert für den Nachbarschaftsradius und Lernrate $\alpha = 0.3$ mit 1 500 000 Iterationen.

7. Mache eine Fortschrittsanzeige für den Lernfortschritt.
8. gib am Ende den quantifizierten und den topologischen Fehler aus.
9. Speichere die gelernten SOM-Gewichte mit np.save('cardio_-SOM_Gewichte.npy', som.get_weights()) und die Skalierung unter np.save('cardio_scaler_mean.npy', scaler.mean_) np.save('cardio_scaler_scale.npy', scaler.scale_) ins Verzeichnis *C:\Daten*
10. Erstelle eine Abbildung **Hits-Map** (Treffer-/BMU-Dichte) der trainierten Daten und speichere sie als `"cardio_SOM.pdf"` im Verzeichnis *C:\Daten*.
    Achte bei der Erstellung der Map darauf, dass MiniSom Koordinaten als (x, y) zurückgibt, aber NumPy-Arrays als [row, col] = [y, x] indexiert werden. Neuron (0,0) links unten.
11. Verwende für die Hits-Map die Skala (`cmap='Blues'`).

Der Startwert $\sigma_0 = 25$ ist die halbe Diagonale - d.h. $\sqrt{35^2 + 35^2}/2$

Wird diese Anweisung nicht beachtet, kann es sein, dass die Karte gespiegelt dargestellt wird.

Prompt 17.1: Prompt für SOM-Training mit MiniSom

Das von Claude Sonnet 4 erstelle Python-Script hat auf einem PC (AMD Ryzen 9 3950X) innerhalb 4 Minuten nachfolgendes SOM-Netz erstellt. Der Quantisierungsfehler ist 0.6896, topographischer Fehler 0.0250 und damit ist die Karte gemäß Tabelle 17.1 von guter Qualität:

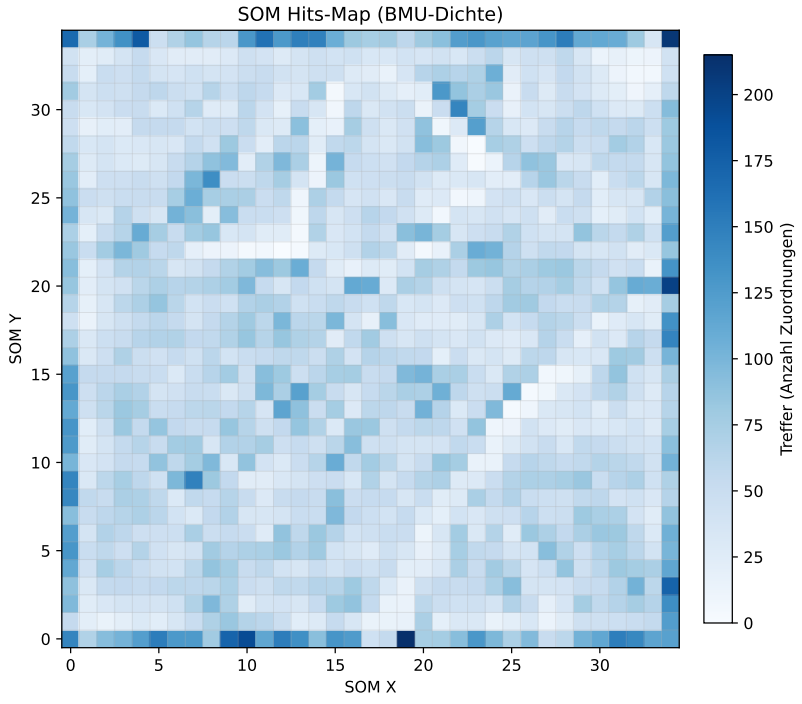


Abbildung 17.4: Self-Organizing Map (SOM) Hits-Map der Cardio-Daten mit 35×35 Neuronen. Die Farbskala zeigt die Anzahl der Datenpunkte (BMU-Dichte) pro Neuron von von 0 (weiß/hellblau) bis 250 (dunkelblau). Die gleichmäßige Verteilung über die gesamte Kartenfläche deutet auf eine erfolgreiche Selbstorganisation der fünf Merkmale hin. Dunkle Bereiche repräsentieren häufige Merkmalskombinationen, während helle Regionen seltene oder nicht existierende Kombinationen im 5-dimensionalen Merkmalsraum anzeigen.

Die Abbildung zeigt eine BMU Hits-Map. D.h. es wird für jedes Neuron der Karte gezeigt, für wie viele der 68 608 Datensätze es das Best Matching Neuron (BMU) ist.

Randeffekt: SOM-Neuronen am Gitterrand haben weniger Nachbarn, größeren Einzugsbereich und werden daher überproportional häufig aktiviert – ein systematisches Problem des Lernalgorithmus

Durch die Einfärbung sieht man schnell, wohin die Daten auf der Karte projiziert wurden.

Wichtig

das mit den erlernten Gewichten gespeicherte SOM-Netz für weitere Analysen verwenden zu können, müssen neben den gewichten auch die Skalierungsdaten aus der Lernphase gespeichert werden. Nur durch diese lässt sich das Netz korrekt anwenden.

Es gibt nun eine Topologie-erhaltene Karte der 5-Dimensionalen Daten. Nachfolgend soll diese Karte verwendet werden, um die Daten zu analysieren.

17.4 Labeling und Visualisierung

Die Abbildung 17.4 gibt noch wirklich nicht viel Einblick in die Daten. Interessant wird nun aber das Labeling der Daten.

Sind die Gewichte einer SOM-Karte trainiert, können die Neuronen anhand von Labels der zugeordneten Daten eingefärbt werden, um Muster zu visualisieren.

Die Topologie wurde vom SOM selbstständig aus den Daten gelernt. Es liegt damit eine Abbildung vom hochdimensionalen Raum auf den 2-Dimensionalen Raum vor. Nachbarschaften der Daten bleiben dabei erhalten. Jetzt können Untermengen der Daten (z.B. durch Labeling) oder neue Daten auf die Karte projeziert werden. Die Verteilung auf der Karte geben dann zusätzlichen Einblick.

Um Unterschiede zwischen Subpopulationen aufzudecken, können für binäre Labels (z. B. Kardioerkrankung ja/nein) separate Hits-Maps erstellt und verglichen werden. So ist im Cardiovascular Disease Dataset von Kaggle [9] das Label “gender“ vorhanden, das mit dem Wert 1 für Frauen und 2 für Männer kodiert ist. Durch die Gegenüberstellung der jeweiligen Hits-Maps lassen sich geschlechtsspezifische Unterschiede in den Datenmustern erkennen.

Folgendes Beispiel verdeutlicht dieses Vorgehen:

Prompt für SOM Hits-Maps Vergleich nach Geschlecht

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit “;“. Die bereits trainierten SOM-Gewichte sind in *cardio_SOM_Gewichte.npy* und die Skalierungsparameter in *cardio_scaler_mean.npy* und *cardio_scaler_scale.npy* gespeichert. Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'` und `'ap_hi'` für die Analyse aus.
2. Lade die gespeicherten SOM-Gewichte aus *cardio_SOM_-Gewichte.npy*.
3. Lade die Skalierungsparameter aus *cardio_scaler_mean.npy*

und *cardio_scaler_scale.npy* und rekonstruiere den StandardScaler.
4. Filtere die Daten basierend auf der Spalte `'gender'`:
   a) weiblich: `gender == 1`
   b) männlich: `gender == 2`
5. Skaliere beide Datensätze mit dem rekonstruierten Scaler.
6. Rekonstruiere das MiniSom-Objekt mit den geladenen Gewichten (Gittergröße 35×35).
7. Berechne für beide Gruppen separat die Best Matching Units (BMUs).
8. Erstelle zwei **Hits-Maps** (Treffer-/BMU-Dichte) in einem 1×2 Subplot-Layout:
   a) Links: Hits-Map für Frauen (`gender=1`) mit Titel "Frauen (gender=1)"
   b) Rechts: Hits-Map für Männer (`gender=2`) mit Titel "Männer (gender=2)"
9. Achte bei der Erstellung der Maps darauf, dass MiniSom Koordinaten als (x, y) zurückgibt, aber NumPy-Arrays als [row, col] = [y, x] indexiert werden.
10. Verwende für beide Hits-Maps die Skala (`cmap='Blues'`) mit einheitlicher Normierung (als Anteil der jeweiligen Daten) für bessere Vergleichbarkeit und zeige die Skala jeweils rechts neben der Grafik.
11. Im Verzeichnis *C:\Daten* speichere die 1×2 Grafik als `"cardio_SOM_gender_comparison.pdf"`

Hier werden zwei getrennte Datensätze auf Basis des Labels “gender“ generiert, die anschließend auf der gelernten Karte unabhängig voneinander projeziert werden

Prompt 17.2: Prompt für SOM Hits-Maps Vergleich nach Geschlecht

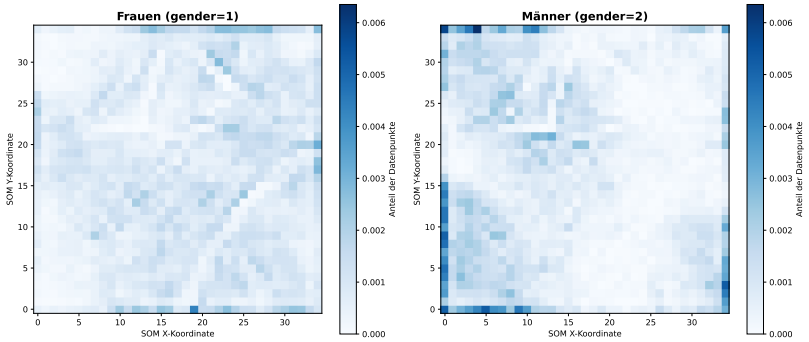


Abbildung 17.5: SOM Hits-Maps Vergleich nach Geschlecht. Die Karten zeigen die Verteilung von Frauen (links, n=44.674) und Männern (rechts, n=23.934).

Abbildung 17.5 zeigt signifikante Unterschiede. Bei Frauen sind die jeweiligen BMU gleichmäßiger verteilt. Die Männer weisen eine stärkere lokale Konzentrationen auf als Frauen.

Es können weitere Labelings durchgeführt werden, um mehr Insights aus den Daten zu bekommen.

Bei kategorialen Labels mit mehreren Ausprägungen (z.B. BMI-Klassen) kann jedem Neuron die Farbe derjenigen Labelklasse zugewiesen werden, die unter den auf ihm abgebildeten Daten am häufigsten vorkommt (**“Majority Voting“**).

Der Body-Mass-Index (BMI) berechnet sich als

$$\frac{\text{Körpergewicht in kg}}{(\text{Körpergröße in m})^2}.$$

Ein Wert über 25 gilt als Übergewicht.

Zusätzlich kann die Farbintensität die “Reinheit“ des Neurons visualisieren:

Eine kräftige Farbe signalisiert, dass fast ausschließlich Daten einer einzigen Klasse auf dem Neuron liegen, während eine blasse Farbe auf eine gemischte Zusammensetzung und somit einen weniger eindeutigen "Sieg" der Mehrheitsklasse hindeutet. Dies wird auch als **Homogenitätsvisualisierung** bezeichnet.

Prompt für SOM BMI-Klassen Majority Voting mit Homogenitätsvisualisierung

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Die bereits trainierten SOM-Gewichte sind in *cardio_SOM_Gewichte.npy* und die Skalierungsparameter in *cardio_scaler_mean.npy* und *cardio_scaler_scale.npy* gespeichert. Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'`, `'ap_hi'`, `'gender'` und `'cardio'` für die Analyse aus.
2. Berechne den BMI als $\frac{\text{weight}}{(\text{height}/100)^2}$ und kategorisiere in BMI-Klassen:
   - Untergewicht: BMI < 20
   - Normalgewicht: $20 \leq$ BMI < 25
   - Übergewicht: $25 \leq$ BMI < 30
   - Adipositas: BMI ≥ 30
3. Lade die gespeicherten SOM-Gewichte aus *cardio_SOM_Gewichte.npy*.
4. Lade die Skalierungsparameter und rekonstruiere den StandardScaler.
5. Filtere die Daten nach Geschlecht (`gender == 1` für Frauen, `gender == 2` für Männer).
6. Skaliere die SOM-Features und rekonstruiere das MiniSom-Objekt (Gittergröße 35×35).
7. Berechne BMUs und erstelle für beide Geschlechter BMI-Klassen-Maps mit Majority Voting:
   a) Bestimme für jedes Neuron die häufigste BMI-Klasse (Majority Voting)
   b) Berechne die "Reinheit" als Anteil der Mehrheitsklasse an allen auf dem Neuron abgebildeten Daten
   c) Weise jedem Neuron die Farbe der Mehrheitsklasse zu (Untergewicht: blau, Normalgewicht: grün, Übergewicht: orange, Adipositas: rot)
   d) Moduliere die Farbintensität entsprechend der Reinheit (kräftige Farbe = hohe Reinheit, blasse Farbe = niedrige Reinheit/gemischte Zusammensetzung)
8. Erstelle ein 1×2 Subplot-Layout:
   a) Oben links: Frauen - BMI-Klassen Majority Voting Map mit Homogenitätsvisualisierung
   b) Oben rechts: Männer - BMI-Klassen Majority Voting Map mit Homogenitätsvisualisierung

Es werden so neue Labels (BMI) definiert, die im Ursprungsdatensatz nicht vorhanden sind. Eine Alternative wäre hier Risikoklassen zu definieren (z.B. übergeichtig, Raucher und inaktiv).

Es muss immer auf die in der Lernphase definierten Skalierung skaliert werden. Dies gilt auch bei völlig neuen Daten, die auf die Karte projeziert werden.

9. Verwende keine Legende oder Erklärung
10. Speichere als `"cardio_SOM_BMI.pdf"`.

Achte bei der Erstellung der Maps darauf, dass MiniSom Koordinaten als (x, y) zurückgibt, aber NumPy-Arrays als [row, col] = [y, x] indexiert werden. Das Neuron (0,0) soll links unten sein.

Prompt 17.3: Prompt für SOM BMI-Klassen Majority Voting mit Homogenitätsvisualisierung

Die nachfolgende Grafik wurde mit dem Python-Script aus dem Prompt durch Sonnet Claude 4 erzeugt:

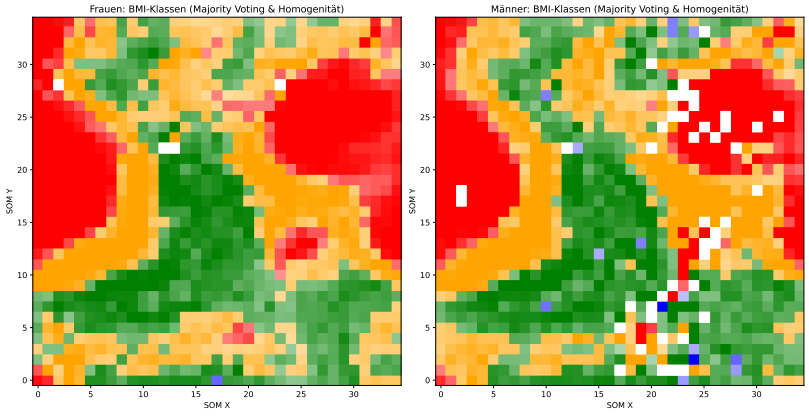


Abbildung 17.6: Jedes Neuron ist entsprechend der häufigsten BMI-Klasse eingefärbt: Grün = Normalgewicht (20–25 kg/m^2), Orange = Übergewicht (25–30 kg/m^2), Rot = Adipositas ($\geq 30\,kg/m^2$), Blau = Untergewicht ($<20\,kg/m^2$), Weiß = Neuron hat keinen Datensatz.

Die Farbintensität in obiger Abbildung repräsentiert die Homogenität des Neurons – kräftige Farben zeigen hohe Reinheit (dominante BMI-Klasse), blasse Farben weisen auf gemischte BMI-Verteilungen hin. Beide Geschlechter zeigen deutliche Cluster von Normalgewicht (grüne Regionen) und Adipositas (rote Bereiche) und ähneln sich sehr.

17.5 Component Planes

Component Planes (auch Komponentenkarten genannt) sind eine fundamentale Visualisierungstechnik für Self-Organizing Maps:

Component Planes (Komponentenkarten) visualisieren die Verteilung einzelner Eingabevariablen über die SOM-Struktur. Jede Component Plane zeigt für eine spezifische Eingabedimension die Gewichtswerte der entsprechenden Komponente über alle Neuronen.

Eine Component Plane gibt Einblick, wie die einzelen Datenfelder im Netz geordnet sind. Hier lassen sich auch Outlier für einezelen Datenfelder finden.

Die Darstellung erfolgt typischerweise als Heatmap, bei der jedes Neuron entsprechend dem Wert seiner Gewichtskomponente eingefärbt wird.

Diese Visualisierung macht es möglich, Gradienten und Cluster in den einzelnen Variablen direkt zu erkennen und deren räumliche Anordnung auf der Karte zu verstehen.

Component Planes sind besonders wertvoll für die Interpretation der gelernten Datenstrukturen, da sie zeigen, welche Bereiche der Karte für bestimmte Merkmalausprägungen spezialisiert sind. Regionen mit ähnlichen Farbmustern über mehrere Component Planes hinweg deuten auf korrelierte Variablen hin, während kontrastierende Muster inverse Beziehungen offenbaren können.

Die systematische Analyse mehrerer Component Planes kann wichtige Einblicke in die Datenstruktur liefern. Beispielsweise können medizinische Parameter wie Blutdruck, Körpergewicht und Alter charakteristische Verteilungsmuster aufweisen, die Risikogruppen oder physiologische Zusammenhänge sichtbar machen.

Durch die Betrachtung aller relevanten Component Planes gleichzeitig lassen sich komplexe multivariate Beziehungen verstehen, die in eindimensionalen Analysen verborgen bleiben würden.

Prompt für SOM Component Planes Cardiovascular Dataset

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Die bereits trainierten SOM-Gewichte sind in *cardio_SOM_Gewichte.npy* und die Skalierungsparameter in *cardio_scaler_mean.npy* und *cardio_scaler_scale.npy* gespeichert. Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'`, `'ap_hi'` für die SOM-Features sowie `'cardio'` für die Zielvariable aus.
2. Lade die gespeicherten SOM-Gewichte aus *cardio_SOM_Gewichte.npy*.
3. Lade die Skalierungsparameter und rekonstruiere den StandardScaler.
4. Rekonstruiere das MiniSom-Objekt (Gittergröße 35×35) mit den geladenen Gewichten.
5. Erstelle Component Planes für alle fünf SOM-Features:
   a) Extrahiere für jede Eingabedimension die entsprechende Gewichtskomponente aller Neuronen
   b) Arrangiere diese Werte in einer 35×35 Matrix entsprechend der SOM-Topologie
   c) Achte dabei auf die korrekte Zuordnung der MiniSom-Koordinaten (x, y) zu NumPy-Indizes [y, x]
6. Erstelle zusätzlich eine Krankheitsverteilungs-Map für kardiovaskuläre Erkrankungen:
   a) Skaliere alle Eingabedaten mit dem rekonstruierten StandardScaler
   b) Bestimme für jeden Patienten das Best Matching Unit (BMU) im SOM-Gitter
   c) Berechne für jedes Neuron den Prozentsatz der zugeordneten Patienten mit `cardio=1`
   d) Erstelle eine 35×35 Matrix mit den lokalen Erkrankungsraten (0-1)
7. Visualisiere alle Component Planes in einem 2×3 Subplot-Layout:
   a) Obere Reihe: height, weight, age
   b) Untere Reihe: ap_lo, ap_hi, kardiovaskuläre Erkrankungsrate

Im Feld `'cardio'` ist mit 1 definiert, ob eine kardiovaskuäre Erkrankung (CVD) vorliegt

Die gegenwärtigen Sprachmodelle sind so leistungsfähig, dass der Prompt nicht zu technisch definiert werden muss. Er kann so formuliert sein, dass man präziße angibt, was man will.

8. Verwende für die Feature Component Planes die Farbskala (`cmap='RdYlBu_r'`)
9. Verwende für die Krankheitsverteilungs-Map eine separate rote Farbskala (`cmap='Reds'`) mit Normierung von 0 bis 1 (entspricht 0% bis 100% Erkrankungsrate).
10. Beschrifte jeden Subplot mit dem entsprechenden Variablennamen und füge entsprechende Colorbars hinzu.
11. Entferne Achsenbeschriftungen und Ticks für eine saubere Darstellung.
12. Speichere die Visualisierung als `"cardio_SOM_component_planes_mit_krankheit.pdf"`.

WICHTIG: Achte bei der Erstellung der Maps darauf, dass MiniSom Koordinaten als (x1, y1) zurückgibt, aber NumPy-Arrays als [row, col] = [y1, x1] indexiert werden. Neuron (0,0) links unten.

Prompt 17.4: Prompt für SOM Component Planes Cardiovascular Dataset

Die Umsetzung des Prompts mit ChatGPT 5 liefert auf Anhieb:

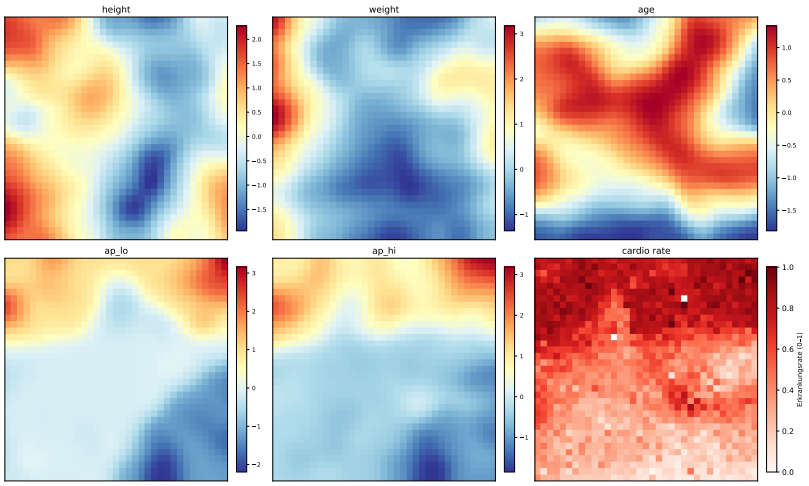


Abbildung 17.7: Dargestellt sind die räumlichen Gewichtsverteilungen für fünf kardiovaskuläre Parameter (height, weight, age, ap_lo, ap_hi) sowie die lokale Verteilung kardiovaskulärer Erkrankungen. Die Feature Component Planes verwenden eine einheitliche RdYlBu_r-Farbskala (rot = hohe, blau = niedrige Parameterwerte), während die Krankheitsverteilung in einer separaten Reds-Skala die lokalen Erkrankungsraten (0-100%) visualisiert.

Die Component Planes der Self-Organizing Map (Abbildung 17.7) visualisieren sowohl die gelernten Gewichtsverteilungen für jeden der fünf Eingabeparameter als auch die räumliche Verteilung kardiovaskulärer Erkrankungen auf dem zweidimensionalen 35×35 SOM-Gitter.

Die Ursprungsdaten wurden skaliert. . Die Gewichte entsprechen dieser Skalierung. D.h. sehr große Menschen haben ein hohes Gewicht (z.B. 3) und sind entsprechend rot abgebildet.

Auf der Karte ist zu erkennen, dass die schwersten Menschen überwiegend eine durchschnitliche Größe aufweisen.

Ist das Gewicht des BMU 0 liegt der entsprechende Datensatz im Erwartungswert dieser Ausprägung der Daten.

Die Component Planes für den diastolischen Blutdruck (`ap_lo`) und den systolischen Blutdruck (`ap_hi`) zeigen nahezu identische topologische Muster mit ausgeprägten rotorangen Regionen in den oberen Bereichen des SOM-Gitters.

Diese starke Übereinstimmung reflektiert die bekannte physiologische Kopplung beider Blutdruckwerte.

Wie nicht anders zu erwarten folgt die kardiovaskuläre Erkrankungsrate dem erhöhtem Blutdruck.

17.6 Overlay

Ein Overlay bei SOM ist eine mehrschichtige Visualisierungstechnik, bei der verschiedene Datenaspekte übereinander gelegt werden, um komplexe Zusammenhänge in den hochdimensionalen Daten sichtbar zu machen.

Die grundlegende Struktur besteht aus einer Basisvisualisierung, die als Hintergrundschicht fungiert und einer oder mehreren Overlay-Schichten, die zusätzliche Informationen darüber legen.

Overlays sind besonders hilfreich, wenn man bestimmte Ausprägungen hervorheben will - z.B. bei Kreditrisikodaten die 2-Dimensionale Datenstruktur und die Ausfallrate. So lassen sich Muster erkennen und ein besthenden Risikomodell verbessern.

Die Basisschicht stellt typischerweise eine kontinuierliche Variable als Heatmap dar, wobei die Farbintensität die Ausprägung der jeweiligen Eigenschaft in den verschiedenen SOM-Regionen widerspiegelt. Diese Grundkarte zeigt die räumliche Verteilung eines wichtigen Merkmals über das gesamte SOM-Gitter hinweg. Darüber werden dann Overlay-Informationen gelegt, meist in Form von Kontourlinien, Punkten oder anderen grafischen Elementen, die eine zweite oder dritte Variable repräsentieren.

Die Inaktivität ist im Datensatz als Feld *activ* hinterlegt und war nicht Teil des Trainings.

Im nachfolgenden Beispiel bildet die Inaktivitäts-Heatmap die Basisschicht, die den Anteil inaktiver Personen in jeder SOM-Region durch unterschiedliche Orangetöne darstellt.

Als Overlay werden Kontourlinien der Wahrscheinlichkeit für Herz-Kreislauf-Erkrankungen (CVD = Cardiovascular Disease) in verschiedenen Farben und Stärken darüber gelegt. Diese Kombination ermöglicht es, direkt zu erkennen, ob Bereiche mit hoher Inaktivität auch mit erhöhtem CVD-Risiko korrelieren, und macht medizinisch relevante Muster sofort sichtbar.

Prompt für SOM Inaktivitäts-CVD-Overlay-Analyse

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Die bereits trainierten SOM-Gewichte sind in *cardio_SOM_Gewichte.npy* und die Skalierungsparameter in *cardio_scaler_mean.npy* und *cardio_scaler_scale.npy* gespeichert. Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'`, `'ap_hi'` für die Analyse aus.
2. Lade die gespeicherten SOM-Gewichte aus *cardio_SOM_-Gewichte.npy*.
3. Lade die Skalierungsparameter und rekonstruiere den Standard Scaler.
4. Skaliere **alle** SOM-Features des **gesamten Datensatzes** mit dem rekonstruierten StandardScaler und rekonstruiere das MiniSom-Objekt (Gittergröße 35×35) mit den geladenen Gewichten.
5. Berechne für Daten des Gesamtdatensatzes die Best Matching Units (BMUs) im SOM-Gitter.
6. Filtere **nach der BMU-Berechnung** die Daten nach Geschlecht (`gender == 1` für Frauen, `gender == 2` für Männer) und erstelle für beide Geschlechter getrennt:
   a) Inaktivitäts-Map (Anteil `active == 0` Patienten pro Neuron)
   b) CVD-Map (Anteil `cardio == 1` der Patienten pro Neuron)
7. Erstelle ein 1×2 Subplot-Layout:
   a) Links: Frauen - Inaktivitäts-Heatmap (`cmap='Oranges'`, `alpha=0.7`) mit CVD-Kontourlinien bei 70%, 85%, 95% (Farben: blau, rot, dunkelrot; Linienbreiten: 1, 2, 3)
   b) Rechts: Männer - entsprechend
8. Beschrifte die Kontourlinien mit CVD-Wahrscheinlichkeiten (`fmt='CVD: %.0f%'`).
9. Speichere als `"cardio_SOM_inactivity_CVD_overlay.pdf"`.

WICHTIG: Achte bei der Erstellung der Maps darauf, dass MiniSom Koordinaten als (x1, y1) zurückgibt, aber NumPy-Arrays als [row, col] = [y1, x1] indexiert werden. Neuron (0,0) links unten.

Auf der Internetseite zum Buch sind alle Prompts zu finden. Diese sind bewusst in Latex-Quellcode. Dies gibt mehr Struktur und das LLM kann die Anweisung besser interbretieren. Fett vorgegebene Anweisungen können vom LLM erkannt werden.

Prompt 17.5: SOM Inaktivitäts-CVD-Overlay-Analyse

Der Python-Code mit ChatGPT 5.0 liefert folgende Abbildung (die gestrichelten Linien wurden nachträglich dazu gefügt):

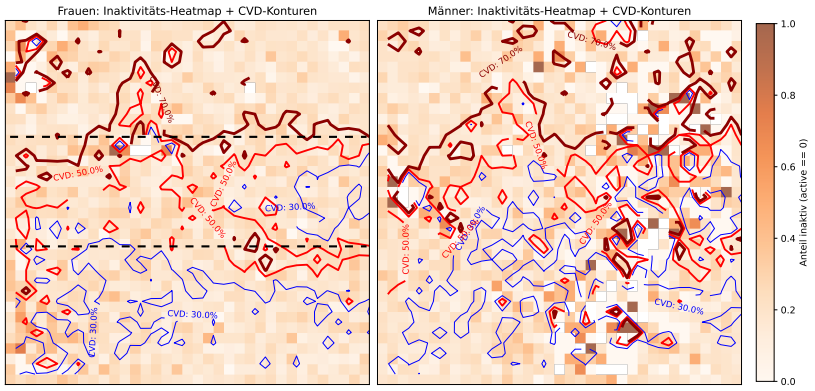


Abbildung 17.8: SOM-basierte Analyse des Zusammenhangs zwischen körperlicher Inaktivität und kardiovaskulärem Risiko nach Geschlecht. Die Heatmaps zeigen den Anteil körperlich inaktiver Personen (active=0) pro SOM-Neuron. Cardiovascular Disease (CVD)-Wahrscheinlichkeitscontours sind als Isolinien überlagert.

Frauen (links, n=44.674) zeigen eine klare vertikale Risiko-Verteilung mit dem oberen SOM-Drittel als Hochrisiko-Zone (50-70% CVD), dem mittleren Bereich als moderates Risiko (≈

50% CVD) und dem unteren Drittel als Niedrigrisiko-Zone (gestichelte Linie). Inaktivitäts-Cluster sind diffus verteilt. Dichte Cluster (mit hohem Anteil an Inaktivität) sind selten.

Männer (rechts, n=23.934) weisen ein komplexeres Muster mit isolierten CVD-Hotspots (bis 70%) auf, die räumlich über die gesamte SOM verteilt sind.

Inaktivität ist bei Männern höher verbreitet und hat auch mehr Auswirkung auf die kardiologische Gesundheit als bei Frauen. Die zeigt der Overlay, der Inaktivitätscluster bei Männern mit einem höheren Risiko versieht.

Overlays haben große Vorteile. Manchmal ist es zu Beginn der Anaylse besser mit unterschiedlichen Labels verschiedene Karten zu erzeugen und diese zu vergleichen, um ein "Gefühl" für die Daten zu entwickeln.

Die Stärke des Overlay-Ansatzes liegt darin, dass er die natürliche topologische Struktur des SOM erhält, während gleichzeitig mehrere Datendimensionen in einer einzigen, interpretierbaren Visualisierung kombiniert werden. Dies ermöglicht es, komplexe multivariate Beziehungen auf einen Blick zu erfassen und dabei sowohl kontinuierliche Verteilungen als auch diskrete Schwellenwerte oder Kategorien gleichzeitig darzustellen.

17.7 Zusammenfassung

Dieses Kapitel hat die Grundlagen und vielseitigen Visualisierungsmöglichkeiten von Selbstorganisierenden Karten (SOMs) detailliert behandelt. Als unüberwachtes Lernverfahren, das hochdimensionale Daten unter Beibehaltung topologischer Beziehungen auf eine niedrigdimensionale Karte projiziert, bieten SOMs einen einzigartigen, visuellen Ansatz zur Datenexploration und Mustererkennung.

SOMs sind ein hervorragendes Beispiel für „Explainable AI"im Bereich des unüberwachten Lernens, da ihre Ergebnisse visuell nachvollziehbar sind.

Die zentralen Erkenntnisse lassen sich wie folgt zusammenfassen:

Der Lernprozess der SOM basiert auf einem kompetitiven Schritt, der **Best-Matching Unit (BMU)** zu finden, und einem kooperativen Schritt, bei dem die Gewichte der BMU und ihrer topologischen Nachbarn angepasst werden. Dieser kooperative Aspekt ist entscheidend für die Erhaltung der Topologie und die Bildung geordneter, kontinuierlicher Karten.

Die Qualitätsbewertung einer trainierten SOM erfolgt primär durch zwei komplementäre Fehlermaße:

den **Quantisierungsfehler**, der die Approximationsqualität der Prototypen misst und

den **Topologiefehler**, der die Erhaltung der Nachbarschaftsbeziehungen quantifiziert.

Das Spannungsverhältnis zwischen diesen beiden Metriken erfordert eine sorgfältige Parameterabstimmung für optimale Kartenqualität.

Für die Interpretation und Analyse trainierter SOMs wurden verschiedene Visualisierungstechniken vorgestellt.

SOM als Datenqualitäts-Radar: Deckt Anomalien, Inkonsistenzen, fehlende Werte, Bias und Verteilungsprobleme visuell auf – essentiell für Datenvalidierung

Hits-Maps visualisieren die Datendichte und ermöglichen die Identifikation von Clustern durch Darstellung der BMU-Häufigkeiten.

Labeling-Techniken wie Majority Voting erlauben die farbkodierte Darstellung kategorialer Variablen mit Homogenitätsvisualisierung zur Bewertung der Klassenreinheit einzelner Neuronen.

Component Planes ermöglichen die separate Analyse einzelner Eingabevariablen durch Darstellung ihrer räumlichen Gewichtsverteilung über die gesamte Karte.

Ältere Bücher zu SOMs bieten oft zahlreiche Ideen und Anwendungsbeispiele, die zur damaligen Zeit aufgrund mangelder Rechnungsleistung schwer umsetzbar waren, aber heute durchaus "wiederbelebt" werden können.[34] zeigt beispielsweise zahlreiche Anwendungsmöglichkeiten von SOMs zum Thema "Finance".

Overlay-Visualisierungen kombinieren verschiedene Datenaspekte durch geschichtete Darstellung von Heatmaps und Kontourlinien für die simultane Analyse multipler Variablen.

SOMs sind wertvolle und mächtige visuelle Werkzeuge. Ähnlich zu bildgebenden Verfahren aus der Medizin erfordert die Anwendung allerdings Übung, um die richtigen Schlussfolgerungen aus den Bildern zu ziehen.

Anomalieerkennung mit Selbstorganisierenden Karten (SOM)

18

Im vorherigen Kapitel 17 wurde das SOM zur Datenvisualisierung dargestellt. Nun soll das Netz zur Anomalieerkennung genutzt werden.

Anomalien, auch als Ausreißer (Outlier) bekannt, sind Datenpunkte, die signifikant von der Mehrheit der Daten abweichen. Sie passen nicht in das erwartete Muster und werden daher von der SOM schlecht repräsentiert. Genau diese Eigenschaft wird ausgenutzt, um Ausreißer zu identifizieren.

Dieses Kapitel beleuchtet spezifische Methoden, wie der Quantisierungsfehler und die Unified Distance Matrix (U-Matrix) zur Aufdeckung von Anomalien genutzt werden können.

Anders als die klassische Mahalanobis-Distanz (Kapitel 14) und die PCA-Methode (Kapitel 15), die von (annähernd) multinormalverteilten und linearen Datenstrukturen ausgehen, kann ein SOM auch stark nichtlineare Datensätze auf Ausreißer untersuchen.

18.1 Outlier-Erkennung durch Quantisierungsfehler

Nachdem die SOM trainiert ist und die Gewichtsvektoren der Neuronen die Struktur der Daten repräsentieren, kann die Karte zur Identifikation von Anomalien verwendet werden. Die grundlegende Annahme ist, dass normale Datenpunkte, die Teil eines dichten Clusters sind, gut von der SOM repräsentiert werden. Anomalien hingegen, die per Definition selten und von den Hauptclustern entfernt sind, werden von keinem Neuron gut repräsentiert.

Der Quantifizierungsfehler ist ein distanzbasiertes Ausreißermaß. Seine Wirksamkeit hängt stark davon ab, ob die verwendete Distanzmetrik (meist euklidisch) für den gegebenen Datenraum aussagekräftig ist.

Der Quantisierungsfehler ist die direkteste und am weitesten verbreitete Methode zur Anomalieerkennung mit SOMs. Er misst, wie gut ein einzelner Datenpunkt von der trainierten Karte repräsentiert wird. Zur Wiederholung:

Der **Quantisierungsfehler (QE)** eines Eingabevektors $\mathbf{x}$ ist der euklidische Abstand zwischen $\mathbf{x}$ und dem Gewichtsvektor $\mathbf{w}_c$ seiner Best-Matching Unit (BMU) c.

$$QE(\mathbf{x}) = \|\mathbf{x} - \mathbf{w}_{c(\mathbf{x})}\|_2 \tag{18.1}$$

Die Intuition dahinter ist einfach:

- **Normale Datenpunkte:** Ein Datenpunkt, der zu einem der Hauptcluster in den Daten gehört, wird eine BMU finden, deren Gewichtsvektor ihm sehr ähnlich ist. Der Abstand zwischen dem Datenpunkt und dem

Gewichtsvektor der BMU wird daher klein sein, was zu einem **niedrigen Quantisierungsfehler** führt.

- **Anomalien/Ausreißer:** Ein anomaler Datenpunkt liegt definitionsgemäß weit entfernt von den dichten Regionen des Datenraums. Folglich wird kein Neuron einen Gewichtsvektor entwickelt haben, der diesem Punkt besonders nahekommt. Selbst die "beste" Übereinstimmung (die BMU) wird noch relativ weit vom Datenpunkt entfernt sein. Dies resultiert in einem **hohen Quantisierungsfehler**.

Der durchschnittliche Quantisierungsfehler über den gesamten Datensatz ist auch ein Maß für die Güte der trainierten SOM. Ein niedrigerer durchschnittlicher QE deutet darauf hin, dass die Karte die Daten insgesamt besser repräsentiert.

Der praktische Anwendungs-Workflow besteht darin, den QE für jeden einzelnen Datenpunkt im Datensatz zu berechnen. Anschließend wird die Verteilung dieser QE-Werte analysiert. Datenpunkte, deren QE-Wert einen bestimmten Schwellenwert überschreitet, werden als Anomalien klassifiziert. Die Bestimmung dieses Schwellenwerts ist ein kritischer Schritt. Gängige Ansätze sind:

Die Wahl des Schwellenwerts ist ein klassisches Problem: Ein zu niedriger Wert führt zu vielen Falsch-Positiven (normale Punkte als Anomalien), ein zu hoher Wert übersieht echte Anomalien (Falsch-Negative).

- **Perzentil-basierter Schwellenwert:** Man definiert, dass alle Datenpunkte über dem 95. oder 99. Perzentil der QE-Verteilung als Anomalien gelten.
- **Statistische Methoden:** Wenn die QE-Werte der „normalen"Daten einer bestimmten Verteilung folgen (z. B. Normalverteilung nach einer Transformation), können statistische Regeln wie der Z-Score (z. B. alle Punkte mit $Z > 3$) angewendet werden.
- **Visuelle Inspektion:** Ein Histogramm oder ein Boxplot der QE-Werte kann helfen, einen natürlichen „Knick"oder eine Grenze zu identifizieren, die die normalen von den anomalen Werten trennt.

To Do Analyse der QE-Verteilung

Nach dem Training einer SOM und der Berechnung der Quantisierungsfehler für alle Datenpunkte sollten folgende Schritte durchgeführt werden:

1. Erstelle ein Histogramm und einen Boxplot der QE-Werte, um einen visuellen Eindruck der Verteilung zu erhalten. Achte auf eine stark rechtsschiefe Verteilung, die typisch ist.
2. Berechne deskriptive Statistiken (Mittelwert, Median, Standardabweichung, Perzentile).
3. Definiere einen Schwellenwert, z. B. das 98. Perzentil.
4. Listen Sie die Datenpunkte auf, die diesen Schwellenwert überschreiten, und analysiere diese potenziellen Anomalien genauer im fachlichen Kontext.

Die obige To Do-Liste soll an folgendem Beispiel abgearbeitet werden:

Prompt für SOM Quantisierungsfehler-Analyse Cardiovascular Dataset

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Die bereits trainierten SOM-Gewichte sind in *cardio_SOM_Gewichte.npy* und die Skalierungsparameter in *cardio_scaler_mean.npy* und *cardio_scaler_scale.npy* gespeichert. Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'`, `'ap_hi'` für die SOM-Features
2. Lade die gespeicherten SOM-Gewichte aus *cardio_SOM_Gewichte.npy*.
3. Lade die Skalierungsparameter und rekonstruiere den StandardScaler.
4. Rekonstruiere das MiniSom-Objekt (Gittergröße 35×35) mit den geladenen Gewichten.
5. Berechne die Quantisierungsfehler für alle Datenpunkte:
   a) Skaliere alle Eingabedaten mit dem rekonstruierten StandardScaler
   b) Bestimme für jeden Datenpunkt das Best Matching Unit (BMU) im SOM-Gitter
   c) Berechne den euklidischen Abstand zwischen jedem Datenpunkt und seinem BMU
   d) Speichere die QE-Werte zusammen mit den ursprünglichen Datenindizes
6. Erstelle eine umfassende statistische Analyse der QE-Verteilung:
   a) Berechne deskriptive Statistiken: Min, Max, Mittelwert, Median, Standardabweichung, MAD, IQR, Schiefe, Kurtosis
   b) Berechne relevante Perzentile: 25., 50., 75., 90., 95., 98., 99., 99,5., 99,9. Perzentil
   c) Gebe alle Statistiken formatiert in einer übersichtlichen Tabelle aus und speichere diese unter `"cardio_SOM_qe_statistiken.txt"`
7. Visualisiere die QE-Verteilung in einem 1×2 Subplot-Layout:
   a) Linkes Subplot: Histogramm der QE-Werte mit 150 Bins
   b) Rechtes Subplot: Boxplot der QE-Werte (vertikal orientiert)
   c) Markiere das 99,9. Perzentil als rote Linie in den Plots
   d) Speichere die Grafik unter `"cardio_SOM_quantisierungsfehler_analyse.pdf"`
8. Identifiziere die 100 Datenpunkte mit den höchsten QE-Werten und speichere die entsprechenden Datenpunkte (mit allen verfügbaren Spalten) geordnet nach absteigenden OE-Werten in `"cardio_SOM_anomalien.csv"`

Verwende für die Grafiken #D8E1F4, andere Blautöne und schwarz.

Der Prompt könnte gekürzt werden, weil das LLM meist weiß, was zu tun ist. Allerdings ist es hilfreich, den Prompt so zu gestalten, dass man selbst weiß, was man tut.

Prompt 18.1: Prompt für SOM Quantisierungsfehler-Analyse Cardiovascular Dataset

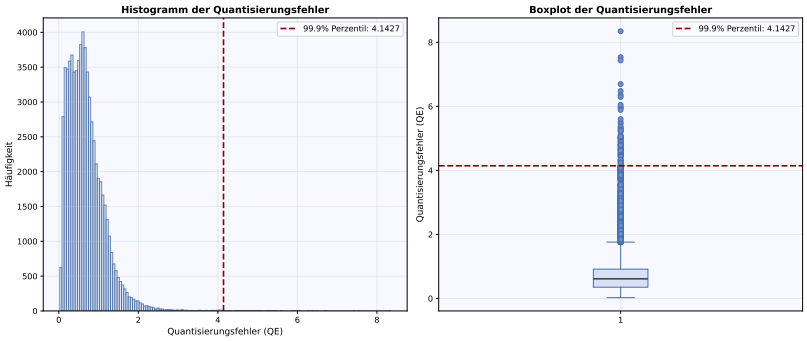


Abbildung 18.1: Histogramm der QE-Werte aller Datenpunkte zeigt die typische rechtsschiefe Verteilung. Die Verteilung zeigt, dass die meisten Datenpunkte geringe Quantisierungsfehler aufweisen (Median bei 0,4), während wenige extreme Ausreißer mit QE-Werten bis über 8 identifiziert werden können (Box-Plot).

Die Top10-Werte aus `"cardio_SOM_anomalien.csv"` sind:

Tabelle 18.1: Top 10 Datenpunkte mit den höchsten Quantisierungsfehlern

ID	QE-Wert	Alter	Größe	Gewicht	RR sys.	RR dia.
28605	8,345	19777	112	167,0	180	120
54282	7,535	21770	161	84,0	196	182
62861	7,434	22652	163	70,0	200	180
21958	6,694	17405	125	167,0	180	90
618	6,478	16765	186	200,0	130	70
6769	6,350	18961	158	74,0	200	170
82269	6,292	20627	157	76,0	180	20
71945	6,046	15117	180	200,0	150	90
8757	5,970	20990	122	161,0	120	80
7054	5,948	22722	173	74,0	220	160

18.2 U-Matrix (Unified Distance Matrix) und Outlier

Die U-Matrix wurde von Alfred Ultsch vorgeschlagen [35] und zählt zu den populärsten Methoden zur Visualisierung von SOMs, da sie die zugrunde liegenden Clusterstrukturen über die Abstände der Neuronen im Gitter intuitiv sichtbar macht.

Während der Quantisierungsfehler ein individuelles Maß für jeden Datenpunkt ist, bietet die U-Matrix eine globale Sicht auf die Struktur der trainierten Karte selbst. Sie visualisiert die Cluster-Struktur der Daten und kann indirekt zur Identifikation von Anomalien genutzt werden.

Die **Unified Distance Matrix (U-Matrix)** ist eine Visualisierung der SOM, die für jedes Neuron auf der Karte den durchschnittlichen Abstand zu den Gewichtsvektoren seiner direkten Nachbarn anzeigt. Das Ergebnis ist eine topografische Karte, die die Dichteverteilung der Daten repräsentiert.

Die U-Matrix berechnet für jedes Neuron i auf der Karte den mittleren Abstand zwischen dessen Gewichtsvektor $\mathbf{w}_i$ und den Gewichtsvektoren der direkten topologischen Nachbarn

$$\mathcal{N}(i) = \{j | d_A(k_i, k_j) = 1\}$$

. Dann ist der Eintrag in der Matrix für Neuron i:

$$U(i) = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \|\mathbf{w}_i - \mathbf{w}_j\|_2$$

Dabei ist $\mathbf{w}_i \in \mathbb{R}^d$ der Gewichtsvektor des Neurons i, $\mathcal{N}(i)$ ist die Menge der unmittelbaren Nachbarn von i gemäß dem topologischen Abstandsmaß $d_A(k_j, k_i) = 1$.

Die Interpretation einer U-Matrix ist analog zu einer geografischen Höhenkarte:

Eine U-Matrix kann auch zur Schätzung der Anzahl der Cluster in den Daten verwendet werden: Die Anzahl der „Täler“, getrennt durch „Bergketten“, gibt eine gute heuristische Schätzung.

Niedrige Werte (Täler): Diese Regionen entsprechen Neuronen, deren Gewichtsvektoren sich sehr ähnlich sind. Datenpunkte, die auf diese Neuronen abgebildet werden, gehören zu einem dichten, homogenen Cluster.

Hohe Werte (Berge): Diese Bereiche zeigen große Abstände zwischen benachbarten Neuronen an. Sie fungieren als Grenzen zwischen Clustern. Ein Neuron, das in einer solchen “Bergregion“ liegt, repräsentiert einen Bereich geringer Datendichte im Eingaberaum.

Ausreißer können auf zwei Arten mit der U-Matrix in Verbindung gebracht werden:

1. **Aktivierung von „Berg-Neuronen“:** Ein Datenpunkt, dessen BMU in einer Region mit hohen U-Matrix-Werten liegt, ist ein potenzieller Ausreißer. Er liegt in einem dünn besiedelten Bereich des Datenraums, zwischen den Hauptclustern.
2. **Aktivierung von selten genutzten Neuronen:** Manchmal bilden isolierte Anomalien ihr eigenes kleines „Hochplateau“oder aktivieren ein Neuron, das von keinem anderen Datenpunkt als BMU gewählt wird. Die Analyse der Aktivierungsfrequenz (Hit Count) der Neuronen in Kombination mit der U-Matrix kann solche Fälle aufdecken.

Die U-Matrix allein klassifiziert Datenpunkte nicht direkt als Anomalien, aber sie bietet einen entscheidenden visuellen Kontext. Sie zeigt, *warum* ein Punkt einen hohen Quantisierungsfehler haben könnte: weil er in einer Region liegt, die eine natürliche Lücke in den Daten darstellt.

Prompt für U-Matrix

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit “;“. Die bereits trainierten SOM-Gewichte sind in *cardio_SOM_Gewichte.npy* und die Skalierungsparameter in *cardio_scaler_mean.npy* und *cardio_scaler_scale.npy* gespeichert. Erstelle einen Python-Quellcode, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'` und `'ap_hi'` für die Analyse aus.
2. Lade die gespeicherten SOM-Gewichte aus *cardio_SOM_-*

Gewichte.npy.
3. Lade die Skalierungsparameter aus *cardio_scaler_mean.npy* und *cardio_scaler_scale.npy* und rekonstruiere den StandardScaler.
4. Skaliere die Daten mit dem rekonstruierten Scaler.
5. Rekonstruiere das MiniSom-Objekt mit den geladenen Gewichten (Gittergröße 35×35).
6. Erstelle eine U-Matrix (Farbe von niedrigen Werten dunkelblau nach hohen Werten hellblau). Gib die Farbscala rechts vom Plot an. Überlagere den Plot mit einem Contourplot (mit Levels 0.2,0.4,0.6, 0.8) in schwarzer Farbe.
7. Erstelle eine BMU-Hit-Matrix (Farbe von niedrigen Werten hellblau nach hohen Werten dunkelblau). Setze alle Werte oberhalb des 95%-Quantils der Hit-Verteilung auf diesen Schwellenwert (Clipping) Gib die Farbscala rechts vom Plot an.
8. Speichere die 2 Matrizen als 1x2-Plot unter `"cardio_SOM_U-Matrix.pdf"`.

Wichtig: Achte bei der Erstellung der Maps darauf, dass MiniSom Koordinaten als (x1, y1) zurückgibt, aber NumPy-Arrays als [row, col] = [y1, x1] indexiert werden. Neuron (0,0) bitte links unten.

Die Contour-Plots geben der Karte mehr "optische" Ordnung.

Durch das Clipping der Anzahl der Hits wird die Karte glatter und leichter zu lesen.

Prompt 18.2: Prompt für U-Matrix

Der Prompt wurde mit Grok Code Fast 1 als Python-Skript umgesetzt. Das Ergebnis ist:

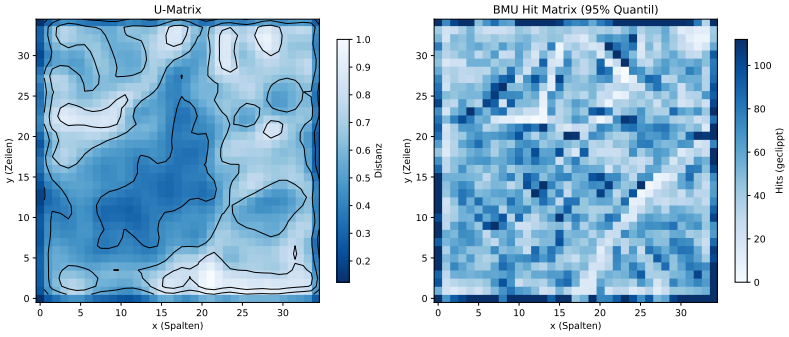


Abbildung 18.2: U-Matrix und BMU-Matrix. Die hellen Bereiche zeigen Regionen, in denen Neuronsgewichte einen großen Abstand haben. Insbesondere wenn eine solche helle Region, der zudem wenig Daten zugeordnet werden (BMU), unmittelbar an einen "dichten" Rand stößt, sind hier Outlier wahrscheinlich.

Die U-Matrix visualisiert die Nachbarschaftsdistanzen zwischen benachbarten SOM-Neuronen, wobei helle Bereiche große Abstände zwischen den Neuronengewichten anzeigen. Diese Regionen markieren typischerweise Cluster-Grenzen oder Übergangszonen im Datenraum.

Meist ist in der Literatur zu SOMs gerade anders herum: Dunkle Farben sind schwach besetzt. Inuitiver ist es, dichter mit kräftiger - wie hier - zu kennzeichnen.

Besonders aussagekräftig für die Outlier-Erkennung sind helle U-Matrix-Regionen mit geringer BMU-Zuordnung (wenige Datenpunkte), die unmittelbar an dicht besetzte Cluster-Bereiche angrenzen. Diese Konstellation deutet auf isolierte, extreme Datenmuster hin, die sich deutlich von den Hauptverteilungen unterscheiden und daher als potentielle Ausreißer klassifiziert werden können.

18.3 Zusammenfassung und Vor- und Nachteile

Wie jede Methode hat auch die Anomalieerkennung mit SOMs spezifische Stärken und Schwächen, die je nach Anwendungsfall abgewogen werden müssen.

Ein wesentlicher Vorteil ist die starke Visualisierungsfähigkeit. Methoden wie die U-Matrix ermöglichen eine intuitive visuelle Analyse der Datenstruktur. Cluster, ihre Grenzen und potenziell anomale Regionen werden auf einen Blick erkennbar, was das Verständnis für komplexe Datensätze erheblich erleichtert.

Ein weiterer Pluspunkt ist die Eignung für nicht-lineare Datenstrukturen. SOMs können komplexe, nicht-lineare Zusammenhänge im Datenraum abbilden, wo lineare Methoden wie PCA an ihre Grenzen stoßen.

Der vielleicht wichtigste Vorteil ist jedoch, dass es sich um ein unüberwachtes Verfahren handelt. Es werden keine gelabelten Trainingsdaten (d.h. keine im Voraus als „normal“oder „anomal“markierten Daten) benötigt, was in vielen realen Szenarien, in denen solche Labels selten oder teuer zu beschaffen sind, von entscheidender Bedeutung ist.

Die Eigenschaft des unüberwachten Lernens macht SOMs besonders wertvoll für die explorative Datenanalyse, bei der noch kein klares Verständnis der Datenstruktur oder möglicher Anomalien existiert.

Auf der anderen Seite stehen die Nachteile.

Das Training einer SOM kann, insbesondere bei großen Datensätzen und großen Karten, rechenintensiv sein. Der iterative Prozess, bei dem für jeden Datenpunkt die Abstände zu allen Neuronen berechnet werden, kann viel Zeit in Anspruch nehmen.

Mit der Nutzung moderner GPUs spielt dieser Aspekt eine immer kleinere Rolle. Zudem können über Plattformen wie Google Colab externe Rechnerkapazitäten genutzt werden. Dies ist ein Grund, warum SOMs seit den 1990er-Jahren eine Renaissance erfahren haben.

Ein weiterer kritischer Punkt ist die Sensitivität gegenüber der Wahl der Hyperparameter. Die Qualität der resultierenden Karte und damit die Güte der Anomalieerkennung hängen stark von der Wahl der Kartengröße, der Lernrate und des Nachbarschaftsradius ab. Die optimale Einstellung dieser Parameter erfordert oft Erfahrung und Experimente.

Schließlich kann die Interpretation subjektiv sein. Während die QE-Werte eine quantitative Grundlage bieten, erfordert die Interpretation der U-Matrix oder die Festlegung eines geeigneten Schwellenwerts oft eine subjektive Entscheidung, die von der Erfahrung des Anwenders abhängt. Es gibt keine universell gültige Regel, ab wann eine „Bergkette“in der U-Matrix signifikant ist oder ein QE-Wert „zu hoch“ist.

Trotz der Subjektivität in der Interpretation kann gerade die Visualisierung der U-Matrix Diskussionen im Team anregen und zu einem tieferen, gemeinsamen Verständnis der Daten führen, was über die reine Anomalieerkennung hinausgeht.

Local Outlier Factor (LOF) zur dichtenbasierten Anomalieerkennung

19

Der Local Outlier Factor (LOF) ist ein einflussreicher Algorithmus zur Erkennung von Ausreißern, der 2000 von Breunig, Kriegel, Sander (LMU München) und Ng (University of Britisch Columbia)vorgestellt wurde (vgl. [36] mit Link zum Download).

In [37] ist das Konzept ab Seite 132 dargestellt.

Seine besondere Stärke liegt in der Fähigkeit, Anomalien in Datensätzen mit variierender Dichte zu identifizieren. Ein Szenario, in dem globale Methoden oft versagen.

Dieses Kapitel erläutert die konzeptionellen Grundlagen des LOF, seine mathematische Formulierung und die praktische Anwendung.

19.1 Grundidee des LOF-Algorithmus

Die zentrale Innovation des LOF-Algorithmus liegt in seiner lokalen Perspektive. Im Gegensatz zu globalen Methoden, die einen Punkt mit dem gesamten Datensatz vergleichen, bewertet LOF die Ausreißereigenschaft eines Datenpunktes ausschließlich relativ zu seiner direkten Nachbarschaft.

Ein Punkt wird dann als Ausreißer klassifiziert, wenn seine lokale Dichte signifikant geringer ist als die Dichte seiner Nachbarn. Man kann sich dies so vorstellen, dass der Punkt isoliert von seiner Umgebung ist, während seine Nachbarn Teil einer dichteren Gruppe sind.

Das Ziel des LOF ist es, Anomalien auch in komplexen Datenstrukturen aufzuspüren. In vielen realen Datensätzen existieren Cluster mit stark unterschiedlichen Dichten. Ein Punkt, der am Rande eines sehr dichten Clusters liegt, könnte global gesehen eine hohe Dichte aufweisen. Eine globale Ausreißererkennung würde ihn daher übersehen. Lokal betrachtet, im Vergleich zu seinen extrem dichten Nachbarn, ist er jedoch ein klarer Ausreißer. Genau diese Fälle kann der LOF-Algorithmus zuverlässig identifizieren.

Globale Methoden definieren Ausreißer oft über einen einzigen Schwellenwert (vgl. z.B. die Mahalanobis-Distanz in Kapitel 14). Ein Punkt, der weiter als ein bestimmter Abstand vom Mittelwert entfernt ist, gilt als Ausreißer. Dies schlägt bei multiplen Clustern unterschiedlicher Dichte fehl.

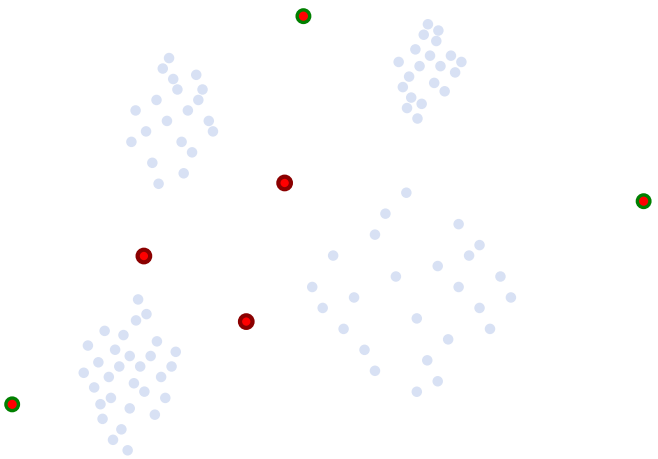


Abbildung 19.1: Konzeptionelle Darstellung der LOF-Stärke. Vergleich globaler vs. lokaler Ausreißererkennung: Globale Algorithmen erkennen hauptsächlichdie grün umrandeten Punkte, die weit vom Datenzentrum entfernt sind. LOF erkennt zusätzlichdie rot umrandeten Punkte, die lokal eine geringe Dichte aufweisen, obwohl sie global nicht extrem erscheinen.

Der LOF-Algorithmus gehört zur Familie der dichtenbasierten Clustering-Verfahren, ähnlich wie DBSCAN, erweitert diesen Ansatz jedoch explizit um ein Scoring für Ausreißer.

19.2 Die Kernkonzepte des LOF

Der LOF-Wert eines Datenpunktes wird nicht direkt berechnet, sondern basiert auf einer Kette von aufeinander aufbauenden Definitionen und Konzepten. Diese werden in den folgenden Abschnitten schrittweise eingeführt, um die finale Formel des LOF-Scores herzuleiten.

19.2.1 k-Distanz und k-Nachbarschaft

Das Fundament des LOF-Algorithmus bildet die Definition der lokalen Nachbarschaft eines Punktes. Diese wird durch den Parameter k gesteuert, der oft auch als *Minimum Points* bezeichnet wird. Er gibt an, wie viele Nachbarn zur Berechnung der lokalen Dichte verwendet werden.

Die Wahl von k ist entscheidend. Ein zu kleines k macht den Algorithmus anfällig für lokales Rauschen. Ein zu großes k führt dazu, dass der lokale Charakter des Algorithmus verloren geht und er sich einer globalen Methode annähert.

Sei A ein Datenpunkt in einem Datensatz $\mathcal{D}$.

Die **k-Distanz** von A, bezeichnet als $\mathbf{d_k(A)}$, ist die Distanz zum k-nächsten Nachbarn von A.

Die **k-Nachbarschaft** von A, bezeichnet als $\mathbf{N_k(A)}$, ist die Menge aller Punkte $B \in \mathcal{D}$ (einschließlich A selbst), deren Distanz zu A kleiner oder gleich der k-Distanz ist, d.h.

$$N_k(A) = \{B \in \mathcal{D} \mid d(A, B) \leq d_k(A)\}$$

.

Als Faustregel sollte k so gewählt werden, dass es die minimale Größe eines als legitim angesehenen Clusters repräsentiert.

Die Anzahl der Punkte in der k-Nachbarschaft, $|N_k(A)|$, kann größer als k sein, falls mehrere Punkte exakt dieselbe Distanz zum Punkt A haben wie der k-nächste Nachbar. Dies wird als „Distanz-Gleichstand'"(tie) bezeichnet.

Beispiel: Bestimmung von $d_3(A)$ und $N_3(A)$

Gegeben sei ein Punkt A und seine Nachbarn mit folgenden euklidischen Distanzen: $d(A, P1) = 2$, $d(A, P2) = 4$, $d(A, P3) = 5$, $d(A, P4) = 7$, $d(A, P5) = 7$.

Für $k = 3$ ist der drittnächste Nachbar $P3$. Die k-Distanz ist somit $d_3(A) = d(A, P3) = 5$.

Die k-Nachbarschaft $N_3(A)$ umfasst alle Punkte mit einer Distanz ≤ 5. Das sind die Punkte $P1, P2$ und $P3$. Also ist $N_3(A) = \{P1, P2, P3\}$.

19.2.2 Erreichbarkeitsdistanz (Reachability Distance)

Um die Dichtemessung zu stabilisieren und die statistischen Schwankungen bei sehr nahen Punkten zu reduzieren, führt LOF das Konzept der Erreichbarkeitsdistanz ein. Diese modifizierte Distanzmessung "glättet" gewissermaßen die Distanzen.

Die **Erreichbarkeitsdistanz** von Punkt A zu Punkt B bezüglich k, bezeichnet als $\textbf{reach-dist}_\textbf{k}(\textbf{A}, \textbf{B})$, ist definiert als das Maximum aus der k-Distanz von B und der tatsächlichen Distanz zwischen A und B.

$$\text{reach-dist}_k(A, B) = \max\{d_k(B), d(A, B)\}$$

Die Erreichbarkeitsdistanz ist asymmetrisch, d.h.,

$$\text{reach-dist}_k(A, B) \neq \text{reach-dist}_k(B, A)$$

im Allgemeinen. Streng genommen ist sie daher keine mathematische Distanz im klassischen Sinne, sondern eine asymmetrische Distanzfunktion.

Beispiel: Berechnung der Erreichbarkeitsdistanz

Gegeben seien die Punkte aus dem vorherigen Beispiel mit $k = 3$ und $d_3(A) = 5$.

Zusätzlich seien die k-Distanzen der Nachbarpunkte bekannt: $d_3(P1) = 3$, $d_3(P2) = 6$, $d_3(P3) = 4$.

Berechnung der Erreichbarkeitsdistanzen von A zu seinen Nachbarn:

$$\text{reach-dist}_3(A, P1) = \max\{d_3(P1), d(A, P1)\} = \max\{3, 2\} = 3$$
$$\text{reach-dist}_3(A, P2) = \max\{d_3(P2), d(A, P2)\} = \max\{6, 4\} = 6$$
$$\text{reach-dist}_3(A, P3) = \max\{d_3(P3), d(A, P3)\} = \max\{4, 5\} = 5$$

Beobachtung: Obwohl $P1$ der nächste Nachbar von A ist (Distanz 2), wird die Erreichbarkeitsdistanz auf 3 erhöht, da $d_3(P1) = 3$. Dies verhindert eine künstlich hohe Dichtemessung aufgrund der sehr geringen Distanz.

Die Intuition dahinter ist, dass wenn A sehr nahe an B liegt und B sich in einem dichten Cluster befindet (also $d_k(B)$ klein

Diese Glättung ist besonders wichtig, um zu verhindern, dass ein Punkt A einen unnatürlich hohen Dichte-Wert erhält, nur weil er zufällig extrem nah an einem Nachbarn B liegt.

ist), die Erreichbarkeitsdistanz effektiv die tatsächliche Distanz $d(A, B)$ ist. Dies verhindert, dass Punkte, die im Inneren eines dichten Clusters liegen, übermäßig kleine Distanzwerte zugewiesen bekommen, was die anschließende Dichteberechnung verzerren würde. Nachfolgende Abbildung soll dies verdeutlichen:

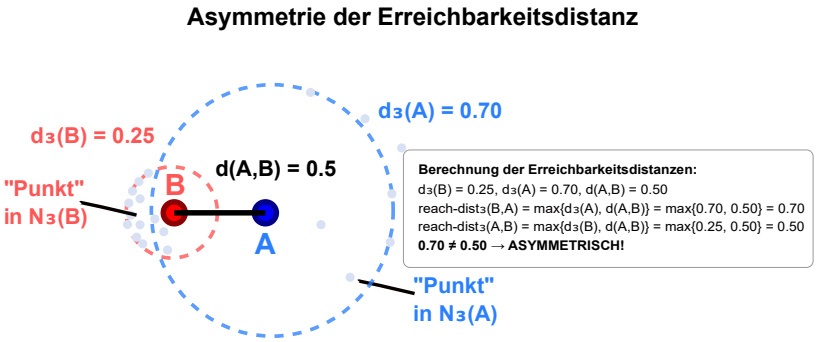


Abbildung 19.2: Konzeptionelle Darstellung der Erreichbarkeitsdistanz. Punkt B liegt in einem dichten Cluster. Er bekommt die tatsächlich Distanz zum Punkt B zugewiesen. Punkt A dagegen ist in einem isolierten Bereich. Hier wird die k-Distanz zugewiesen. Dies zeigt auch die Asymmetrie der Erreichbarkeitsdistanz.

19.2.3 Lokale Erreichbarkeitsdichte

Aufbauend auf der Erreichbarkeitsdistanz wird die lokale Dichte eines Punktes definiert.

Die Lokale Erreichbarkeitsdichte (lrd) ist im Wesentlichen der Kehrwert der durchschnittlichen Distanz, die ein Punkt zu seinen Nachbarn hat.

Historisch gesehen revolutionierte LOF die Ausreißererkennung, weil es das erste Verfahren war, das eine quantitative Bewertung der „Ausreißerhaftigkeit'auf einer lokalen, relativen Basis ermöglichte.

Die **lokale Erreichbarkeitsdichte** eines Punktes A, bezeichnet als $\mathbf{lrd_k(A)}$, ist der Kehrwert der durchschnittlichen Erreichbarkeitsdistanz von A zu all seinen Nachbarn in $N_k(A)$.

$$\text{lrd}_k(A) = \frac{1}{\frac{\sum_{B \in N_k(A)} \text{reach-dist}_k(A,B)}{|N_k(A)|}} = \frac{|N_k(A)|}{\sum_{B \in N_k(A)} \text{reach-dist}_k(A,B)}$$

Ein hoher $\text{lrd}_k(A)$-Wert bedeutet, dass die durchschnittliche Erreichbarkeitsdistanz zu den Nachbarn klein ist. Der Punkt A befindet sich also in einer dichten Region.

Ein niedriger $\text{lrd}_k(A)$-Wert hingegen deutet auf eine große durchschnittliche Erreichbarkeitsdistanz hin, was bedeutet, dass der Punkt A in einer dünn besiedelten oder isolierten Region liegt.

19.3 LOF-Scores

Der finale Schritt ist die Berechnung des eigentlichen Local Outlier Factors. Der LOF-Score setzt die Dichte eines Punktes ins Verhältnis zur Dichte seiner Nachbarn.

Der **LOF-Score** eines Punktes A ist das Verhältnis der durchschnittlichen lrd_k-Werte seiner Nachbarn zur eigenen $\text{lrd}_k(A)$:

$$\text{LOF}_k(A) = \frac{\frac{\sum_{B \in N_k(A)} \text{lrd}_k(B)}{|N_k(A)|}}{\text{lrd}_k(A)}.$$

Die relative Natur des LOF-Scores ist seine größte Stärke. Er ist nicht von der absoluten Dichte eines Clusters abhängig, sondern nur vom Dichtekontrast zwischen einem Punkt und seiner Umgebung.

Diese Formel ist der Kern des Algorithmus. Sie quantifiziert, um wie viel die Dichte eines Punktes geringer (oder höher) ist als die Dichte seiner unmittelbaren Umgebung.

19.3.1 Interpretation der LOF-Werte

Die Interpretation des resultierenden LOF-Scores ist intuitiv und liefert eine klare Indikation über die Ausreißereigenschaft eines Punktes. Man unterscheidet typischerweise drei Fälle:

LOF $\approx$ 1:
Ein Wert nahe 1 bedeutet, dass die lokale Dichte des Punktes A ($\text{lrd}_k(A)$) ungefähr der durchschnittlichen lokalen Dichte seiner Nachbarn entspricht. Der Punkt ist also Teil einer Region mit homogener Dichte und wird als typischer **Inlier** betrachtet. Er liegt vollständig innerhalb eines Clusters.

In der Praxis gibt es keinen universellen Schwellenwert für einen "guten" LOF-Score. Oft werden die Punkte mit den höchsten LOF-Werten (z.B. die Top 1%) als die relevantesten Ausreißer betrachtet und genauer untersucht.

LOF > 1:
Ein Wert signifikant größer als 1 deutet darauf hin, dass die lokale Dichte von A wesentlich geringer ist als die seiner Nachbarn. Der Punkt ist also weiter von seinen Nachbarn entfernt, als diese untereinander entfernt sind. Er ist ein potenzieller **Ausreißer**. Je höher der LOF-Wert, desto stärker ist die Ausreißereigenschaft ausgeprägt. Ein LOF-Wert von 2 würde beispielsweise bedeuten, dass die lokale Dichte des Punktes nur halb so groß ist wie die seiner Nachbarn.

LOF < 1:
Ein Wert kleiner als 1 ist ebenfalls möglich. Er bedeutet, dass die lokale Dichte von A höher ist als die seiner Nachbarn. Dies ist der Fall für Punkte, die sich im inneren Kern eines besonders dichten Clusters befinden. Diese Punkte sind gewissermaßen das Gegenteil von Ausreißern.

19.4 Praktische Anwendung und Überlegungen

Die Implementierung des LOF-Algorithmus erfordert einige praktische Überlegungen, insbesondere bei der Wahl

des Parameters k und im Umgang mit hochdimensionalen Daten.

19.4.1 Wahl des Parameters k

Die wichtigste und oft schwierigste Entscheidung bei der Anwendung von LOF ist die Wahl von k, der Anzahl der zu berücksichtigenden Nachbarn.

Es gibt keine goldene Regel, aber einige Heuristiken und Überlegungen.

Ein zu kleiner Wert für k kann den Algorithmus sehr anfällig für lokales Rauschen machen, da einzelne zufällige Punkte bereits die Nachbarschaftsdefinition stark beeinflussen können.

Ein zu großer Wert für k hingegen kann dazu führen, dass die lokale Perspektive verloren geht. Wenn k sehr groß gewählt wird, umfassen die Nachbarschaften große Teile der Cluster, und der Algorithmus beginnt, sich wie eine globale Methode zu verhalten, wodurch seine Fähigkeit, lokale Ausreißer in dichten Clustern zu finden, abnimmt.

In der Literatur wird oft ein Bereich für k empfohlen, z.B. $k \in [10, 50]$. Eine gängige Praxis ist es, den Algorithmus für verschiedene Werte von k laufen zu lassen und die Stabilität der Ergebnisse zu analysieren.

Das ursprüngliche LOF-Paper empfiehlt $k \geq 10$ zur Vermeidung von Fluktuationen [36]. In der Praxis wird oft $k = 20$ als Standardwert verwendet [38]. Die Stabilität der Ausreißer über verschiedene k-Werte hinweg ist ein Indiz für deren Robustheit.

19.5 Vor- und Nachteile

Der LOF-Algorithmus bietet entscheidende Vorteile.

Er ist sehr effektiv in Datensätzen mit variierender Dichte, in denen globale Methoden versagen. Zudem erfordert er keine Annahmen über die zugrundeliegende Datenverteilung (z.B. Normalverteilung), was ihn sehr flexibel einsetzbar macht.

Dem stehen jedoch auch Nachteile gegenüber.

Der Rechenaufwand ist mit einer Komplexität von $O(n^2)$ relativ hoch, da für jeden Punkt die Distanzen zu allen anderen Punkten berechnet werden müssen, um die Nachbarschaften zu finden. Für sehr große Datensätze kann dies prohibitiv sein.

Ein weiteres Problem ist der bereits erwähnte Fluch der Dimensionalität. Bei Daten mit sehr vielen Merkmalen kann

Der **Fluch der Dimensionalität** (Curse of Dimensionality) beschreibt das Phänomen, dass in hochdimensionalen Räumen das Distanzkonzept an Aussagekraft verliert. Der Abstand zwischen dem nächsten und dem entferntesten Nachbarn eines Punktes nähert sich an, wodurch dichtenbasierte Methoden wie LOF erschwert werden.

die zugrundeliegende Distanzmessung (z.B. euklidische Distanz) ihre Aussagekraft verlieren, was die Leistung des LOF beeinträchtigt.

19.6 LOF für kategoriale Daten

Während viele Outlier-Detection-Verfahren wie PCA-basierte Ansätze, Autoencoder und die Mahalanobis-Distanz ausschließlich auf numerische Daten beschränkt sind, bietet das konzeptuelle Framework von LOF eine bemerkenswerte Flexibilität.

Der Grund hierfür liegt in der fundamentalen Architektur des Algorithmus: LOF basiert primär auf dem Konzept der Nachbarschaft und der relativen Dichteschätzung, nicht auf spezifischen mathematischen Operationen wie Matrixinversionen oder Gradienten, die numerische Werte voraussetzen.

Der entscheidende Schlüssel für diese Erweiterung liegt in der Erkenntnis, dass LOF lediglich ein Distanz- oder Ähnlichkeitsmaß zwischen Datenpunkten benötigt, um Nachbarschaften zu definieren und lokale Dichten zu berechnen.

Diese konzeptuelle Klarheit ermöglicht es, LOF durch die Wahl eines geeigneten Distanzmaßes auf verschiedenste Datentypen zu erweitern.

19.6.1 Die Gower-Distanz als universelle Lösung

Die Gower-Distanz, entwickelt von John C. Gower im Jahr 1971 ([39]), stellt eine elegante Lösung für die Herausforderung gemischter Datentypen dar.

Ihr fundamentales Prinzip besteht darin, verschiedene Variablentypen durch typspezifische Distanzberechnungen zu behandeln und diese anschließend in einem gewichteten Durchschnitt zu kombinieren.

Dadurch entsteht ein einheitliches Distanzmaß mit einem normierten Wertebereich zwischen null und eins.

Die **Gower-Distanz** zwischen zwei Objekten i und j ist definiert als:

$$d_{\text{Gower}}(i,j) = \frac{\sum_{k=1}^{p} w_{ijk} \cdot d_{ijk}}{\sum_{k=1}^{p} w_{ijk}} \tag{19.1}$$

wobei d_{ijk} die variablenspezifische Distanz zwischen den Objekten i und j für die Variable k darstellt und w_{ijk} das entsprechende Gewicht ist.

Für numerische Variablen erfolgt die Distanzberechnung durch Normalisierung auf den Wertebereich der jeweiligen Variable.

Konkret wird die absolute Differenz durch die Spannweite der Variable geteilt: $d_{ijk} = |x_{ik} - x_{jk}|/R_k$, wobei $R_k = \max(x_k) - \min(x_k)$ die Spannweite der Variable k bezeichnet.

Diese Normalisierung stellt sicher, dass alle numerischen Variablen unabhängig von ihrer ursprünglichen Skalierung gleichmäßig zum Gesamtdistanzmaß beitragen.

Kategoriale Variablen werden hingegen binär behandelt. Die Distanz beträgt null, wenn beide Objekte denselben Kategorienwert aufweisen, und eins, wenn sie sich unterscheiden: $d_{ijk} = 0$ falls $x_{ik} = x_{jk}$, sonst $d_{ijk} = 1$. Diese simple aber effektive Behandlung spiegelt die Natur kategorialer Daten wider, bei denen keine natürliche Ordnung oder metrische Struktur existiert.

19.6.2 Veranschaulichendes Beispiel

Die praktische Anwendung der Gower-Distanz lässt sich anhand eines konkreten Beispiels verdeutlichen.

Betrachtet wird ein Datensatz mit Mitarbeiterprofilen, der sowohl numerische als auch kategoriale Informationen enthält.

Die Tabelle zeigt acht Mitarbeiter mit den Attributen Alter (numerisch), Jahresgehalt (numerisch), Abteilungszugehörigkeit (kategorial) und Geschlecht (kategorial).

Tabelle 19.1: Beispieldatensatz mit gemischten Datentypen

Person	Alter	Gehalt	Abteilung	Geschlecht
A	25	45.000€	IT	M
B	30	50.000€	IT	W
C	35	55.000€	HR	W
D	28	48.000€	IT	M
E	42	65.000€	Finance	M
F	29	46.000€	Finance	W
G	38	58.000€	HR	M
H	26	47.000€	IT	W

Die Berechnung der Gower-Distanz zwischen Person A und Person B erfolgt schrittweise.

Zunächst werden die numerischen Variablen normalisiert. Für das Alter ergibt sich die Spannweite als $R_{\text{Alter}} = 42 - 25 = 17$, woraus die normalisierte Distanz $d_{\text{Alter}}(A, B) = |25 - 30|/17 \approx 0.294$ folgt.

Entsprechend beträgt für das Gehalt die Spannweite $R_{\text{Gehalt}} = 65.000 - 45.000 = 20.000$, und die normalisierte Distanz errechnet sich als $d_{\text{Gehalt}}(A, B) = |45.000 - 50.000|/20.000 = 0.25$.

In Scikit-Learn kann LOF mit Gower-Distanzen über `metric='precomputed'` verwendet werden.

Für die kategorialen Variablen erfolgt die binäre Bewertung. Da beide Personen der IT-Abteilung angehören, beträgt $d_{\text{Abteilung}}(A, B) = 0$.

$d_{\text{Geschlecht}}(A, B)$ ist 1, da sich das Geschlecht unterscheidet.

Die Gower-Distanz normalisiert alle Variablen auf den Wertebereich $[0, 1]$ und verhindert dadurch, dass Variablen mit größeren numerischen Bereichen die Distanzberechnung dominieren.

Die finale Gower-Distanz ergibt sich als gewichteter Durchschnitt: $d_{\text{Gower}}(A, B) = (0.294 + 0.25 + 0 + 1)/4 \approx 0.386$.

Dieser Wert spiegelt wider, dass die Personen A und B in einigen Aspekten ähnlich sind (gleiche Abteilung), sich aber in anderen unterscheiden (Geschlecht und numerische Werte).

Warnung: Gower-Distanzmatrix bei großen Datensätzen

Die Berechnung einer vollständigen Gower-Distanzmatrix hat eine quadratische Speicherkomplexität von $O(n^2)$. Bei großen Datensätzen kann dies zu erheblichen Performance- und Speicherproblemen führen:

Beispiel-Speicherverbrauch:

- 10.000 Datenpunkte: ≈ 800 MB RAM
- 50.000 Datenpunkte: ≈ 20 GB RAM
- 100.000 Datenpunkte: ≈ 80 GB RAM

Empfohlene Grenzwerte:

- Für Standard-Hardware: maximal 10.000 Datenpunkte
- Bei begrenztem RAM (< 16 GB): maximal 5.000 Datenpunkte
- Für größere Datensätze: Sampling oder approximative Methoden verwenden

Die Berechnungszeit steigt ebenfalls quadratisch an. Ein Datensatz mit 50.000 Punkten benötigt 25-mal mehr Zeit als ein Datensatz mit 10.000 Punkten.

19.6.3 Integration der Gower-Distanz in den LOF-Algorithmus

Die Asymmetrie der Erreichbarkeitsdistanz bleibt auch bei Verwendung der Gower-Distanz erhalten, da sie auf unterschiedlichen lokalen k-Distanzen basiert, nicht auf der zugrundeliegenden Distanzfunktion.

Die Einbindung der Gower-Distanz in den LOF-Algorithmus erfolgt nahtlos durch Substitution des Distanzmaßes in allen relevanten Berechnungsschritten. Der konzeptuelle Ablauf

des Algorithmus bleibt dabei vollständig erhalten, lediglich die zugrundeliegende Distanzfunktion wird ausgetauscht.

Diese nahtlose Integration demonstriert die konzeptuelle Stärke des LOF-Frameworks. Während andere Outlier-Detection-Verfahren durch ihre mathematische Struktur auf bestimmte Datentypen beschränkt sind, ermöglicht die distanzmaßunabhängige Natur von LOF eine flexible Anpassung an diverse Datenanforderungen.

Die Gower-Distanz stellt nur eine von vielen möglichen Erweiterungen dar – prinzipiell können beliebige problemspezifische Distanzfunktionen verwendet werden, solange sie die grundlegenden Eigenschaften einer Metrik erfüllen oder zumindest eine sinnvolle Ähnlichkeitsmessung ermöglichen.

To Do Durchführung einer LOF-Analyse

1. **Datenvorverarbeitung:**
   - Bei **rein numerischen Daten:** Stelle sicher, dass alle Merkmale eine angemessene Skalierung aufweisen (z.B. Standardisierung), da die Distanzmessung sonst von Merkmalen mit großen Wertebereichen dominiert wird.
   - Bei **gemischten Datentypen:** Berechne zunächst die Gower-Distanzmatrix, die automatisch numerische Variablen normalisiert und kategoriale Variablen binär behandelt.
2. **Parameterwahl für *k*:** Wähle $k = 20$. Alternativ: Teste einen sinnvollen Bereich für k, beispielsweise von $k = 15$ bis $k = 40$. Analysiere, welche Datenpunkte über verschiedene k-Werte hinweg konsistent hohe LOF-Scores aufweisen.
3. **Algorithmus-Anwendung:**
   - **Numerische Daten:** Verwende LOF mit euklidischer Distanz oder anderen metrischen Distanzmaßen.
   - **Gemischte Daten:** Setze LOF mit `metric='precomputed'` und der zuvor berechneten Gower-Distanzmatrix ein.
4. **Ergebnis-Interpretation:** Sortiere die Ergebnisse nach dem LOF-Score in absteigender Reihenfolge. Untersuche die Top-N Ausreißer (z.B. die obersten 1–5%) manuell, um die Ursache für ihre Anomalie zu verstehen.
5. **Visualisierung:**
   - **LOF-Score-Verteilung:** Erstelle ein Histogramm der LOF-Werte, um die Gesamtverteilung zu verstehen. Normale Punkte sollten sich um LOF ≈ 1 clustern, während Ausreißer deutlich höhere Werte aufweisen.
   - **Ausreißer-Identifikation:** Verwende einen BoxPlot der LOF-Scores zur statistischen Ausreißererkennung. Punkte außerhalb der Whisker (> Q3 + 1.5· IQR) können als potentielle Anomalien betrachtet werden.

19.7 Praxisbeispiel: Cardiodaten

Abschließend soll der LOF-Algorithmus auf das schon bekannte Beispiel aus [9] angewandt werden:

Prompt für LOF Anomalieerkennung Cardiovascular Dataset

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv* enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen Python-Quellcode mit scikit-learn und LOF, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'`, `'ap_hi'`.
2. Skaliere die Daten mit dem StandardScaler
3. Trainiere den Local Outlier Factor mit folgenden Parametern mit Anzahl der Nachbarn (n_neighbors): 20
4. Berechne die LOF-Scores für alle Datenpunkte mit `-clf.negative_outlier_factor_`
5. Identifiziere die 100 Datenpunkte mit den höchsten LOF-Scores (höchste Anomalität) und speichere die entsprechenden Datenpunkte (mit allen verfügbaren Spalten) geordnet nach absteigenden LOF-Scores in `"cardio_LOF_-anomalien.csv"`

Hier kann man auch alle Spalten ohne 'id' wählen, mit dem Hinweis, dass zum Teil kategoriale Daten Dabei sind und daher *metric='precomputed'* mit Glower verwendet werden soll.

scikit-learn verwendet normalerweise negative Werte für Outlier - je negativer desto mehr Outlier um verschiedene Outlier-Algortihmen vergleichbar zu machen. Hier soll aber auf die "originalen Werte" zurückgegriffen werden.

Prompt 19.1: Prompt für LOF Anomalieerkennung Cardiovascular Dataset

Die Top-10 Outlier haben auf den ersten Blick nichts mit Ausreißern zu tun:

ID	Alter (Tage)	Größe	Gewicht	RR sys.	RR dia.	LOF-Score
35732	21952	165	65.0	115	80	9.686
70668	19013	165	65.0	125	80	9.309
43842	10859	159	59.0	120	80	7.710
7349	22107	165	65.0	125	80	7.116
49159	18188	165	65.0	120	82	6.876
79749	10964	160	59.0	110	70	5.827
94562	21956	160	60.0	120	84	5.413
91843	22098	166	65.0	115	80	5.311
61286	22109	165	66.0	115	80	5.286
13951	21622	165	65.0	120	82	5.253

Tabelle 19.2: Top 10 Datenpunkte mit den höchsten LOF-Scores

Die Erklärung ist, dass LOF die lokale Dichte relativ zu den Nachbarn misst (vgl. Abbildung 19.1 und Abbildung 19.2).

Bei den Cardiodaten sind der systolische und diastolische Blutdruck fast immer auf 10er-Einheiten gerundet (vgl. Abbildung 8.2 in Kapitel 8: "Feature Reliability Score"). Sie liegen somit in dichten Clustern vor. LOF identifiziert die Ausreißer, bei denen das nicht der Fall ist (z.B. keine Rundung auf 10er-Einheiten) oder die Randpunkte der dichten Cluster darstellen.

Dies zeigt, dass LOF eine wertvolle Ergänzung zu anderen Outlier-Detektionsmethoden ist, wobei die Interpretation jedoch wesentlich schwieriger sein kann.

Merke

Ein wichtiges Charakteristikum von LOF ist, dass es im Gegensatz zu anderen Outlier-Detektionsverfahren wie PCA-basierten Methoden oder der Mahalanobis-Distanz nicht nur isolierte Randpunkte identifiziert, sondern auch Punkte innerhalb von Clustern als Anomalien erkennen kann. LOF erfasst damit auch *lokale* Anomalien durch den Vergleich der Punktdichte in der unmittelbaren Nachbarschaft. Diese Eigenschaft macht LOF besonders wertvoll für die Erkennung subtiler Anomalien in komplexen Datenstrukturen mit variierenden Clusterdichten, die von globalen Verfahren übersehen werden könnten.

19.8 Zusammenfassung

Der Local Outlier Factor (LOF) ist eine fundamentale und leistungsstarke Methode zur dichtenbasierten Anomalieerkennung. Seine Stärke liegt in der lokalen, relativen Bewertung der Dichte eines Punktes im Vergleich zu seiner direkten Umgebung.

Die Kernkonzepte des Algorithmus bauen logisch aufeinander auf:

Ausgehend von der k-Nachbarschaft wird die Erreichbarkeitsdistanz eingeführt, um die Distanzmessung zu stabilisieren. Daraus wird die lokale Erreichbarkeitsdichte (LRD) als Maß für die lokale Punktdichte berechnet.

Der finale LOF-Score ergibt sich schließlich aus dem Verhältnis der durchschnittlichen LRD der Nachbarn zur LRD des Punktes selbst. Ein Score größer als 1 deutet auf einen potenziellen Ausreißer hin.

Obwohl der Algorithmus rechenintensiv sein kann , ist er aufgrund seiner Flexibilität (andere Distanzmaße) und seiner Fähigkeit, Ausreißer in Clustern variierender Dichte zu finden, ein unverzichtbares Werkzeug in der modernen Datenanalyse und im maschinellen Lernen.

Isolation Forest: Anomalieerkennung durch Isolierung

20

Während viele traditionelle Methoden auf Distanz- oder Dichtemessungen basieren, um Ausreißer zu finden, verfolgt der Isolation-Forest-Algorithmus einen fundamental anderen und oft effizienteren Ansatz.

Er beruht auf mehreren zufälligen Trennungen von "subsamples" der Daten auf Basis der verfügbaren Features/Datenspalten. Punkte, die schnell getrennt werden können, sind weitestgehend von den anderen Punkten isoliert. Dies ist ein starkes Indiz für einen Ausreißer.

Der Isolation Forest wurde im Paper von Liu et al. (2008) als erster Algorithmus vorgestellt, der explizit auf dem Prinzip der Isolierung basiert ([40]). In [41] wurde das Verfahren von den Autoren erweitert.

20.1 Grundidee und Funktionsprinzip

Das Prinzip des Isolation Forest basiert auf einer einfachen, aber wirkungsvollen Beobachtung:

Anomalien sind typischerweise „wenige und anders" (*few and different*). Aufgrund dieser Eigenschaft sind sie anfälliger für einen Prozess der Isolierung.

Die Idee, dass Anomalien „anders" sind, bedeutet, dass ihre Merkmalswerte oft außerhalb der üblichen Wertebereiche der normalen Datenpunkte liegen. „Wenige" bezieht sich auf ihre geringe relative Häufigkeit.

Genau dieses Isolationsprinzip macht sich der Algorithmus zunutze. Anstatt Profile für normale Datenpunkte zu erstellen und zu sehen, was nicht passt, versucht der Isolation Forest explizit, jeden einzelnen Datenpunkt zu isolieren.

Die Annahme ist, dass anomale Punkte im Durchschnitt mit deutlich weniger Schritten isoliert werden können als normale Punkte.

20.1.1 Der Isolation Tree (iTree)

Grundlage ist der binäre Baum

Ein **binärer Baum** ist eine gerichtete, azyklische Struktur $T = (V, E, r)$ mit einer ausgezeichneten Wurzel $r \in V$, bei der jeder Knoten höchstens zwei Nachfolger besitzt. Zu jedem Knoten $v \in V$ existiert genau ein eindeutiger Pfad von der Wurzel r zu v.

Binäre Bäume sind ein fundamentales Konzept der Informatik und finden Anwendung in vielen Bereichen: Binary Search Trees für effiziente Sortierung und Suche, Merkle Trees (Hash-Bäume) in Blockchain-Technologien wie Bitcoin. Isolation Forest nutzt jedoch zufällige statt optimierte Splits für die Anomalie-Erkennung.

Die reine formale Definition wird durch eine Abbildung deutlicher. Dabei ist $V = \{r, v_1, v_2, v_3, v_4\}$ und $E = \{(r, v_1), (r, v_2), (v_1, v_3), (v_1, v_4)\}$:

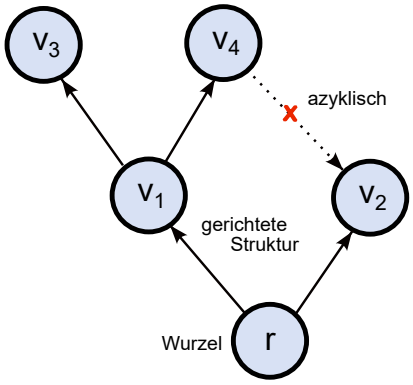


Abbildung 20.1: Beispiel eines binären Baums

Im Algorithmus des Isolation Forest heißt ein binärer Baum **iTree**. Dies soll hervorheben, dass der binäre Baum - anders als bei den meisten anderen Anwendungen - zufällig und nicht optimiert aufgebaut wird.

20.1.2 Konstruktion des iTree

Ausgangspunkt sind die verfügbaren Daten $\mathcal{D} \subset \mathbb{R}^d$ der Dimension d, d.h. es gibt d verfügbare Features.

Ein iTree wird in einer **Traingsphase** erstellt. Dazu werden folgende Schritte durchgeführt:

$256 = 2^8$ - wenn die Daten perfekt balanciert sind, hat man eine erwartete Tiefe des Baumes von ungefähr 8.

Auswahl eines "subsamples": Es werden zufällig N Datensätze aus $\mathcal{D}$ ausgewählt. Als Heuristik hat sich hier $N = 256$ bewährt ([41], Seite 23). Diese Daten liegen jetzt auf dem Root-Knoten des Baumes.

Die Auswahl des "Feature" erfolgt mit Zurücklegen.

Ermittlung der "children"-Knoten: Es wird aus den d verfügbaren Features bzw. Datenspalten eines zufällig ausgewählt (z.B. i) . Auf jedem Knoten wird für das zufällig gewählte Merkmal der Wertebereich (Minimum und Maximum) der auf diesem Knoten liegenden Teildaten des Subsamples bestimmt. Innerhalb dieses Bereichs wird ein zufälliger Trennwert $x \in \mathbb{R}$ gewählt. Auf Basis dieses Wertes werden die Daten auf die nächsten beiden Knoten verteilt, je nachdem, ob die entsprechende Ausprägung i des Datensatzes kleiner x oder größer gleich x ist.

Ein Knoten, der selbst keine "Kinder" mehr hat, heißt **Blattknoten**.

Weiterer Aufbau des iTree: Das Vorgehen wird suzessive mit den auf den Knoten zugewiesen Teilmengen des Subsamples wiederholt. Sobald in einem Knoten nur mehr ein Datensatz liegt (Isolierung), wird der Baum an dieser Stelle nicht mehr weiterentwickelt.

Größe der iTrees

Für eine Stichprobe der Größe $N = 256$ ergeben sich für die Tiefe des Baumes (d.h. bis alle Daten aus dem Subsample einen Blattknoten erreicht haben) folgende Grenzen:

Best Case: $\lceil \log_2(N) \rceil = 8$,
Worst Case: $N - 1 = 255$,
Expected Case: $c(N) \approx 10{,}24$.

$\lceil \cdot \rceil$ = Aufrundung zur nächsten ganzen Zahl.

$c(N)$ ist die erwartete Pfadlänge eines zufälligen binären Baums und ist definiert durch

$$c(N) = 2H(N-1) - \frac{2(N-1)}{N},$$

wobei $H(i) = \sum_{k=1}^{i} \frac{1}{k}$ die harmonische Zahl ist.

Die strikte Zufälligkeit bei der Auswahl von Merkmal und Trennwert ist entscheidend. Sie verhindert, dass sich die Bäume auf bestimmte, dominante Merkmale spezialisieren.

In jedem Knoten des Baumes geschieht dies durch zwei zufällige Wahlen:

1. Es wird ein zufälliges Merkmal (Feature/Datenfeld) aus der Menge der verfügbaren Merkmale ausgewählt (mit Zurücklegen).
2. Es wird ein zufälliger Trennwert (Split-Point) innerhalb des Wertebereichs dieses Merkmals (zwischen dem Minimum und Maximum der im aktuellen Knoten vorhandenen Datenpunkte) gewählt.

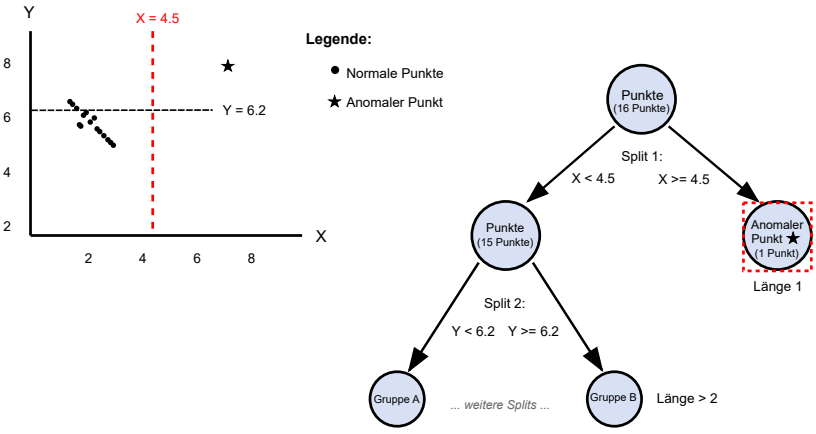


Abbildung 20.2: Schematischer Aufbau eines Isolation Tree (iTree). Ein anomaler Punkt (Stern) wird durch einen zufälligen Split auf der X-Achse schnell von der dichten Wolke normaler Punkte getrennt und hat daher eine sehr kurze Pfadlänge vom Wurzel- zum Blattknoten.

Ein Baum könnte über den gesamten Datensatz aufgebaut werden. Das wäre aber sowohl rechenintensiv, auch käme es zu einem Art Overfitting der Anomalieerkennung, bei der zu viele Datenpunkte als Outlier identifiziert würden (Swamping).

Die Effizienz des Isolation Forest ergibt sich daraus, dass für den Aufbau der Bäume nicht der gesamte Datensatz verwendet wird, sondern nur Stichproben fester Größe, typischerweise 256 Datensätze. Das zugrunde liegende Prinzip beruht darauf, dass sich der Datenraum bereits mit wenigen zufälligen "subsamples" gut beschreiben lässt.

20.1.3 Aufbau eines Isolation Forest

Wie oben beschrieben werden auf Basis unabhängig voneinander gezogener "subsamples" der Größe N aus $\mathcal{D}$ verschiedene iTrees aufgebaut. Die so konstruierten binären Bäume bilden den **Isolation Forest**.

Die Kombination vieler solcher Bäume, die jeweils auf unterschiedlichen Teildaten trainiert werden, sorgt für ein robustes und stabiles Ergebnis.

Ein **Isolation Forest** ist ein Ensemble von m zufällig erzeugten Isolation Trees (iTrees).

Das Ensemble-Prinzip ist die Grundlage vieler erfolgreicher Machine-Learning-Modelle. Der bekannteste Verwandte ist der Random Forest für Klassifikations- und Regressionsaufgaben, der ebenfalls auf einer Vielzahl von zufällig erstellten Bäumen basiert. Isolation Forest ist aber ein unüberwachter Algorithmus, d.h. er benötigt kein Labeling.

Ein Isolation Forest besteht normalerweise aus 100 bis 500 Bäumen.

20.2 Evaluierungsphase

Nachdem der Isoaltion Forrest aus den Subsamples aufgebaut wurde, kann er genutzt werden, um potentielle Ausreißer und Anomalien zu erkennen.

20.2.1 Pfadermittlung pro Datenpunkt

Mathematisch betrachtet teilt jeder iTree den Datenraum suzessive und zufällig in kleinere Subräume auf. Wenn dabei in einem Subraum nur ein Datensatz aus dem Subsample liegt, dann wird der Raum nicht mehr weiter aufgeteilt (**Blatt-Knoten**). Wenn in der Evaluierungspahse ein Punkt relativ schnell in einem Blatt-Knoten endet (kleiner Pfad), dann ist er wahrscheinlich in einem "dünnen", aber noch relativ großen Unterraum.

Jeder Datensatz x aus dem Gesamtdatensatz $\mathcal{D}$ durchläuft jeden verfügbaren iTree. Dabei wird jeweils die Pfadlänge $h(x)$ gemessen, bis x an einen Endknoten/Blattknoten gelangt.

Im Kontext von Isolation Forest entspricht die **Pfadlänge** $h(x)$ eines Datenpunkts x der Anzahl der Splits (Kanten), die durchlaufen werden müssen, um den Punkt von der Wurzel bis zu einem Blattknoten zu isolieren.

Indem die Ergebnisse – genauer gesagt die Pfadlängen $h(x)$ für jeden Datenpunkt – über alle Bäume im Forest gemittelt werden, werden die zufälligen Schwankungen einzelner Bäume ausgeglichen.

Hat ein Punkt x über alle iTrees im Mittel eine kurze Pfadlänge, so liegt er in einem Bereich der Daten, die nur sehr knapp besetzt sind. Die Wahrscheinlichkeit für einen Outlier ist entsprechend hoch.

20.2.2 Anomalie-Score und Interpretation

Das ist die originale Definition aus [41], Seite 10, Gleichung (2). Die meisten praktischen Implementierungen invertieren und verschieben den Original-Score für bessere Benutzerfreundlichkeit, behalten aber die gleiche Grundlogik bei.

Die Kernmetrik des Isolation Forest ist die Pfadlänge. Auf Basis der durchschnittlichen Pfadlänge über alle Bäume wird ein standardisierter Anomalie-Score berechnet.

Der **Anomalie-Score** $s(x, N)$ für einen Datenpunkt x aus einem Datensatz mit N Instanzen (bzw. der Größe der für die Bäume verwendeten

Stichprobe) wird wie folgt berechnet:

$$s(x, N) = 2^{-\frac{\bar{h}(x)}{c(N)}}$$

Dabei ist $\bar{h}(x)$ die durchschnittliche Pfadlänge von x über alle iTrees und $c(N)$ ist ein Normalisierungsfaktor (wie auf Seite 229 beschrieben), der der durchschnittlichen Pfadlänge einer erfolglosen Suche in einem binären Suchbaum entspricht.

Die Normalisierungskonstante $c(N)$ ist entscheidend, um die Pfadlängen über verschiedene Baumgrößen oder Datensätze hinweg vergleichbar zu machen. Sie stellt eine theoretische Obergrenze für die durchschnittliche Pfadlänge in den Bäumen dar.

Die Interpretation des Scores, der im Bereich $[0, 1]$ liegt, ist wie folgt:

- Wenn $\bar{h}$ sehr klein ist (der Punkt wird häufig schnell isoliert), nähert sich der Exponent der Null und $s(x, N)$ geht gegen $2^0 = 1$. Ein Score nahe 1 deutet stark auf eine **Anomalie** hin.
- Wenn $\bar{h}$ sich der durchschnittlichen Pfadlänge $c(N)$ nähert, geht der Exponent gegen -1 und $s(x, N)$ geht gegen $2^{-1} = 0.5$. Ein Score um 0.5 bedeutet, dass der Algorithmus keine klare Aussage treffen kann.
- Wenn $\bar{h}$ groß ist (deutlich größer als der Durchschnitt), wird der Exponent stark negativ und $s(x, N)$ nähert sich der Null an. Ein Score, der deutlich unter 0.5 liegt, deutet auf einen **normalen Datenpunkt** hin.

Beispiel: Berechnung des Anomalie-Scores

Angenommen, ein Isolation Forest wurde mit Stichproben der Größe $N = 256$ trainiert. Der Normalisierungsfaktor $c(256)$ wäre in diesem Fall 10.24.

- **Punkt A (Anomalie):** Dieser Punkt wird in den meisten Bäumen sehr schnell isoliert. Die durchschnittliche Pfadlänge sei $\bar{h}(A) = 4.0$. Sein Anomalie-Score wäre: $s(A, 256) = 2^{-\frac{4.0}{10.24}} \approx 2^{-0.39} \approx 0.763$. Dieser hohe Wert deutet auf eine Anomalie hin.
- **Punkt B (Normaler Punkt):** Dieser Punkt liegt in einer dichten Region und erfordert viele Splits. Die durchschnittliche Pfadlänge sei $\bar{h}(B) = 10.5$. Sein Anomalie-Score wäre: $s(B, 256) = 2^{-\frac{10.5}{10.24}} \approx 2^{-1.025} \approx 0.491$. Dieser Wert liegt nahe bei 0.5, d.h. es lässt sich keine Aussage machen, ob eine Anomalie vorliegt,

20.3 Eigenschaften und Anwendung

Der Isolation Forest zeichnet sich durch eine Reihe von vorteilhaften Eigenschaften aus, besitzt aber auch einige

Einschränkungen, die bei der Anwendung berücksichtigt werden müssen.

Die wichtigsten Hyperparameter des Modells sind die Anzahl der Bäume im Forest (normalerweise 100 bis 500) und die Größe der für jeden Baum verwendeten Stichprobe (N oder meist 256).

Eine höhere Anzahl von Bäumen führt zu stabileren Ergebnissen, erhöht aber die Rechenzeit.

Die Größe des Subsamples beeinflusst die Fähigkeit der Bäume, zwischen normalen und anomalen Punkten zu unterscheiden. Ein zu kleiner Wert kann dazu führen, dass auch normale Punkte zu schnell isoliert werden, während ein zu großer Wert die Effizienz des Algorithmus verringert.

Der **Fluch der Dimensionalität** beschreibt das Phänomen, dass in hochdimensionalen Räumen das Volumen so stark zunimmt, dass die verfügbaren Datenpunkte spärlich werden. Distanzmetriken verlieren ihre Aussagekraft, da die Distanz zwischen jedem Paar von Punkten annähernd gleich groß wird.

Die Messung, ob eine Anomalie vorliegt, erfolgt über die Länge der Pfade und nicht durch eine Distanzmessung. Damit ist er robuster bei hoch dimensionierten Daten.

Der Aufbau der Bäume erfolgt unabhängig und parallelisierbar. Da keine distanzbasierten Berechnungen notwendig sind, skaliert der Algorithmus linear mit der Anzahl der Datenpunkte und benötigt einen vergleichsweise geringen Speicherbedarf.

Da die Bäume unabhängig sind, müssen nicht alle gleichzeitig im Speicher gehalten werden. Sie können nacheinander aufgebaut und evaluiert werden.

Die Nachteile des Algorithmus liegen in seiner zufälligen Natur. Er kann Schwierigkeiten haben, wenn Anomalien selbst dichte, lokale Cluster bilden. In diesem Fall sind sie nicht mehr "wenige und anders" im Sinne des Algorithmus und werden möglicherweise nicht als anomal erkannt.

Ein weiteres potenzielles Problem ist, dass durch die zufällige Merkmalswahl irrelevante Merkmale für die Teilung verwendet werden können, was die Effektivität der Isolierung verringern kann. Dieser Effekt wird durch die Ensemble-Struktur zwar stark abgemildert, kann aber in Datensätzen mit sehr vielen irrelevanten Merkmalen eine Rolle spielen.

Dieses Problem ist auch als "Masking" bekannt: nahe beieinander liegende Anomalien "maskieren" sich gegenseitig, was es für den Algorithmus schwierig macht, einzelne Punkte dieser Gruppe zu isolieren, da sie lokal eine höhere Dichte aufweisen.

20.4 Praxisbeispiel: Cardiodaten

Abschließend soll der Algorithmus auf das schon bekannte Beispiel aus [9] angewandt werden:

Prompt für Isolation Forest Anomalieerkennung Cardiovascular Dataset

Das Verzeichnis ist *C:\Daten*. Die Datei *cardio_train_bereinigt.csv*

enthält in der ersten Zeile die Feldnamen. Die Trennung ist mit ";". Erstelle einen Python-Quellcode mit pyod und IForest, der folgendes macht:

1. Lade die Datei und wähle die Spalten `'height'`, `'weight'`, `'age'`, `'ap_lo'`, `'ap_hi'`.
2. Trainiere den Isolation Forest mit folgenden Parametern:
   a) Anzahl der iTrees: 500
   b) Sample Size: 256
   c) Random State: 42
3. Berechne die Anomalie-Scores für alle Datenpunkte mit `.dpredict_proba(X)[:, 1]`.
4. Erstelle eine umfassende statistische Analyse der Anomalie-Score-Verteilung:
   a) Berechne deskriptive Statistiken: Min, Max, Mittelwert, Median, Standardabweichung, MAD, IQR, Schiefe, Kurtosis
   b) Berechne relevante Perzentile: 1., 2., 5., 10., 25., 50., 75., 90., 95., 98., 99. Perzentil
   c) Bestimme die Anzahl der als anomal klassifizierten Datenpunkte
   d) Gebe alle Statistiken formatiert in einer übersichtlichen Tabelle aus und speichere diese unter `"cardio_IF_anomalie_statistiken.txt"`

   Visualisiere die Anomalie-Score-Verteilung in einem 1×2 Subplot-Layout:
   a) Linkes Subplot: Histogramm der Werte mit 150 Bins
   b) Rechtes Subplot: Boxplot der Werte (vertikal orientiert)
   c) Markiere das 99,9. Perzentil als rote Linie in den Plots
   d) Speichere die Grafik unter `"cardio_IF_anomalie_analyse.pdf"`
5. Identifiziere die 100 Datenpunkte mit den höchsten Anomalie-Scores (höchste Anomalität) und speichere die entsprechenden Datenpunkte (mit allen verfügbaren Spalten) geordnet nach absteigenden Anomalie-Scores in `"cardio_IF_anomalien.csv"`

Verwende für die Grafiken #D8E1F4, andere Blautöne und schwarz.

Hier wird der Anomalie-Score in einer Wahrscheinlichkeit ausgedrückt - d.h. bei 1 sicher eine Anomalie, bei 0 sicher normaler Datenpunkt. Es ist damit nicht der Score aus [41].

Prompt 20.1: Prompt für Isolation Forest Anomalieerkennung Cardiovascular Dataset

Das LLM (Grok Code Fast 1) hat das Script für folgende Abbildung generiert:

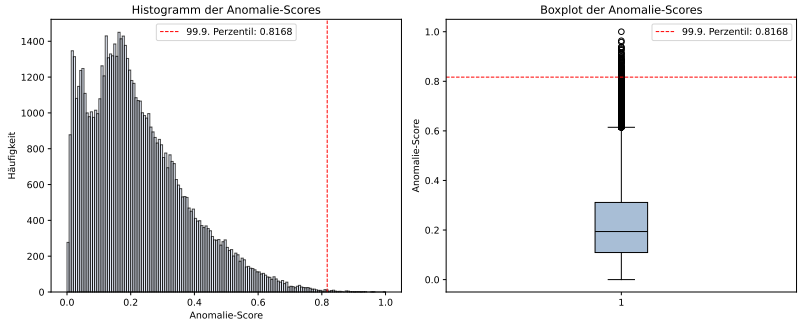


Abbildung 20.3: Histogramm der Anomalie-Scores aller Datenpunkte. Die Verteilung zeigt, dass die meisten Datenpunkte unauffällig sind. Die Verteilung ist typisch rechtsschief.

Analog zu den Outliern beim SOM werden auch hier die 10 größten Ausreißer angegeben:

Tabelle 20.1: Top 10 Datenpunkte mit den höchsten Anomalie-Scores

ID	Alter	Größe	Gewicht	RR sys.	RR dia.	Score
28605	19777	112	167.0	180	120	1.000
8600	16058	159	150.0	200	130	0.963
50043	23394	170	152.0	200	140	0.958
10899	15937	183	154.0	160	120	0.938
25757	15086	190	165.0	160	60	0.937
97768	15370	179	100.0	200	140	0.932
39296	14577	186	100.0	180	110	0.928
72770	22012	170	152.0	220	120	0.925
92892	17564	171	123.0	195	130	0.915
77447	16777	180	115.0	210	110	0.914

Unter den Top 10-Outlier von den Outliern des SOM (vgl. Tabelle 18.1).

20.5 Zusammenfassung

Der Isolation Forest ist ein unsupervised Algorithmus zur Anomalie-Erkennung.

Er stellt einen innovativen und leistungsstarken Ansatz zur Anomalieerkennung dar, der sich durch sein einzigartiges Isolationsprinzip von traditionellen, distanzbasierten Methoden abhebt.

Die zentrale Idee, dass Anomalien als „wenige und andere“ Datenpunkte leichter zu isolieren sind als normale Punkte, bildet die Grundlage für einen Algorithmus, der sich durch hohe Effizienz, Skalierbarkeit und besondere Eignung für hochdimensionale Daten auszeichnet.

Appendix

Statistische Grundlagen der Datenanalyse

A

Die statistische Beschreibung von Daten ist das Fundament jeder seriösen Datenanalyse. Bevor komplexere Verfahren zur Anomalieerkennung oder zum Datenqualitätsmanagement angewendet werden können, ist ein solides Verständnis der fundamentalen statistischen Kennzahlen und Visualisierungstechniken unerlässlich. Dieses Kapitel legt das mathematische und konzeptuelle Fundament, indem es die wichtigsten Maße der zentralen Tendenz, der Streuung sowie der Verteilungsform systematisch vorstellt.

Daten allein sind stumm – erst durch ihre statistische Charakterisierung und vor allem durch ihre Visualisierung werden Muster, Trends und Anomalien sichtbar. Das berühmte Datasaurus Dozen (vgl. Abbildung A.3) verdeutlicht eindringlich, warum reine Kennzahlen ohne visuelle Analyse unzureichend sind: Dreizehn völlig unterschiedliche Datensätze können nahezu identische statistische Eigenschaften aufweisen, aber völlig verschiedene Strukturen verbergen. Diese fundamentale Erkenntnis durchzieht die gesamte moderne Datenanalyse und unterstreicht die Notwendigkeit einer ausgewogenen Kombination aus quantitativer Beschreibung und visueller Exploration.

Die Entwicklung der deskriptiven Statistik geht auf die frühen Arbeiten von Astronomen und Vermessern im 17. und 18. Jahrhundert zurück, die versuchten, aus mehreren ungenauen Messungen den "wahren" Wert zu ermitteln.

A.1 Maße der zentralen Tendenz: Wo liegt das Zentrum der Daten?

Um die Verteilung von Daten zu verstehen, ist der erste Schritt die Bestimmung ihres "Zentrums" oder ihres "typischen" Wertes. Maße der zentralen Tendenz geben eine einzelne Zahl an, die versucht, die Mitte einer Verteilung zusammenzufassen. Die Wahl des richtigen Maßes hängt stark von der Verteilungsform der Daten und deren Anfälligkeit für Extremwerte ab.

A.1.1 Der Mittelwert (Durchschnitt)

Der arithmetische Mittelwert ist das wohl bekannteste Maß der zentralen Tendenz. Er wird berechnet, indem alle Werte eines Datensatzes summiert und anschließend durch die Anzahl der Werte dividiert werden. Man kann ihn sich

als den "Schwerpunkt" der Daten vorstellen, ähnlich dem physikalischen Schwerpunkt eines Objekts.

Neben dem arithmetischen Mittel gibt es auch das **geometrische Mittel** (für Wachstumsraten) und das **harmonische Mittel** (für Raten & Verhältnisse), die in spezifischen Kontexten passendere Zentralmaße sind.

Für einen Datensatz mit n Beobachtungen $x_1, x_2, \ldots, x_n$ ist der **arithmetische Mittelwert**, oft mit $\bar{x}$ (für eine Stichprobe) oder μ (für die Grundgesamtheit) bezeichnet, definiert als:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + \cdots + x_n}{n}$$

Die entscheidende Eigenschaft des Mittelwerts ist seine hohe Anfälligkeit gegenüber Ausreißern. Da jeder einzelne Wert direkt in die Berechnung einfließt, kann ein einziger extremer Wert den Mittelwert signifikant verzerren und ihn zu einem wenig repräsentativen Maß für das tatsächliche Zentrum der Daten machen.

In der Finanzwelt wird oft der getrimmte oder gestutzte Mittelwert verwendet. Hierbei wird ein bestimmter Prozentsatz der kleinsten und größten Werte entfernt, bevor der Mittelwert berechnet wird, um ihn robuster gegen Ausreißer zu machen.

Beispiel: Einfluss eines Ausreißers auf den Mittelwert

Gegeben sei eine Stichprobe der monatlichen Lieferzeiten in Tagen für ein Produkt: $\{5, 7, 6, 5, 8, 7\}$.

Der Mittelwert berechnet sich zu: $\bar{x} = \frac{5+7+6+5+8+7}{6} = \frac{38}{6} \approx 6.33$ Tage.

Nun wird fälschlicherweise eine Lieferzeit mit 50 Tagen erfasst. Der neue Datensatz lautet: $\{5, 7, 6, 5, 8, 7, 50\}$.

Der neue Mittelwert ist nun: $\bar{x}_{\text{neu}} = \frac{5+7+6+5+8+7+50}{7} = \frac{88}{7} \approx 12.57$ Tage.

Der Mittelwert hat sich mehr als verdoppelt und repräsentiert die "typische" Lieferzeit von rund 6 Tagen nicht mehr adäquat.

A.1.2 Der Median (Zentralwert)

Im Gegensatz zum Mittelwert ist der Median ein robustes Maß der zentralen Tendenz. Seine Robustheit rührt daher, dass seine Position nicht vom Wert der Extrempunkte abhängt, sondern nur von ihrer Anzahl. Der Median ist derjenige Wert, der eine nach Größe sortierte Datenreihe in zwei exakt gleich große Hälften teilt.

Der Median ist identisch mit dem 50. Perzentil oder dem 2. Quartil (Q2). Er teilt die Daten in eine untere und eine obere Hälfte.

Der **Median (Zentralwert)** ist der Wert, der an der mittleren Position einer sortierten Liste von Zahlen liegt. Bei einer ungeraden Anzahl n von Werten ist es der Wert an der Position $\frac{n+1}{2}$. Bei einer geraden Anzahl n ist es üblicherweise der arithmetische Mittelwert der beiden mittleren Werte an den Positionen $\frac{n}{2}$ und $\frac{n}{2} + 1$.

Aufgrund seiner Unempfindlichkeit gegenüber Ausreißern ist der Mediandas bevorzugte Maß der zentralen Tendenz,

wenn Daten eine schiefe Verteilung aufweisen oder das Vorhandensein von Extremwerten vermutet wird. In vielen ökonomischen Statistiken, wie z.B. bei der Angabe von Einkommen, wird der Median anstelle des Mittelwerts verwendet, um eine Verzerrung durch wenige extrem hohe Einkommen zu vermeiden.

In der digitalen Bildverarbeitung wird der **Medianfilter** verwendet, um "Salz-und-Pfeffer-Rauschen" zu entfernen. Jeder Pixel wird durch den Median der Pixel in seiner Umgebung ersetzt, was Ausreißer effektiv eliminiert.

A.1.3 Der Modus (Modalwert)

Der Modus ist das einfachste Maß der zentralen Tendenz. Er beschreibt den Wert, der in einem Datensatz am häufigsten auftritt. Seine größte Stärke liegt darin, dass er als einziges der hier vorgestellten Maße auch für kategoriale (nicht-numerische) Daten anwendbar ist.

Der **Modus (Modalwert)** ist der häufigste Wert in einer Datenreihe.

Bei kontinuierlichen, numerischen Daten ist der Modus oft wenig aussagekräftig, da selten exakt derselbe Wert mehrfach auftritt. Hier gruppiert man die Daten (z.B. im Histogramm) und spricht vom **Modalintervall**, dem Intervall mit der größten Häufigkeit.

Ein Datensatz kann einen Modus (**unimodal**), mehrere Modi (**multimodal**) oder gar keinen Modus haben. Letzteres ist der Fall, wenn alle Werte gleich häufig vorkommen (z.B. in $\{1, 2, 3, 4, 5\}$)

Die Analyse der Modalität ist besonders nützlich, um Cluster oder Subpopulationen in den Daten zu identifizieren. Eine bimodale Verteilung (zwei Modi) bei den Testergebnissen einer Prüfung könnte beispielsweise darauf hindeuten, dass es zwei Gruppen von Teilnehmern mit unterschiedlichem Vorwissen gab.

A.2 Streuungsmaße: Wie verteilt sind die Daten?

Maße der zentralen Tendenz allein sind unzureichend, um einen Datensatz zu beschreiben. Zwei Datensätze können denselben Mittelwert haben, aber völlig unterschiedlich verteilt sein. Streuungsmaße (auch Dispersionsmaße genannt) quantifizieren, wie stark die Datenpunkte um das Zentrum herum streuen oder wie weit sie voneinander entfernt sind.

A.2.1 Spannweite, Quartile und Interquartilsabstand (IQR)

Eine einfache, aber anfällige Methode zur Beschreibung der Streuung ist die Spannweite. Sie ist die Differenz zwischen dem größten und dem kleinsten Wert im Datensatz ($R = x_{\max} - x_{\min}$). Ähnlich wie der Mittelwert wird die Spannweite extrem von Ausreißern beeinflusst und gibt keine Auskunft über die Verteilung der Daten zwischen den Extrempunkten. Eine robustere Alternative basiert auf Quartilen. Quartile sind die Punkte, die eine sortierte Datenreihe in vier gleich große Abschnitte teilen.

Neben den Quartilen (4 Gruppen) gibt es weitere **Quantile**, z.B. **Dezile** (10 Gruppen) oder **Perzentile** (100 Gruppen), die eine feinere Unterteilung der Daten ermöglichen.

Das **erste Quartil (Q1)**, auch unteres Quartil genannt, ist der Wert, unter dem 25% der Daten liegen.

Das **zweite Quartil (Q2)** ist der Median, unter dem 50% der Daten liegen.

Das **dritte Quartil (Q3)**, auch oberes Quartil genannt, ist der Wert, unter dem 75% der Daten liegen.

Aus diesen Quartilen lässt sich ein äußerst nützliches und robustes Streuungsmaß ableiten: der Interquartilsabstand.

Der **Interquartilsabstand (IQR)** ist die Differenz zwischen dem dritten und dem ersten Quartil. Er umfasst die mittleren 50% der Daten.

$$\text{IQR} = Q3 - Q1$$

Da der IQR auf Quartilen basiert und die äußeren 25% der Daten an beiden Enden der Verteilung ignoriert, ist er unempfindlich gegenüber Ausreißern und bietet ein stabiles Maß für die Streuung des zentralen Datenkörpers.

A.2.2 Varianz und Standardabweichung

Varianz und Standardabweichung sind die gebräuchlichsten Streuungsmaße, wenn der Mittelwert als zentrales Maß verwendet wird. Sie beschreiben die durchschnittliche Abweichung der Datenpunkte vom Mittelwert.

Die Division durch $n-1$ anstelle von n bei der Stichprobenvarianz wird als **Bessel-Korrektur** bezeichnet. Sie korrigiert die Tatsache, dass die Varianz einer Stichprobe systematisch die Varianz der Grundgesamtheit unterschätzt und liefert einen erwartungstreuen Schätzer.

Die **(Stichproben-)Varianz** (s^2) ist die durchschnittliche quadratische Abweichung jedes Datenpunktes x_i vom Mittelwert $\bar{x}$.

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

Da die Varianz in quadrierten Einheiten der Originaldaten vorliegt (z.B. Euro^2), ist sie schwer zu interpretieren. Daher wird in der Praxis meist die Standardabweichung verwendet.

Die **(Stichproben-)Standardabweichung** (s) ist die positive Quadratwurzel der Varianz. Sie hat die gleiche Einheit wie die ursprünglichen Daten und ist daher leichter interpretierbar.

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

Der **Variationskoeffizient** ($s/\bar{x}$) ist ein relatives Streuungsmaß. Er setzt die Standardabweichung ins Verhältnis zum Mittelwert und erlaubt so den Vergleich der Streuung von Datensätzen mit unterschiedlichen Skalen oder Einheiten.

Beide Maße, Varianz und Standardabweichung, sind aufgrund ihrer Abhängigkeit vom Mittelwert und der Quadrierung der Abweichungen sehr anfällig für Ausreißer. Ein einzelner extremer Wert kann die Standardabweichung erheblich aufblähen.

Für normalverteilte Daten gilt die **68-95-99.7-Regel**: ca. 68% der Daten liegen innerhalb ±1 Standardabweichung vom Mittelwert, ca. 95% innerhalb ±2 und ca. 99.7% innerhalb ±3.

A.3 Maße der Verteilungsform: Schiefe und Wölbung

Während Maße der zentralen Tendenz und Streuung wichtige Aspekte einer Verteilung beschreiben, geben sie keine Auskunft über deren Form. Zwei Verteilungen können denselben Mittelwert und dieselbe Standardabweichung haben, aber völlig unterschiedliche Formen aufweisen. Die Charakterisierung der Verteilungsform erfolgt hauptsächlich durch zwei Kennzahlen: die Schiefe (Asymmetrie) und die Wölbung (Kurtosis).

Karl Pearson entwickelte Ende des 19. Jahrhunderts verschiedene Maße für die Schiefe. Das hier vorgestellte Momentbasierte Maß ist das gebräuchlichste.

A.3.1 Schiefe (Skewness)

Die Schiefe misst die Asymmetrie einer Verteilung. Sie quantifiziert, ob und in welche Richtung die Verteilung von einer symmetrischen Form abweicht.

Die **Schiefe (Skewness)** einer Verteilung ist das standardisierte dritte Moment um den Mittelwert:

$$\text{Schiefe} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^3}{s^3}$$

wobei s die Standardabweichung ist.

Als Faustregel gilt: Werte zwischen -0.5 und 0.5 deuten auf eine annähernd symmetrische Verteilung hin, Werte zwischen 0.5 und 1 (bzw. -1 und -0.5) auf moderate Schiefe, und Werte über 1 (bzw. unter -1) auf starke Schiefe.

Die Interpretation der Schiefe erfolgt anhand des Vorzeichens und der Größenordnung:

- **Schiefe = 0:** Perfekt symmetrische Verteilung (theoretisch)
- **Schiefe > 0:** Rechtsschiefe (positive Schiefe) – der "Schwanz" der Verteilung ist nach rechts verlängert, die meisten Werte liegen links vom Mittelwert
- **Schiefe < 0:** Linksschiefe (negative Schiefe) – der "Schwanz" der Verteilung ist nach links verlängert, die meisten Werte liegen rechts vom Mittelwert

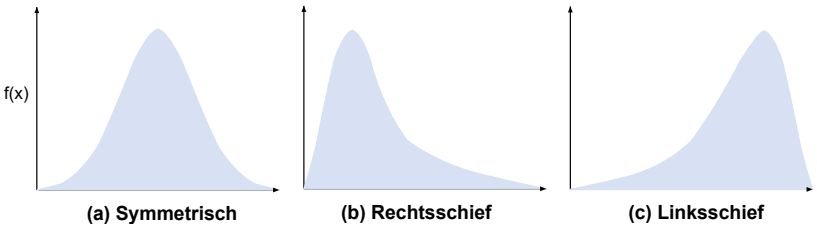


Abbildung A.1: Schematische Verteilungen mit unterschiedlicher Skewness/Schiefe. (a) Symmetrisch, (b) positive Schiefe und (c) negative Schiefe.

Beispiel: Interpretation der Schiefe bei Einkommensdaten

Einkommensdaten zeigen typischerweise eine positive Schiefe. Die meisten Menschen haben ein mittleres Einkommen, während wenige sehr hohe Einkommen erzielen. Diese "High Earner" erzeugen den langen rechten Schwanz der Verteilung. Ein Schiefe-Wert von +2.1 würde eine starke Rechtsschiefe anzeigen, was bedeutet, dass der Mittelwert deutlich größer als der Median ist und die Verteilung einen ausgeprägten rechten Schwanz aufweist.

A.3.2 Wölbung (Kurtosis)

Der Begriff "Kurtosis" stammt vom griechischen Wort "kyrtos" ab, was "gewölbt" oder "bauchig" bedeutet.

Die Wölbung beschreibt die "Schwere" der Ränder (Tails) einer Verteilung im Vergleich zu einer Normalverteilung. Sie quantifiziert, ob eine Verteilung mehr oder weniger Extremwerte in den Randbereichen aufweist als eine Normalverteilung mit gleicher Varianz.

Die **Wölbung (Kurtosis)** einer Verteilung ist das standardisierte vierte Moment um den Mittelwert:

$$\text{Kurtosis} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^4}{s^4}$$

Die hier gezeigten Formeln für Schiefe und Kurtosis verwenden $\frac{1}{n}$ und sind die vereinfachten biased Versionen für Einführungszwecke. In der statistischen Praxis werden oft unbiased Schätzer verwendet, z.B. mit Korrekturfaktoren $\frac{n}{(n-1)(n-2)}$ für Schiefe oder $\frac{n(n+1)}{(n-1)(n-2)(n-3)}$ für Kurtosis [42]. Software wie R oder Python's scipy.stats implementieren verschiedene Varianten.

Da die Normalverteilung eine Kurtosis von 3 hat, wird oft die **Exzess-Kurtosis** verwendet:

$$\text{Exzess-Kurtosis} = \text{Kurtosis} - 3$$

Die Interpretation erfolgt relativ zur Normalverteilung:

- **Exzess-Kurtosis = 0:** Mesokurtisch – ähnlich der Normalverteilung
- **Exzess-Kurtosis > 0:** Leptokurtisch – schwerere Ränder, spitzere Spitze, mehr Extremwerte
- **Exzess-Kurtosis < 0:** Platykurtisch – leichtere Ränder, flachere Spitze, weniger Extremwerte

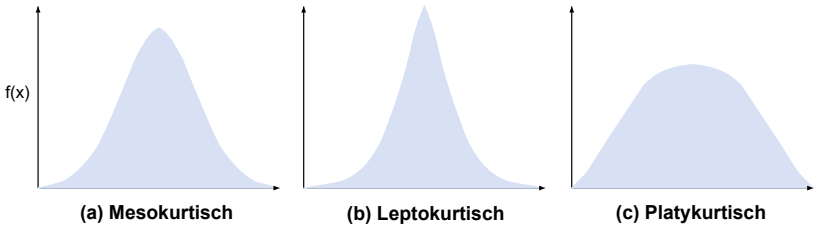


Abbildung A.2: Schematische Verteilungen mit unterschiedlicher Wölbung. (a) Wölbung wie Normalverteilung, (b) mehr Extremwerte/Fat Tails und (c) leichtere Ränder im Vergleich zur Normalverteilung.

Beispiel: Kurtosis in Finanzrenditen

Die täglichen Renditen von Aktien zeigen oft eine Exzess-Kurtosis von 3-6, was bedeutet, dass extreme positive oder negative Renditen häufiger auftreten als bei einer Normalverteilung erwartet. Dies ist der Grund, warum Finanzmodelle, die eine Normalverteilung annehmen, Marktcrashs und große Gewinne systematisch unterschätzen.

Hohe Kurtosis ist in der Finanzwelt besonders relevant, da sie auf ein erhöhtes Risiko extremer Verluste hinweist – ein Phänomen, das als "Fat-Tail-Risiko" bekannt ist.

A.4 Visuelle Datenexploration: Die Macht der Visualisierung

Die Datasaurus Dozen, eine moderne Erweiterung des klassischen Anscombe-Quartetts, demonstriert eindrucksvoll die fundamentale Bedeutung der Datenvisualisierung. Dreizehn völlig unterschiedliche Datensätze – von Dinosauriern über Sterne bis hin zu Kreisen – weisen nahezu identische statistische Kennzahlen auf: gleicher Mittelwert, gleiche Standardabweichung, gleiche Korrelation. Erst die visuelle Darstellung offenbart ihre wahre Natur und macht deutlich, dass numerische Zusammenfassungen allein niemals ausreichen.

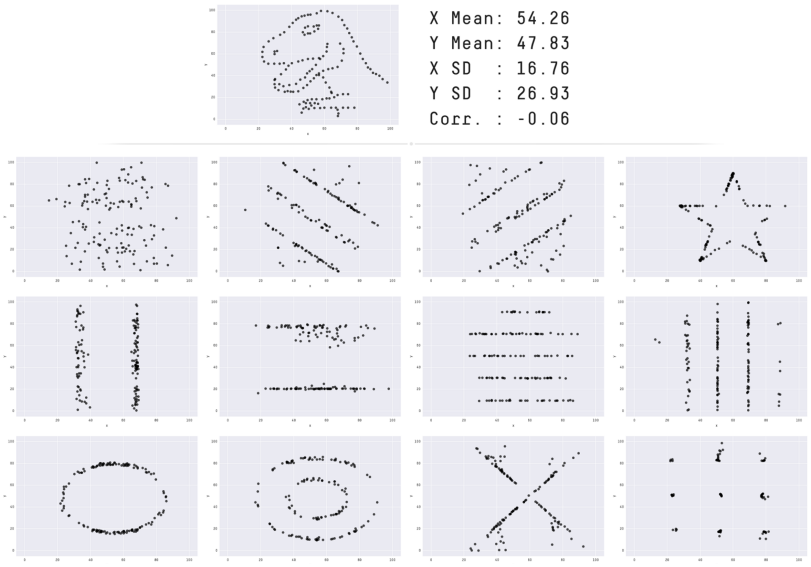


Abbildung A.3: The Datasaurus Dozen. Jeder Datensatz hat die gleichen statistischen Kenngrößen (Mittelwert, Standardabweichung, Pearson's Korrelation). Das Datasaurus Dozen wurde 2017 von Justin Matejka und George Fitzmaurice bei Autodesk Research entwickelt (Quelle: [43]).

John W. Tukey sagte: "The greatest value of a picture is when it forces us to notice what we never expected to see."

Diese Erkenntnis ist nicht nur akademisch interessant, sondern hat praktische Konsequenzen für jede Art von Datenanalyse. Anomalien, Cluster, Trends und Muster, die in den Rohdaten verborgen sind, werden oft erst durch geeignete Visualisierungen sichtbar. Die folgenden Abschnitte stellen die wichtigsten grafischen Werkzeuge zur Exploration univariater Daten vor.

A.4.1 Das Histogramm

Die Wahl der Intervallbreite (**bin width**) ist entscheidend! Zu breite Intervalle können wichtige Muster verschleiern, zu schmale lassen die Verteilung verrauscht und unklar erscheinen.

Ein Histogramm ist eine grafische Darstellung der Häufigkeitsverteilung von numerischen Daten. Es gruppiert die Werte in Intervalle (sogenannte "bins") und zeigt die Anzahl der Datenpunkte in jedem Intervall als Balken an.

Histogramme ermöglichen es, auf einen Blick zu erkennen:

- Die **Form der Verteilung** (symmetrisch, schief, multimodal)
- Die **zentrale Tendenz** (wo sich die meisten Werte konzentrieren)
- Die **Streuung** (wie breit die Verteilung ist)
- **Ausreißer** (isolierte Balken fernab der Hauptverteilung)
- **Datenlücken** (Bereiche ohne Beobachtungen)

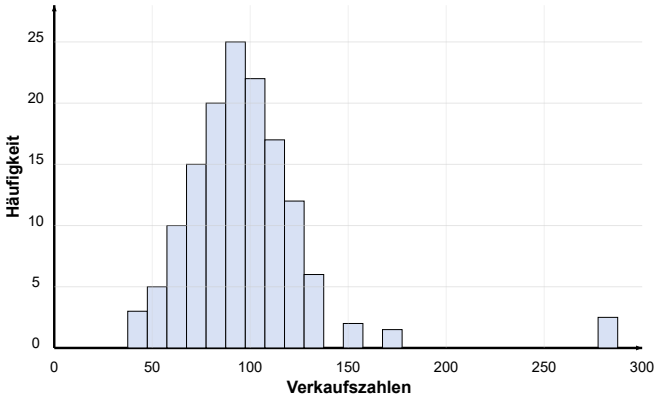


Abbildung A.4: Histogramm von Verkaufszahlen. Der isolierte Balken auf der rechten Seite deutet auf einen potenziellen Ausreißer oder ein seltenes, extremes Ereignis hin. Die Hauptverteilung zeigt eine leichte Rechtsschiefe.

A.4.2 Der Boxplot (Kastengrafik)

Der Boxplot, entwickelt vom Statistiker John W. Tukey, ist das primäre Werkzeug zur visuellen Darstellung der wichtigsten statistischen Kennzahlen und zur Identifikation von Ausreißern. Er visualisiert die Fünf-Punkte-Zusammenfassung eines Datensatzes auf eine sehr kompakte und aussagekräftige Weise.

Ein Boxplot besteht aus folgenden Elementen:

- **Die Box:** Repräsentiert den Interquartilsabstand (IQR) und damit die mittleren 50% der Daten
- **Die Medianslinie:** Eine Linie innerhalb der Box, die den Median (Q2) markiert
- **Die Whiskers (Antennen):** Linien, die sich von der Box zu den extremsten Datenpunkten erstrecken, die noch als "normal" gelten
- **Ausreißerpunkte:** Einzelne Punkte außerhalb der Whiskers, die als potenzielle Anomalien markiert sind

John W. Tukey (1915–2000) war ein amerikanischer Mathematiker und Statistiker, dessen Arbeit die Praxis der Datenanalyse revolutionierte. Er schlug den Begriff "Bit"(als Kurzform für "binary digit") vor, der dann von Claude Shannon in dessen grundlegendem Werk zur Informationstheorie verwendet wurde ([44]). Tukey entwickelte auch den Boxplot als Teil seiner Philosophie der explorativen Datenanalyse ([11]).

Die Whiskers erstrecken sich typischerweise bis zum weitesten Datenpunkt, der noch innerhalb von $1.5 \times \text{IQR}$ von den Quartilsgrenzen liegt. Alle Punkte außerhalb dieser Grenzen werden explizit als potenzielle Ausreißer dargestellt.

Die $1.5 \times \text{IQR}$-Regel ist eine Heuristik, keine mathematische Gewissheit. Für normalverteilte Daten markiert sie ca. 0.7% der Punkte als Ausreißer. Wichtig ist, diese Punkte zu untersuchen, nicht blind zu löschen.

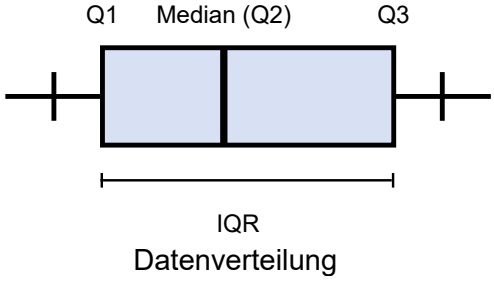


Abbildung A.5: Ein Boxplot, der die Verteilung von Testergebnissen zeigt. Der Median liegt links von der Mitte der Box, was auf eine leichte Rechtsschiefe hindeutet.

Ein ähnlicher Anwendungsfall sind Candle-Sticks für Zeitreihen wie Börsenkurse, wo jeder Tag als ein einzelner Candle-Stick dargestellt wird. Dabei repräsentieren die Stick-Grenzen die Eröffnungs- und Schlusskurse, während die "Whiskers"die Tages-Höchst- und -Tiefstwerte anzeigen – eine Darstellung, die auch als **Candlestick-Chart** bekannt ist. Allerdings werden anders als in Boxplots keine statistischen Größen, sondern tatsächliche Werte dargestellt.

Der große Vorteil von Boxplots liegt in ihrer Kompaktheit und ihrer Fähigkeit, mehrere Datensätze oder Gruppen nebeneinander zu vergleichen. Sie eignen sich hervorragend für die schnelle Identifikation von Ausreißern, den Vergleich der zentralen Tendenz zwischen Gruppen und die Bewertung der Streuung und Symmetrie von Verteilungen. Darüber hinaus ermöglichen sie die Erkennung von Unterschieden zwischen Kategorien oder Zeitperioden auf einen Blick.

A.5 Praktische Umsetzung in der Datenanalyse

Die Kenntnis statistischer Maße und Visualisierungen mündet in einen systematischen Workflow. Die folgenden Elemente bieten einen praxisnahen Leitfaden für die ersten Schritte jeder Datenanalyse.

To Do: Explorative Erstanalyse

Für jede relevante Variable (Spalte) in einem neuen Datensatz sollten die folgenden Schritte routinemäßig durchgeführt werden:

1. **Kennzahlen berechnen:** Ermitteln Sie die zentralen statistischen Maße: Mittelwert, Median, Standardabweichung, IQR, Minimum, Maximum, Anzahl der Werte, Anzahl der fehlenden Werte.
2. **Verteilungsform bestimmen:** Berechnen Sie Schiefe und Kurtosis, um eine erste quantitative Einschätzung der Verteilungsform zu erhalten.
3. **Visuelle Analyse (Histogramm):** Erstellen Sie ein Histogramm, um die Verteilung visuell zu erfassen. Achten Sie auf die Form (Symmetrie, Modalität), die Streuung und potenzielle Datenlücken.
4. **Ausreißer-Analyse (Boxplot):** Erstellen Sie einen Boxplot. Identifizieren Sie die nach der $1.5 \times$ IQR-Regel markierten Ausreißer.
5. **Synthese:** Vergleichen Sie die Kennzahlen mit den Visualisierungen. Führen Ausreißer zu einer starken Diskrepanz zwischen Mittelwert und Median? Bestätigt das Histogramm die berechnete Schiefe? Dokumentieren Sie die Befunde.

Insbesondere ein Histogramm ist wichtig. Nur die visuelle Betrachtung hilft, die Daten zu verstehen.

Für eine abstrakte Datei ‘C:\Daten\daten.csv‘ mit Feldern 1 bis 20 ist folgender Prompt eine Ausgangsbasis:

Prompt: Deskriptive Statistik generieren

Erstelle ein Skript in {Sprache}, das die folgende Aufgabe ausführt:

1. Lade den Datensatz aus ‘C:\Daten\daten.csv‘.

2. Für jede numerische Spalte von ‘Feld1‘ bis ‘Feld20‘:
    a) Berechne die folgenden deskriptiven Statistiken: Anzahl, Anzahl fehlender Werte, Mittelwert, Median, Standardabweichung, Interquartilsabstand (IQR), Minimum, 25. Perzentil (Q1), 50. Perzentil (Median/Q2), 75. Perzentil (Q3), Maximum, Schiefe und Exzess-Kurtosis.
    b) Gib die Ergebnisse in einer übersichtlichen Tabelle aus und speichere diese in ’tabelle_{Feld}.xlsx’}.
3. Für jede der Spalten
    a) Erstelle ein Histogramm, um die Datenverteilung zu visualisieren.
    b) Erstelle einen Boxplot, um die Streuung und potenzielle Ausreißer darzustellen.
    c) Speichere jede Grafik als separate Bilddatei (‘histogramm_{Feld}.pdf‘, ‘boxplot_{Feld}.pdf‘).
4. Gib eine kurze textliche Zusammenfassung aus, die für jede Spalte auf eine hohe Schiefe (> 1 oder < -1) oder eine hohe Kurtosis (> 2) hinweist.

Prompt A.1: Prompt für explorative Erstanalyse

A.6 Zusammenfassung

Dieses Kapitel hat das statistische Fundament für alle weiteren Analysen gelegt. Die vorgestellten Maße der zentralen Tendenz (Mittelwert, Median, Modus), der Streuung (Varianz, Standardabweichung, IQR) und der Verteilungsform (Schiefe, Wölbung) bilden das Grundgerüst zur Beschreibung von Datenverteilungen. Eine der wichtigsten Erkenntnisse ist die Unterscheidung zwischen anfälligen und robusten Maßen. Während Mittelwert und Standardabweichung durch Extremwerte stark beeinflusst werden können, bleiben Median und IQR stabil und sind oft die bessere Wahl für reale, “unsaubere“ Datensätze.

Die wichtigsten Standardwerke für die Einführung in die Statistik mit guter Darstellung von Mittelwert, Median, IQR und Histogrammen sind [42] als deutschsprachiges Lehrbuch, [45] für didaktisch hervorragende Grundlagenvermittlung und [46] für anwendungsorientierte Wirtschaftsstatistik. Ergänzend dazu bieten [47] als Nachschlagewerk und [48] als kommentierte Formelsammlung wertvolle Unterstützung.

Die zentrale Lektion des Datasaurus Dozen ist, dass statistische Kennzahlen niemals ohne visuelle Exploration interpretiert werden sollten. Histogramme und Boxplots sind unverzichtbare Werkzeuge, um die wahre Struktur der Daten zu verstehen, Annahmen zu überprüfen und potenzielle Probleme wie Ausreißer oder multimodale Verteilungen zu identifizieren. Ein fundiertes Verständnis dieser Grundlagen ist damit die entscheidende Voraussetzung für die Auswahl, Anwendung und Interpretation der komplexeren Methoden zur Datenqualitätssicherung und Anomalieerkennung, die in den entsprechenden Kapiteln behandelt werden.

Maximum-Likelihood-Schätzung und Parameterschätzmethoden

B

Die Schätzung unbekannter Parameter aus beobachteten Daten bildet das Herzstück der statistischen Inferenz. In diesem Kapitel betrachten wir zwei fundamentale Ansätze: die Maximum-Likelihood-Schätzung (MLE) und. die Momentenmethode. Diese Methoden sind nicht nur von theoretischem Interesse, sondern bilden auch die Grundlage für viele moderne statistische Verfahren, einschließlich des Akaike-Informationskriteriums (AIC) (vgl. Kapitel 11).

Ein sehr gutes Buch zum Thema ist *Methoden der statistischen Inferenz* von Leonhard Held ([49])

Die Bedeutung dieser Schätzmethoden erstreckt sich weit über die reine Parameterschätzung hinaus. Sie liefern die mathematischen Werkzeuge für Modellvergleich, Hypothesentests und Unsicherheitsquantifizierung. Ein tiefes Verständnis ihrer Prinzipien, Eigenschaften und Anwendungsbereiche ist daher unerlässlich für jeden, der sich mit statistischer Datenanalyse beschäftigt.

B.1 Das Prinzip der größten Plausibilität

Der Begriff „Likelihood“ wurde von R. A. Fisher populär gemacht. Er betonte den Unterschied zur Wahrscheinlichkeit: Wahrscheinlichkeit bezieht sich auf zukünftige Ereignisse bei bekannten Parametern, Likelihood auf bekannte Daten bei unbekannten Parametern.

Das Kernprinzip der Maximum-Likelihood-Schätzung ist intuitiv:

Gegeben ein Satz von beobachteten Daten und ein parametrisches Modell, wählen wir diejenigen Parameterwerte, unter denen die Beobachtung unserer Daten am plausibelsten oder „wahrscheinlichsten“ erscheint.

Die **Likelihood-Funktion**, bezeichnet mit $L(\theta|x)$, ist die gemeinsame Wahrscheinlichkeitsdichte- oder Wahrscheinlichkeitsmassenfunktion der beobachteten Daten $x = (x_1, x_2, \ldots, x_n)$, betrachtet als Funktion des Parametervektors θ. Wenn die Beobachtungen als unabhängig und identisch verteilt (i.i.d.) angenommen werden, vereinfacht sich die gemeinsame Dichte zum Produkt der einzelnen Dichten:

$$L(\theta|x) = f(x_1, x_2, \ldots, x_n|\theta) = \prod_{i=1}^{n} f(x_i|\theta)$$

Obwohl MLE heute allgegenwärtig ist, war sie bei ihrer Einführung durch Fisher in den 1920er Jahren umstritten und konkurrierte lange mit der Momentenmethode.

Es ist wichtig zu betonen, dass $L(\theta|x)$ keine Wahrscheinlichkeitsdichte für θ ist. Sie ist eine Funktion, die misst, wie gut verschiedene Werte von θ die beobachteten Daten erklären.

Aus mathematischen und numerischen Gründen ist es oft praktischer, nicht die Likelihood-Funktion selbst, sondern deren natürlichen Logarithmus, die Log-Likelihood-Funktion, zu maximieren.

Es werden Produkte zu Summen. Diese sind nicht nur numerisch stabiler, sondern auch die Ableitungen sind leichter zu berechnen. Das erleichtert auch das Finden extremer Punkte. Nachdem die ln-Funktion monoton steigend ist, sind Extremwerte der Log-Likelihood- auch Extremwerte der Likelihood-Funktion.

Die **Log-Likelihood-Funktion** $\ell(\theta|x)$ ist der natürliche Logarithmus der Likelihood-Funktion:

$$\ell(\theta|x) = \ln(L(\theta|x)) = \ln\left(\prod_{i=1}^{n} f(x_i|\theta)\right) = \sum_{i=1}^{n} \ln(f(x_i|\theta))$$

Da der Logarithmus eine streng monotone Funktion ist, führt die Maximierung von $\ell(\theta|x)$ zum selben Ergebnis wie die Maximierung von $L(\theta|x)$. Die Verwendung des Logarithmus wandelt das Produkt in eine Summe um, was die Differentiation erheblich vereinfacht und numerische Probleme wie Unterlauf bei sehr kleinen Wahrscheinlichkeiten vermeidet.

arg max ist genau der Argumentwert θ, der $L(\theta|x)$ maximiert.

Der **Maximum-Likelihood-Schätzer**, bezeichnet mit $\hat{\theta}_{\text{ML}}$, ist derjenige Parameterwert, der die Likelihood-Funktion (und somit die Log-Likelihood-Funktion) maximiert. Er wird formal definiert als:

$$\hat{\theta}_{\text{ML}} = \arg\max_{\theta} L(\theta|x) = \arg\max_{\theta} \ell(\theta|x)$$

In der Praxis wird $\hat{\theta}_{\text{ML}}$ typischerweise durch analytisches oder numerisches Lösen der Gleichung $\frac{\partial \ell(\theta|x)}{\partial \theta} = 0$ und anschließendem Prüfen der 2. Ableitung gefunden.

Beispiel: MLE für eine Exponentialverteilung

Angenommen, wir haben eine Stichprobe $x = (x_1, \ldots, x_n)$ von Lebensdauern, die wir mit einer Exponentialverteilung mit dem Ratenparameter λ modellieren. Die Dichtefunktion ist $f(x_i|\lambda) = \lambda e^{-\lambda x_i}$ für $x_i \geq 0$. Die Log-Likelihood-Funktion ist:

$$\begin{aligned}\ell(\lambda|x) &= \sum_{i=1}^{n} \ln(\lambda e^{-\lambda x_i}) \\ &= \sum_{i=1}^{n} (\ln(\lambda) - \lambda x_i) \\ &= n \ln(\lambda) - \lambda \sum_{i=1}^{n} x_i\end{aligned}$$

Um den Maximierer zu finden, leiten wir nach λ ab und setzen die Ableitung gleich Null:

$$\frac{\partial \ell}{\partial \lambda} = \frac{n}{\lambda} - \sum_{i=1}^{n} x_i \stackrel{!}{=} 0$$

Das Auflösen nach λ ergibt den Maximum-Likelihood-Schätzer:

$$\hat{\lambda}_{\text{ML}} = \frac{n}{\sum_{i=1}^{n} x_i} = \frac{1}{\bar{x}}$$

Der Schätzer für die Rate λ ist also der Kehrwert des Stichprobenmittelwerts $\bar{x}$.

B.2 Numerische Verfahren zur Lösung von MLE-Problemen

In vielen praktischen Anwendungen lässt sich das Maximum der Log-Likelihood-Funktion nicht analytisch bestimmen. In solchen Fällen kommen numerische Optimierungsverfahren zum Einsatz.

Newton-Raphson-Verfahren: Das am häufigsten verwendete Verfahren ist die Newton-Raphson-Methode, die die zweite Ableitung der Log-Likelihood-Funktion nutzt:

$$\theta^{(k+1)} = \theta^{(k)} - \left[\frac{\partial^2 \ell}{\partial \theta \partial \theta^T}\right]^{-1} \frac{\partial \ell}{\partial \theta}$$

Das Newton-Raphson-Verfahren konvergiert quadratisch, wenn es in der Nähe der Lösung startet, kann aber bei schlechten Startwerten divergieren oder sehr langsam konvergieren.

B.3 Eigenschaften der Maximum-Likelihood-Schätzung

Die Popularität der MLE beruht auf ihren wünschenswerten asymptotischen Eigenschaften, das heißt, auf ihrem Verhalten bei großen Stichprobenumfängen ($n \to \infty$). Unter bestimmten Regularitätsbedingungen, die in den meisten praktischen Anwendungen erfüllt sind, weist der ML-Schätzer folgende Eigenschaften auf:

Die MLE ist **konsistent**. Das bedeutet, dass der Schätzer $\hat{\theta}_{\text{ML}}$ mit wachsender Stichprobengröße gegen den wahren, aber unbekannten Parameterwert θ_0 konvergiert:

$$\hat{\theta}_{\text{ML}} \xrightarrow{P} \theta_0 \quad \text{für } n \to \infty$$

Die Regularitätsbedingungen für die MLE umfassen u.a. die Annahme, dass der Parameterraum kompakt ist und dass die wahre Datenverteilung zum Modell gehört. Verstöße können die Gültigkeit der Eigenschaften beeinträchtigen.

Dies stellt sicher, dass wir mit genügend Daten dem wahren Wert beliebig nahekommen können.

Die MLE ist **asymptotisch normalverteilt**. Für große n kann die

Bei einer logistischen Regression werden hieraus auch die Z-Werte für die einzelnen Prädikatoren berechnet.

Verteilung von $\hat{\theta}_{\text{ML}}$ durch eine Normalverteilung approximiert werden:

$$\hat{\theta}_{\text{ML}} \approx \mathcal{N}(\theta_0, I(\theta_0)^{-1})$$

Hierbei ist $I(\theta_0)$ die **Fisher-Informationsmatrix**, die die Krümmung der Log-Likelihood-Funktion am wahren Parameterwert misst und angibt, wie viel Information die Daten über den Parameter enthalten.

Die Fisher-Informationsmatrix ist definiert als:

$$\begin{aligned} I(\theta) &= -\mathbb{E}\left[\frac{\partial^2 \ell(\theta)}{\partial\theta\partial\theta^T}\right] \\ &= \begin{pmatrix} -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_1^2}\right] & -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_1\partial\theta_2}\right] & \cdots & -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_1\partial\theta_p}\right] \\ -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_2\partial\theta_1}\right] & -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_2^2}\right] & \cdots & -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_2\partial\theta_p}\right] \\ \vdots & \vdots & \ddots & \vdots \\ -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_p\partial\theta_1}\right] & -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_p\partial\theta_2}\right] & \cdots & -\mathbb{E}\left[\frac{\partial^2\ell}{\partial\theta_p^2}\right] \end{pmatrix} \end{aligned}$$

wobei $\ell(\theta) = \ln L(\theta)$ die Log-Likelihood-Funktion und $\theta = (\theta_1, \ldots, \theta_p)^T$ der p-dimensionale Parametervektor ist.

Diese Eigenschaft ist die Grundlage für die Konstruktion von Konfidenzintervallen und die Durchführung von Hypothesentests.

Beispiel: Asymptotische Normalität der MLE für eine Normalverteilung

Angenommen, wir haben eine Stichprobe $x = (x_1, \ldots, x_n)$ aus einer Normalverteilung $\mathcal{N}(\mu, \sigma^2)$ mit unbekannten Parametern $\theta = (\mu, \sigma^2)$.

Maximum-Likelihood-Schätzer: Die MLE sind gegeben durch:

$$\hat{\mu}_{\text{ML}} = \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\hat{\sigma}^2_{\text{ML}} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Die Fisher-Information quantifiziert, wie "scharf" unsere statistische Lupe ist! Je höher n desto größer die entsprechende Fisher-Information.

Fisher-Informationsmatrix: Die Fisher-Informationsmatrix für n Beobachtungen ist:

$$I(\theta) = n \cdot \begin{pmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{1}{2\sigma^4} \end{pmatrix}$$

Die inverse Fisher-Informationsmatrix ist:

$$I(\theta)^{-1} = \begin{pmatrix} \frac{\sigma^2}{n} & 0 \\ 0 & \frac{2\sigma^4}{n} \end{pmatrix}$$

Asymptotische Verteilung: Für große n gilt:

$$\begin{pmatrix} \hat{\mu}_{\text{ML}} \\ \hat{\sigma}^2_{\text{ML}} \end{pmatrix} \approx \mathcal{N}\left(\begin{pmatrix} \mu \\ \sigma^2 \end{pmatrix}, \begin{pmatrix} \frac{\sigma^2}{n} & 0 \\ 0 & \frac{2\sigma^4}{n} \end{pmatrix} \right)$$

Das bedeutet:

$$\hat{\mu}_{\text{ML}} \approx \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right) \quad \text{mit Standardfehler } \frac{\sigma}{\sqrt{n}}$$

$$\hat{\sigma}^2_{\text{ML}} \approx \mathcal{N}\left(\sigma^2, \frac{2\sigma^4}{n}\right) \quad \text{mit Standardfehler } \sigma^2\sqrt{\frac{2}{n}}$$

Numerisches Beispiel: Für $\mu = 10$, $\sigma^2 = 4$ und $n = 100$ ergibt sich asymptotisch:

$$\hat{\mu}_{\text{ML}} \approx \mathcal{N}(10, 0{,}04)$$
$$\hat{\sigma}^2_{\text{ML}} \approx \mathcal{N}(4, 0{,}32)$$

Die Schätzer sind asymptotisch unabhängig und ihre Unsicherheit nimmt mit $1/\sqrt{n}$ ab.

Die MLE ist **asymptotisch effizient**. Das bedeutet, dass sie unter allen asymptotisch unverzerrten Schätzern die kleinstmögliche Varianz erreicht.

Die Cramér-Rao-Ungleichung, benannt nach Harald Cramér und C. R. Rao, gibt eine untere Grenze für die Varianz eines unverzerrten Schätzers an. Ein Schätzer, der diese Grenze erreicht, ist „maximal effizient".

B.4 Momentenmethode als Alternative zur MLE

Die Momentenmethode ist eine der ältesten und intuitivsten Parameterschätzmethoden in der Statistik. Sie wurde bereits im 19. Jahrhundert von Karl Pearson entwickelt und bildet eine wichtige Alternative zur Maximum-Likelihood-Schätzung, insbesondere wenn die MLE analytisch schwer zu berechnen ist.

B.4.1 Das Prinzip der Momentengleichsetzung

Das Grundprinzip der Momentenmethode ist elegant und direkt:

Wir setzen die theoretischen Momente einer Verteilung, ausgedrückt als Funktionen der unbekannten Parameter, gleich den entsprechenden empirischen Momenten aus den beobachteten Daten und lösen das entstehende Gleichungssystem nach den Parametern auf.

Karl Pearson führte die Momentenmethode 1894 ein, lange bevor Fisher die Maximum-Likelihood-Methode entwickelte. Sie war die dominante Schätzmethode bis in die 1920er Jahre.

Alternativ können auch zentralisierte Momente verwendet werden, die um den Erwartungswert zentriert sind:

$$\mu_k^c(\theta) = \mathbb{E}[(X - \mathbb{E}[X])^k]$$

Das **k-te theoretische Moment** einer Zufallsvariable X mit Parameter θ ist definiert als:

$$\mu_k(\theta) = \mathbb{E}[X^k] = \int_{-\infty}^{\infty} x^k f(x|\theta)\, dx$$

für kontinuierliche Verteilungen bzw. als entsprechende Summe für diskrete Verteilungen.

Das **k-te empirische Moment** einer Stichprobe $x = (x_1, x_2, \ldots, x_n)$ ist definiert als:

$$\hat{\mu}_k = \frac{1}{n} \sum_{i=1}^{n} x_i^k$$

Die ersten vier Momente haben spezielle Bedeutungen: Das erste Moment ist der Erwartungswert, das zweite zentralisierte Moment die Varianz, das dritte charakterisiert die Schiefe und das vierte die Wölbung (Kurtosis).

Der **Momentenschätzer** für einen p-dimensionalen Parametervektor $\theta = (\theta_1, \ldots, \theta_p)^T$ wird durch Lösung des Gleichungssystems

$$\mu_k(\theta) = \hat{\mu}_k \quad \text{für } k = 1, 2, \ldots, p$$

bestimmt. Der resultierende Schätzer wird mit $\hat{\theta}_{\text{MM}}$ bezeichnet.

Bei der Momentenmethode werden die einzelnen empirischen Moment als beste Schätzer für die "wahren" Momente genommen. Hat eine Verteilung nur endlich viele Momente, die sich aus der Stichprobe ermitteln lassen, kann so die Verteilung bestimmt werden.

Beispiel: Momentenmethode für eine Normalverteilung

Angenommen, wir haben eine Stichprobe $x = (x_1, \ldots, x_n)$ aus einer Normalverteilung $\mathcal{N}(\mu, \sigma^2)$ mit unbekannten Parametern $\theta = (\mu, \sigma^2)$.

Theoretische Momente:

$$\mu_1(\mu, \sigma^2) = \mathbb{E}[X] = \mu$$
$$\mu_2(\mu, \sigma^2) = \mathbb{E}[X^2] = \text{Var}(X) + (\mathbb{E}[X])^2 = \sigma^2 + \mu^2$$

Empirische Momente:

$$\hat{\mu}_1 = \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$
$$\hat{\mu}_2 = \frac{1}{n} \sum_{i=1}^{n} x_i^2$$

Momentengleichungen:

$$\mu = \bar{x}$$
$$\sigma^2 + \mu^2 = \frac{1}{n} \sum_{i=1}^{n} x_i^2$$

Momentenschätzer: Das Auflösen ergibt:

$$\hat{\mu}_{\text{MM}} = \bar{x}$$
$$\hat{\sigma}^2_{\text{MM}} = \frac{1}{n} \sum_{i=1}^{n} x_i^2 - \bar{x}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

B.4.2 Eigenschaften der Momentenmethode

Die Momentenmethode besitzt ebenfalls wünschenswerte asymptotische Eigenschaften, auch wenn sie in der Regel nicht so effizient ist wie die Maximum-Likelihood-Schätzung.

Momentenschätzer sind **konsistent**, das heißt:

$$\hat{\theta}_{\text{MM}} \xrightarrow{P} \theta_0 \quad \text{für } n \to \infty$$

vorausgesetzt, die theoretischen Momente existieren und das Gleichungssystem ist eindeutig lösbar.

Die Konsistenz der Momentenschätzer folgt aus dem starken Gesetz der großen Zahlen, welches besagt, dass empirische Momente gegen die theoretischen Momente konvergieren.

Momentenschätzer sind **asymptotisch normalverteilt**:

$$\sqrt{n}(\hat{\theta}_{\text{MM}} - \theta_0) \xrightarrow{d} \mathcal{N}(0, V)$$

wobei die Kovarianzmatrix V über die Delta-Methode aus den Momenten berechnet werden kann.

Wichtig: Grenzen der Robustheit

Obwohl die Momentenmethode in manchen Situationen robuster ist als die MLE, sollte beachtet werden:

- Höhere Momente (3., 4., etc.) sind sehr ausreißerempfindlich
- Die Robustheit nimmt mit der Anzahl verwendeter Momente ab
- Bei stark schiefen Verteilungen können Momentenschätzer instabil werden

Die relative Effizienz der Momentenmethode zur MLE hängt von der spezifischen Verteilung ab. Bei der Normalverteilung sind beide gleich effizient, bei anderen Verteilungen ist die MLE meist überlegen.

B.5 Vergleich der Schätzmethoden

Ein systematischer Vergleich der beiden Hauptschätzmethoden zeigt sowohl Gemeinsamkeiten als auch wichtige Unterschiede:

Analytische Lösbarkeit: Die Momentenmethode führt oft zu expliziten, geschlossenen Formeln für die Schätzer. Die MLE erfordert hingegen häufig numerische Optimierungsverfahren, insbesondere bei komplexeren Modellen.

Numerische Stabilität: Bei der Momentenmethode können numerische Probleme durch schlecht konditionierte Momentengleichungen auftreten. Die MLE kann durch lokale Maxima oder flache Likelihood-Flächen Schwierigkeiten bereiten.

Die Wahl der Schätzmethode sollte auch praktische Aspekte berücksichtigen: verfügbare Software, Rechenzeit und die Erfahrung des Anwenders.

Tabelle B.1: Statistische Eigenschaften von MLE und Momentenmethode

Eigenschaft	MLE	Momentenmethode
Konsistenz	✓	✓
Asymptotische Normalität	✓	✓
Asymptotische Effizienz	✓	Meist nicht
Verzerrung bei kleinen Stichproben	Möglich	Möglich
Robustheit gegenüber Ausreißern	Gering	Moderat
Rechenkomplexität	Hoch	Niedrig
Anwendbarkeit	Universal	Eingeschränkt

Wird nach einer passenden "einfachen" Verteilung reicht die Momentummethode bei großen Stichproben und ist auch leichter umzusetzen. Die MLE spielt ihr Stärken bei unbeknnaten Verteilungen aus, für die im Vorfeld keine Momente bekannt sind, z.B. Machinelles Lernen oder logstische Regression

Für viele Standardverteilungen (Normalverteilung, Exponentialverteilung, Poisson-Verteilung) liefern beide Methoden **identische Schätzer**. Bei komplexeren Verteilungen unterscheiden sich die Schätzer, wobei die Unterschiede bei großen Stichproben meist gering sind.

Beispiel: Unterschiede bei der Weibull-Verteilung

Für die Weibull-Verteilung mit Parametern k (Form) und λ (Skala) unterscheiden sich die Schätzer erheblich:

MLE: Erfordert numerische Lösung des Gleichungssystems:

$$\frac{1}{k} + \frac{1}{n}\sum_{i=1}^{n} \ln(x_i) - \frac{\sum_{i=1}^{n} x_i^k \ln(x_i)}{\sum_{i=1}^{n} x_i^k} = 0$$

Dabei ist

$$\hat{\lambda} = \left(\frac{1}{n}\sum_{i=1}^{n} x_i^k\right)^{1/k}.$$

Momentenmethode: Verwendet die ersten beiden Momente und führt zu geschlossenen Formeln, die über die Gamma-Funktion ausgedrückt werden. Diese sind jedoch weniger präzise als die MLE-Schätzer.

B.6 Zusammenfassung

Dieses Kapitel beleuchtet zwei zentrale Methoden der statistischen Parameterschätzung: die Maximum-Likelihood-Schätzung (MLE) und die Momentenmethode. Beide dienen der Inferenz unbekannter Parameter aus Daten und bilden die Basis für Modellselektion (z. B. AIC).

MLE-Prinzip: Die Likelihood-Funktion $L(\theta|x)$ misst, wie plausibel Daten unter gegebenen Parametern sind. Der ML-Schätzer $\hat{\theta}_{ML}$ maximiert L oder die Log-Likelihood $\ell(\theta|x)$. Für analytisch unlösbare Fälle eignen sich numerische Verfahren wie Newton-Raphson. Asymptotisch ist MLE konsistent, normalverteilt und effizient (basierend auf der Fisher-Information).

Momentenmethode: Setzt theoretische Momente ($\mathbb{E}[X^k]$) gleich empirischen Momenten aus der Stichprobe und löst nach Parametern auf. Sie ist einfach, oft explizit lösbar und asymptotisch konsistent/normalverteilt, aber weniger effizient als MLE.

Vergleich: MLE ist universell anwendbar, effizient, aber rechenintensiv und sensibel auf Ausreißer. Die Momentenmethode ist robuster und einfacher, eignet sich für Standardverteilungen (wo sie oft mit MLE übereinstimmt), ist aber bei komplexen Modellen eingeschränkt. In der Praxis ergänzen sie sich, z. B. in maschinellem Lernen oder Regressionsanalysen, und erfordern Berücksichtigung von Stichprobengröße und Modellannahmen für zuverlässige Inferenz.

Kolmogorov-Smirnov Anpassungstest

C

Der Kolmogorov-Smirnov Anpassungstest ist ein nichtparametrischer statistischer Test, der verwendet wird, um zu prüfen, ob eine gegebene Stichprobe aus einer bestimmten theoretischen Verteilung stammt.

Der Kolmogorov-Smirnov Test wurde 1933 von Andrey Kolmogorov vorgeschlagen und 1948 von Nikolai Smirnov erweitert. Er ist einer der am häufigsten verwendeten Anpassungstests in der Statistik.

Der **Kolmogorov-Smirnov Test** prüft die Nullhypothese H_0, dass eine Stichprobe $X_1, X_2, \ldots, X_n$ aus einer spezifizierten kontinuierlichen Verteilung mit Verteilungsfunktion $F_0(x)$ stammt.

Warnung: Nur für "echte" theoretische Verteilungen

Die bestimmte theoretische Verteilung $F_0(x)$ gegen die getestet wird, darf beim Kolmogorov-Smirnov-Anpassungstest nicht aus der Stichprobe berechnet werden. Sie muss vielmehr unabhängig davon vorgegeben werden. Für andere Fälle, die die theoretische Verteilung aus der Stichprobe ableiten (wie in Kapitel 11), werden die Verfahren in Abschnitt C.3 und C.4 gezeigt.

C.1 Theoretischer Hintergrund

Ausgangspunkt für den Kolmogorov-Smirnov Anpassungstest ist die empirische Verteilungsfunktion:

Die **empirische Verteilungsfunktion** $F_n(x)$ wird wie folgt berechnet:

$$F_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{X_i \leq x}$$

wobei $\mathbf{1}_{X_i \leq x}$ die Indikatorfunktion ist, die 1 ist, wenn $X_i \leq x$ und 0 sonst.

Die empirische Verteilungsfunktion für n Stichprobedaten wird folgendermaßen berechnet:

Man sortiert die Daten und berechnet den bis zu $X_{(i)}$ vorkommenden Datenanteil als $F(X_{(i)})$.

1. Sortiere die Stichprobenwerte: $X_{(1)} \leq X_{(2)} \leq \ldots \leq X_{(n)}$
2. Für jeden sortierten Wert $X_{(i)}$ ist $F_n(X_{(i)}) = \frac{i}{n}$.

Für jedes $X_{(i)}$ berechnet man anschließend den Abstand zwischen $F_n(X_{(i)})$ und $F_0(X_{(i)})$. Dies führt zur Kolmogorov-Smirnov-Teststatistik:

Die **Kolmogorov-Smirnov-Teststatistik** D_n ist definiert als:

$$D_n = \max_x |F_n(x) - F_0(x)|.$$

Die Statistik ist die maximale Abweichung zwischen der empirischen und der angenommenen, theoretischen Verteilungsfunktion.

D_n ist damit der größte Abstand zwischen empirischer und theoretischer Verteilungsfunktion.

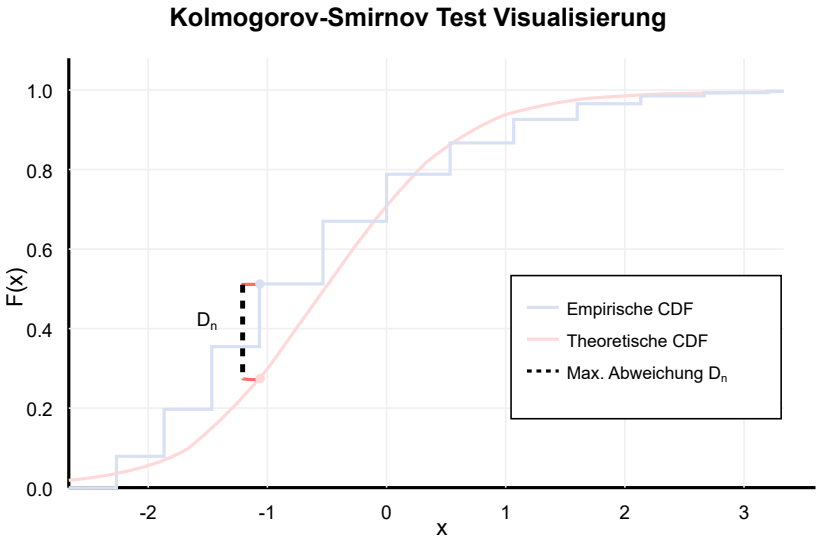


Abbildung C.1: Kolmogorov-Smirnov Test Visualisierung: Die blaue Treppenfunktion zeigt die empirische Verteilungsfunktion (CDF=Cumulative Distribution Function) der Stichprobendaten, während die rote glatte Kurve die theoretische Verteilungsfunktion darstellt. Die schwarz gestrichelte Linie markiert die maximale vertikale Abweichung D_n zwischen beiden Funktionen, welche die KS-Teststatistik darstellt.

Unter der Annahme der Nullhypothese (H_0), dass die Stichprobendaten tatsächlich aus der theoretischen Verteilung stammen, folgt die Statistik $\sqrt{n} \cdot D_n$ für ausreichend große Stichprobenumfänge ($n \geq 35$) der Kolmogorov-Verteilung.

Die punktweise Varianz der empirischen Verteilungsfunktion ist $\mathrm{Var}(F_n(x)) = F_0(x) \cdot (1 - F_0(x))/n$. Die $\sqrt{n}$-Normierung lässt sich heuristisch damit begründen, dass die punktweise Standardabweichung von der Größenordnung $1/\sqrt{n}$ ist.

Die **Verteilungsfunktion der Kolmogorov-Verteilung** ist definiert als:

$$K(x) = 1 - 2\sum_{k=1}^{\infty}(-1)^{k-1}e^{-2k^2x^2} \quad \text{für } x > 0$$

Für praktische Anwendungen wird oft eine Vereinfachung verwendet, bei der nur das erste Glied der unendlichen Summe (für $k = 1$) berücksichtigt wird. Dies führt zu einer guten Näherung der Verteilungsfunktion:

Die **Approximative Verteilungsfunktion der Kolmogorov-Verteilung** ist

$$K(x) \approx 1 - 2e^{-2x^2} \quad \text{für } x > 0$$

Andrey Kolmogorov hat 1933 bewiesen, dass die Verteilung der skalierten Statistik $\sqrt{n}D_n$ für große n gegen eine bestimmte, von der ursprünglichen Verteilung unabhängige Grenzkurve konvergiert — die heute als Kolmogorov-Verteilung bekannte Verteilung.

Mithilfe dieser Näherungsformel können die für den Test benötigten Quantile (kritische Werte) $k_{1-\alpha}$ einfach hergeleitet werden. Ein Quantil $k_{1-\alpha}$ ist der Wert, bei dem die Wahrscheinlichkeit, diesen zu überschreiten, gleich dem Signifikanzniveau α ist. Man löst dazu die Gleichung $\alpha = P(X > x) = 1 - K(x) \approx 2e^{-2x^2}$ nach x auf. Dies führt zur folgenden Näherungsformel für das $(1 - \alpha)$-Quantil:

$$k_{1-\alpha} \approx \sqrt{-\frac{1}{2}\ln\left(\frac{\alpha}{2}\right)}$$

Basierend auf dieser Formel ergeben sich die folgenden genäherten Quantile für gängige Signifikanzniveaus:

Signifikanzniveau α	Quantil $(1-\alpha)$	Kritischer Wert $k_{1-\alpha}$
0.20	0.80	1.073
0.10	0.90	1.224
0.05	0.95	1.358
0.02	0.98	1.517
0.01	0.99	1.628

Tabelle C.1: Kritische Werte der Kolmogorov-Verteilung

Hinweis: kritischer Wert

Ist $\sqrt{n} \cdot D_n$ - und damit der größte Abstand zwischen der empirischen und der theoretischen Verteilung multipliziert mit der Skalierung $\sqrt{n}$ - größer als $k_{1-\alpha}$ muss die Nullhypothese, dass die Stichprobendaten aus der theoretischen Verteilung stammen, auf einem Niveau von α abgelehnt werden.

Die Universalität der Kolmogorov-Verteilung stellt eine besondere Eigenschaft dar, da dieselbe Testverteilung für jeden Kolmogorov-Smirnov Test verwendet werden kann, unabhängig von der spezifischen theoretischen Verteilung gegen die die Anpassungsgüte geprüft wird.

Analog kann der p-Wert für den Kolmogorov-Smirnov-Test berechnet werden:

Der p-Wert ist die Wahrscheinlichkeit, einen Wert größer oder gleich der beobachteten Teststatistik $x = \sqrt{n}D_n$ zu erhalten. Das ist die Gegenwahrscheinlichkeit zur Verteilungsfunktion:

$$p = P(K \geq x) = 1 - K(x)$$

Jetzt setzt man die Näherungsformel für $K(x)$ ein:

$$p \approx 1 - (1 - 2e^{-2x^2})$$

Durch Vereinfachen erhält man eine direkte Formel für den p-Wert:

$$p \approx 2e^{-2x^2}$$

wobei x die skalierte Teststatistik $\sqrt{n}D_n$ ist.

Hinweis: p-Wert

Für $x = \sqrt{n} \cdot D_n$ ist der p-Wert des Kolmogorov-Smirnov-Tests $2e^{-2x^2}$. Falls der p-Wert kleiner als α ist, muss die Nullhypothese, dass die Stichprobendaten aus der theoretischen Verteilung stammen, auf einem Niveau von α abgelehnt werden.

Folgendes Beispiel fasst das Vorgehen zusammen:

Merke:

Der KS-Test ist **nicht anwendbar**, wenn die Verteilungsparameter der theoretischen Verteilungsfunktion aus den Stichproben geschätzt werden. Bei diskreten Daten ist der KS-Test zu konservativ, d.h. bestätigt die H_0-Hypothese bevorzugt. Zudem kann er bei großen Stichprobengrößen n auch praktisch unwichtige Abweichungen als signifikant identifizieren.

Anwendungsfall: Kundenumsatz-Verteilung

Ein Datensatz enthält 1.000 Werte des Merkmals „Kundenumsatz". Es soll geprüft werden, ob diese normalverteilt sind.

Gegeben:
Stichprobe: $n = 1.000$ Umsatzwerte
Hypothese: H_0: Aufgrund anderen Kundendaten (siehe die Warnung zu Beginn des Kapitels) wird angenommen, dass eine Normalverteilung mit $\mu = 5.000$ und $\sigma = 1.200$ vorliegt.
Signifikanzniveau: $\alpha = 0,05$

Durchführung:

1. Berechnung der empirischen Verteilungsfunktion für alle 1.000 sortierten Werte
2. Berechnung der theoretischen Verteilungsfunktion $F_0(x) = \Phi(\frac{x-5000}{1200})$, wobei Φ die Standard-Normalverteilung ist
3. Bestimmung der maximalen Abweichung: $D_{1000} = \max_i |F_{1000}(X_{(i)}) - F_0(X_{(i)})|$

Angenommen, die Berechnung ergibt: $D_{1000} = 0,055$ und damit skaliert $\sqrt{1000} \cdot 0,055 = 1,739$

Kritischer Wert ist $k_{1-\alpha} = 1.358$ (vgl. Tabelle C.1)

Entscheidung:
Wegen $1.739 > 1.358$ wird H_0 abgelehnt. Die Daten sind nicht normalverteilt. Der P-Wert ist $2e^{-2x^2} = 2e^{-2(\sqrt{1000} \cdot 0,055)^2} = 0,0047$.

C.2 2-Stichproben-Test

Der Kolmogorov-Smirnov-Test kann auch erweitert werden, um zu prüfen, ob zwei unabhängige Stichproben aus derselben Verteilung stammen.

Im Gegensatz zum Ein-Stichproben-Test wird hier nicht gegen eine theoretische Verteilung, sondern die empirische Verteilungsfunktion der einen Stichprobe gegen die der anderen Stichprobe getestet.

Der 2-Stichproben-Test ist einer der wichtigsten Tests in der Statistik und hat zahreiche Anwendungen - vgl. Kapitel 12.

Der **Kolmogorov-Smirnov 2-Stichproben-Test** prüft die Nullhypothese H_0, dass zwei unabhängige Stichproben $X_1, \ldots, X_n$ und $Y_1, \ldots, Y_m$ aus derselben kontinuierlichen Verteilung stammen. Die Verteilungsfunktion selbst muss dabei nicht bekannt sein. $H_0 : F_X(x) = F_Y(x)$ für alle x.

Die Teststatistik ist analog zum Ein-Stichproben-Fall die maximale vertikale Distanz zwischen den beiden empirischen Verteilungsfunktionen $F_n(x)$ und $G_m(x)$.

Die **Teststatistik des 2-Stichproben-Tests** $D_{n,m}$ ist definiert als:

$$D_{n,m} = \sup_x |F_n(x) - G_m(x)|.$$

wobei n und m die jeweiligen Stichprobengrößen sind.

Die Skalierung der Teststatistik unterscheidet sich vom Ein-Stichproben-Fall, da nun die Unsicherheit beider Stichproben berücksichtigt werden muss.

Die mit skalierte $\sqrt{\frac{nm}{n+m}}$ Teststatistik

$$x = \sqrt{\frac{nm}{n+m}} \cdot D_{n,m}$$

konvergiert für ausreichend große Stichprobenumfänge (in der Praxis oft $n, m \geq 35$) ebenfalls gegen die Kolmogorov-Verteilung.

Die Entscheidungsfindung erfolgt analog zum Ein-Stichproben-Test. Der berechnete Wert x wird mit dem kritischen Wert $k_{1-\alpha}$ aus Tabelle C.1 verglichen oder zur Berechnung des entsprechenden p-Wertes verwendet.

Der Skalierungsfaktor $\sqrt{\frac{nm}{n+m}}$ entsteht aus der kombinierten Varianz der beiden empirischen Verteilungsfunktionen. Unter H_0 ist die Varianz der Differenz $F_n(x) - G_m(x)$ proportional zu $\frac{1}{n} + \frac{1}{m} = \frac{n+m}{nm}$. Die Skalierung normiert die Teststatistik mit der Wurzel des Kehrwerts dieses Terms.

Entscheidungsregel 2-Stichproben-Test

Ist der Wert der skalierten Teststatistik $\sqrt{\frac{nm}{n+m}} \cdot D_{n,m}$ größer als der kritische Wert $k_{1-\alpha}$, so wird die Nullhypothese, dass beide Stichproben aus derselben Verteilung stammen, zum Signifikanzniveau α abgelehnt. Alternativ wird H_0 abgelehnt, wenn der p-Wert $p \approx 2e^{-2x^2}$ kleiner als α ist.

Ein Anwendungsbeispiel verdeutlicht das Vorgehen:

Anwendungsfall: Kundenumsätze zweier Filialen

Es soll geprüft werden, ob sich die Verteilung der Kundenumsätze zwischen zwei Filialen (A und B) unterscheidet.

Gegeben:
Stichprobe A: $n = 200$ Umsatzwerte
Stichprobe B: $m = 250$ Umsatzwerte
Hypothese: H_0: Die Umsätze in Filiale A und B stammen aus derselben Verteilung.
Signifikanzniveau: $\alpha = 0,05$

Durchführung:

1. Berechnung der empirischen Verteilungsfunktion $F_{200}(x)$ für Filiale A.
2. Berechnung der empirischen Verteilungsfunktion $G_{250}(x)$ für Filiale B.

3. Bestimmung der maximalen Abweichung:
   $D_{200,250} = \sup_x |F_{200}(x) - G_{250}(x)|$

Angenommen, die Berechnung ergibt: $D_{200,250} = 0,12$

Skalierung der Teststatistik: Der Skalierungsfaktor ist $\sqrt{\frac{200 \cdot 250}{200+250}} = \sqrt{\frac{50000}{450}} \approx 10,54$. Die skalierte Teststatistik ist $x = 10,54 \cdot 0,12 \approx 1,265$.

Der kritische Wert für $\alpha = 0,05$ ist $k_{0,95} = 1,358$ (vgl. Tabelle C.1).

Entscheidung:
Wegen $1,265 < 1,358$ kann H_0 nicht abgelehnt werden. Es gibt auf dem 5%-Niveau keinen statistisch signifikanten Nachweis, dass sich die Umsatzverteilungen der beiden Filialen unterscheiden. Der P-Wert ist $p \approx 2e^{-2(1,265)^2} \approx 2e^{-3,19} \approx 0,083$. Da $0,083 > 0,05$, wird die Nullhypothese ebenfalls nicht verworfen.

C.3 Lilliefors-Test: Eine Modifikation für geschätzte Parameter

C.3.1 Definition und Grundlagen

Der Lilliefors-Test ist eine wichtige Modifikation des Kolmogorov-Smirnov-Anpassungstests. Er wird speziell für den häufigen Anwendungsfall entwickelt, dass die Parameter der theoretischen Verteilung (wie der Mittelwert μ oder die Standardabweichung σ) nicht vorab bekannt sind, sondern aus der Stichprobe geschätzt werden müssen.

Der Test wurde 1967 von Hubert Lilliefors vorgestellt, um die Limitierung des KS-Tests zu beheben. Er publizierte Tabellen für den Test auf Normalverteilung (1967) und Exponentialverteilung (1969).

Der **Lilliefors-Test** prüft die Nullhypothese H_0, dass eine Stichprobe aus einer bestimmten Verteilungsfamilie (z.B. der Familie der Normalverteilungen) stammt, deren Parameter aus der Stichprobe selbst geschätzt wurden.

Anwendung: Wenn Parameter unbekannt sind

Im Gegensatz zum klassischen KS-Test wird der Lilliefors-Test verwendet, wenn die theoretische Verteilung $F_0(x)$ durch Schätzung von Parametern wie dem Stichprobenmittelwert $\bar{X}$ und der Stichprobenstandardabweichung s an die Daten angepasst wird. Dies ist in der Praxis der Regelfall.

C.3.2 Teststatistik

Die Berechnung der Teststatistik selbst ist identisch zu der des Kolmogorov-Smirnov-Tests. Es wird ebenfalls die maxi-

male absolute Abweichung zwischen der empirischen Verteilungsfunktion $F_n(x)$ und der nun angepassten theoretischen Verteilungsfunktion $F_0(x)$ bestimmt.

Die **Lilliefors-Teststatistik** D_n ist definiert als:

$$D_n = \max_x |F_n(x) - F_0(x)|,$$

wobei die Parameter von $F_0(x)$ aus der Stichprobe geschätzt sind.

Der entscheidende Unterschied liegt in der Verteilung der Teststatistik. Da die theoretische Verteilung künstlich an die Daten angenähert wurde, ist der Abstand D_n tendenziell kleiner. Die Verwendung der klassischen Kolmogorov-Verteilung wäre daher zu konservativ (würde H_0 zu selten ablehnen).

C.3.3 Kritische Werte

Achtung: Andere kritische Werte!

Die Teststatistik D_n wird beim Lilliefors-Test **nicht** gegen die Kolmogorov-Verteilung getestet. Stattdessen werden spezielle, strengere (d.h. kleinere) kritische Werte verwendet. Diese wurden durch Monte-Carlo-Simulationen ermittelt und hängen von der getesteten Verteilungsfamilie ab (z.B. Normalverteilung, Exponentialverteilung etc.).

Für den häufigen Fall des Tests auf Normalverteilung können die kritischen Werte für größere Stichproben ($n > 30$) wie folgt angenähert werden:

Signifikanzniveau α	Kritischer Wert
0.20	$\frac{0{,}768}{\sqrt{n}}$
0.10	$\frac{0{,}819}{\sqrt{n}}$
0.05	$\frac{0{,}886}{\sqrt{n}}$
0.02	$\frac{0{,}955}{\sqrt{n}}$
0.01	$\frac{1{,}031}{\sqrt{n}}$

Tabelle C.2: Kritische Werte des Lilliefors-Tests für Normalverteilung

C.3.4 Anwendungsbeispiel

Anwendungsfall: Kundenumsatz-Verteilung (geschätzt)

Ein Datensatz enthält 1.000 Werte des Merkmals „Kundenumsatz“. Es soll geprüft werden, ob diese normalverteilt sind. Mittelwert und Standardabweichung sind unbekannt.

Gegeben:
Stichprobe: $n = 1.000$ Umsatzwerte
Hypothese: H_0: Die Daten sind normalverteilt.

Signifikanzniveau: $\alpha = 0,05$

Durchführung:

1. Schätze die Parameter aus den Daten:
   z.B. $\bar{X} = 5.000$ und $s = 1.200$
2. Berechnung der theoretischen Verteilungsfunktion
   $F_0(x) = \Phi(\frac{x-\bar{X}}{s})$, mit den geschätzten Werten
3. Bestimmung der maximalen Abweichung:
   $D_{1000} = \max_x |F_{1000}(x) - F_0(x)|$

Angenommen, die Berechnung ergibt wieder: $D_{1000} = 0,055$

Kritischer Wert für den Lilliefors-Test bei $n = 1000$ und $\alpha = 0,05$ ist (vgl. Tabelle C.2):
$D_{kritisch} \approx \frac{0,886}{\sqrt{1000}} \approx 0,028$

Entscheidung:
Wegen $D_{1000} = 0,055 > D_{kritisch} = 0,028$ wird H_0 abgelehnt. Die Daten sind nicht normalverteilt.

Beachten Sie, dass der kritische Wert (0,028) viel kleiner ist als der des KS-Tests (0,043). Die Nullhypothese wird daher leichter abgelehnt, was den konservativen Effekt der Parameterschätzung korrigiert.

Hinweis: p-Wert beim Lilliefors-Test

Da die Verteilung der Teststatistik nicht analytisch herleitbar ist, gibt es keine einfache Formel zur Berechnung des p-Wertes (wie $p \approx 2e^{-2x^2}$). Statistiksoftware ermittelt den p-Wert stattdessen durch interne, hochpräzise Näherungsformeln oder Tabellen, die auf den Simulationsergebnissen basieren.

C.4 Bootstrap-Anpassungstest: Die universelle Simulationsmethode

C.4.1 Grundlagen des Bootstrap-Anpassungstests

Das Bootstrapping ist eine moderne, computerintensive Simulationsmethode, die eine extrem flexible Alternative zu klassischen Anpassungstests wie dem Kolmogorov-Smirnov- oder Lilliefors-Test darstellt.

Der Hauptvorteil des Bootstrap-Ansatzes ist seine Universalität: Er kann für praktisch jede beliebige Verteilungsfamilie angewendet werden, auch wenn für diese keine vordefinierten kritischen Werte existieren.

Das Bootstrap-Verfahren wurde 1979 von Bradley Efron eingeführt und hat die moderne Statistik revolutioniert, indem es die Analyse komplexer Probleme durch Rechenleistung ermöglichte.

Der **Bootstrap-Anpassungstest** prüft die Nullhypothese H_0, dass eine Stichprobe aus einer bestimmten Verteilungsfamilie stammt, indem die Verteilung der Teststatistik durch wiederholtes Ziehen aus der an die Daten angepassten Verteilung simuliert wird.

Der Prozess, auch als parametrischer Bootstrap bekannt, folgt einer klaren Logik. Als Teststatistik wird oft die KS-Statistik D_n verwendet, aber auch andere Maße sind möglich.

C.4.2 Ablauf des Bootstrap-Anpassungstests

Angenommen, wir wollen testen, ob unsere Daten einer Gamma-Verteilung folgen.

Erzeuge die Verteilung selbst! Anstatt die Teststatistik D_n mit einer festen, theoretischen Verteilung (wie der Kolmogorov- oder Lilliefors-Tabelle) zu vergleichen, beantwortet das Bootstrapping die Frage: „Wenn die Daten wirklich aus der von mir geschätzten Verteilung stammen, wie würde dann die Verteilung der Teststatistiken aussehen?"Man erzeugt also eine maßgeschneiderte Vergleichsverteilung, die perfekt zur Datensituation passt.

Möglicher Ablauf Bootsrap

1. **Anpassung (Fit):** Schätze die Parameter der gewünschten Verteilung (z.B. Form k und Skala θ der Gamma-Verteilung) aus der Originalstichprobe. Dies ergibt die bestmögliche theoretische Verteilung F_0.
2. **Beobachtete Teststatistik:** Berechne die KS-Teststatistik D_{obs} als maximalen Abstand zwischen der empirischen Verteilungsfunktion F_n der Originaldaten und der in Schritt 1 angepassten Verteilung F_0. Dieser Wert ist unser Referenzwert.
3. **Simulation (Bootstrap-Schleife):** Wiederhole die folgenden Schritte sehr oft (z.B. $B = 10.000$ Mal):
   a) **Generieren:** Erzeuge eine neue "Bootstrap-Stichprobe" mit Umfang n, indem du n Zufallszahlen aus der in Schritt 1 angepassten Verteilung F_0 ziehst.
   b) **Anpassen & Testen:** Behandle diese neue Stichprobe wie echte Daten: Schätze ihre Parameter (z.B. k_{boot} und θ_{boot}) und berechne die KS-Statistik D_{boot} für diese neue Stichprobe gegen ihre eigene, neu angepasste Verteilung.
   c) **Speichern:** Lege den Wert von D_{boot} ab.
4. **p-Wert berechnen:** Nach Abschluss der Simulationen hat man eine Verteilung von B Bootstrap-Teststatistiken ($D_{boot,i}$, $i = 1, \ldots, B$). Der p-Wert ist der Anteil dieser simulierten Statistiken, die größer oder gleich der ursprünglich beobachteten Statistik D_{obs} sind.

$$p = \frac{\text{Anzahl}(D_{boot} \geq D_{obs})}{B}$$

C.4.3 Anwendungsbeispiel

Anwendungsfall: Kundenumsatz (Gamma-Verteilung)

Die Hypothese lautet nun, dass die 1.000 Kundenumsätze einer Gamma-Verteilung folgen, da die Daten rechtsschief sind und keine negativen Werte annehmen können.

Hypothese: H_0: Die Daten stammen aus einer Gamma-Verteilung.

Durchführung (konzeptionell):

1. Schätze die Parameter k und θ für die Gamma-Verteilung aus den 1.000 Umsätzen.
2. Berechne die KS-Statistik D_{obs} zwischen den empirischen Daten und dieser besten Gamma-Verteilung. Angenommen, das Ergebnis ist $D_{obs} = 0,021$.
3. Starte die Simulation:
   - Ziehe 1.000 Zufallszahlen aus der in (1) gefundenen Gamma-Verteilung.
   - Schätze für diese neuen Daten wieder k_{boot} und θ_{boot} und berechne D_{boot}.
   - Wiederhole dies 10.000 Mal.

Entscheidung:
Angenommen, von den 10.000 simulierten D_{boot}-Werten waren 850 größer oder gleich unserem beobachteten Wert $D_{obs} = 0,021$.

Der p-Wert ist dann $p = \frac{850}{10.000} = 0,085$.

Da $p = 0,085 > \alpha = 0,05$, wird die Nullhypothese beibehalten. Es gibt keine ausreichende Evidenz, um die Annahme einer Gamma-Verteilung abzulehnen.

C.5 Zusammenfassung

Zusammenfassend können folgende Vor- und Nachteile genannt werden:

Flexibilität wird durch Rechenintesnität erreicht. Mit immer größerer Rechnerleistung ist Bootstrapping immer weiter verbreitet

Vorteile: Das Verfahren ist extrem universell und kann auf jede Verteilung mit geschätzten Parametern angewendet werden, für die keine Standardtests existieren. Es ist oft mächtiger als der Lilliefors-Test.

Nachteile: Es ist rechenintensiv und liefert kein Ergebnis über eine einfache Formel, sondern nur durch die Simulation. Die Ergebnisse können bei jeder Durchführung minimal variieren (obwohl dieser Effekt bei einer hohen Anzahl an Wiederholungen vernachlässigbar ist).

Hauptkomponentenanalyse (PCA) D

Die Hauptkomponentenanalyse, weithin bekannt als PCA (Principal Component Analysis), ist eine der fundamentalsten und am weitesten verbreiteten Techniken in der multivariaten Datenanalyse und im maschinellen Lernen. Sie dient primär der Dimensionsreduktion und der Merkmalsextraktion, indem sie die Struktur von hochdimensionalen Daten vereinfacht, ohne dabei einen signifikanten Informationsverlust zu erleiden.

Ein sehr gutes Buch, das die Principial Component Anaylse sehr ausführlich behandelt, ist [50].

D.1 Herleitung

Zur besseren Verständlichkeit wird im Folgenden ein 3-dimensionaler Datenraum verwendet. Das Prinzip ist aber auf beliebige endliche Dimensionen anwendbar.

In einem dreidimensionalen Raum ist jeder Punkt durch Koordinaten wie $(x_1, x_2, x_3) \in \mathbb{R}^3$ bezeichnet. Diese Koordinaten beschreiben den Punkt als Linearkombination der Standardbasisvektoren:

Der Begriff „Basis“ stammt aus dem Griechischen und bedeutet „Grundlage“ oder „Fundament“ - passend für die fundamentale Rolle in der Linearen Algebra.

$$x_1 \cdot e_1 + x_2 \cdot e_2 + x_3 \cdot e_3 = x_1 \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + x_2 \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + x_3 \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Eine Menge von Vektoren heißt Basis, wenn aus ihr alle Punkte des Raumes über Linearkombinationen darstellbar sind (Erzeugendensystem) und sich kein Basisvektor über Linearkombinationen der anderen Basisvektoren darstellen lässt (linear unabhängig).

Eine **Basis** eines Vektorraums ist eine Menge von linear unabhängigen Vektoren, die den gesamten Vektorraum aufspannen. Jeder Vektor im Raum kann eindeutig als Linearkombination der Basisvektoren dargestellt werden.

D.1.1 Basiswechsel

Die Anzahl der Basisvektoren entspricht im endlichen Fall immer der Dimension des Raumes (in unserem Beispiel 3). Es gibt unendlich viele Möglichkeiten, eine Basis auszuwählen. Wird beispielsweise anstelle von e_1, e_2, e_3 eine andere

Basis verwendet, z.B. v_1, v_2, v_3, dann ändern sich auch die Koordinaten (x_1, x_2, x_3) des ausgewählten Punktes.

Beispiel: Angenommen wir betrachten die neue Basis

$$v_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \quad v_2 = \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix}, \quad v_3 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \tag{D.1}$$

Der Punkt $(1, 1, 1)^T$ auf Basis der Standardbasis entspricht dann dem Punkt $(0, 0.5, 1)^T$ für die Basis v_1, v_2, v_3, da

$$0 \cdot \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.5 \cdot \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix} + 1 \cdot \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \tag{D.2}$$

Der Übergang von $(1, 1, 1)^T$ von der Standardbasis auf $(0, 0.5, 1)^T$ in der Basis v_1, v_2, v_3 heißt Basiswechsel.

D.2 Principal Component Analysis

Die PCA wurde 1901 von Karl Pearson entwickelt und unabhängig davon 1933 von Harold Hotelling wiederentdeckt. Hotelling nannte sie „Principal Component Analysis“ - ein Name, der sich durchsetzte.

Wenn wir viele Punkte (Daten) im Raum haben, können wir eine Basis so wählen, dass sie die Struktur der Daten besser beschreibt. Die Idee der PCA ist, eine Basis zu finden, bei der:

- Der erste Basisvektor v_1 die meiste Varianz der Daten erklärt, d.h. die Hauptstreurichtung der Daten ist.
- Der zweite Basisvektor v_2 orthogonal (also senkrecht bzw. das Skalarprodukt ist 0) zu v_1 ist und die zweitmeiste Varianz der Daten erklärt.
- Der dritte Basisvektor v_3 orthogonal zu den beiden anderen ist und die restliche Varianz erklärt.

Das Ergebnis ist eine neue Orthogonalbasis, die optimal auf die Verteilung der Daten abgestimmt ist.

Eigenvektoren bilden eine solche Basis:

Für eine quadratische Matrix A ist ein Vektor $v \neq 0$ ein **Eigenvektor** zum **Eigenwert** λ, wenn gilt: $A \cdot v = \lambda \cdot v$ Der Eigenvektor zeigt eine Richtung an, in der die Matrix nur eine Streckung (oder Stauchung) um den Faktor λ bewirkt.

Mit Pricincipal Component oder Hauptkomponente bezeichnet man bestimmte Eigenvektoren:

Eine **Hauptkomponente** (Principal Component) ist ein Eigenvektor der Kovarianzmatrix der zentrierten Daten. Die erste Hauptkomponente entspricht dem Eigenvektor mit dem größten Eigenwert und zeigt die Richtung der maximalen Varianz in den Daten an. Die weiteren Hauptkomponenten sind orthogonal zueinander und erklären absteigend die verbleibende Varianz.

d.h. die Eigenvektoren werden gemäß der Größe ihrer Eigenwerte geordnet und dann durchnummeriert.

D.3 Vorgehen

Die PCA beginnt damit, die Daten zu zentrieren. Dazu wird der Mittelwert jeder Dimension von allen Punkten subtrahiert, sodass der Datensatz um den Ursprung zentriert ist.

Die Zentrierung ist mathematisch notwendig, da die Kovarianzmatrix per Definition die Varianz *um den Mittelwert* misst. Ohne Zentrierung würde PCA hauptsächlich die Richtung zum Ursprung finden.

Wenn die Variablen sehr unterschiedliche Skalenordnungen aufweisen (z.B. eine Variable in Kilometern, eine andere in Gramm), werden die Daten zusätzlich standardisiert. Dies geschieht über den Z-Score. Ein Datenpunkt $x \in \mathbb{R}^n$ hat dann folgende Normierung:

$$x_{\text{std},i} = \frac{x_i - \mu_i}{\sigma_i}$$

wobei μ_i der Mittelwert und σ_i die Standardabweichung der jeweiligen Variable i ist. Dadurch erhalten alle Variablen Mittelwert 0 und Standardabweichung 1, was verhindert, dass Variablen mit großen Werten die PCA dominieren.

Dies wird auch als Normalisierung oder Z-Standardisierung bezeichnet.

Standardisierung ist entscheidend

Vor der Anwendung einer Hauptkomponentenanalyse sollten die Daten standardisiert werden (Z-Score-Transformation), falls die Variablen unterschiedliche Einheiten oder Größenordnungen haben. Ohne Standardisierung würden Variablen mit größeren Werten die Analyse dominieren und die PCA verfälschte Ergebnisse liefern.

Danach wird die Varianz-Kovarianz-Matrix berechnet. Diese Matrix beschreibt, wie die Dimensionen miteinander variieren und gibt Aufschluss über die Streuung der Daten in verschiedenen Richtungen. Im Falle eine Z-Standardisierung ist Varianz-Kovarianz-Matrix eine Korrelationsmatrix.

Der Spektralsatz garantiert, dass symmetrische Matrizen (wie die Kovarianzmatrix) immer diagonalisierbar sind und orthogonale Eigenvektoren besitzen. Dies ist die mathematische Grundlage der PCA.

Im nächsten Schritt erfolgt die Eigenwertzerlegung der Varianz-Kovarianz-Matrix. Dabei werden die Eigenvektoren und Eigenwerte berechnet. Die Eigenvektoren sind die neuen

Basisvektoren, welche die Richtung der maximalen Datenstreuung angeben. Die zugehörigen Eigenwerte geben an, wie viel Varianz jeder Basisvektor erklärt.

Die Eigenvektoren werden anschließend nach den absteigenden Eigenwerten sortiert. Der Eigenvektor mit dem größten Eigenwert wird zur ersten Hauptkomponente, da er die meiste Varianz der Daten erklärt. Der zweite Eigenvektor mit dem nächstgrößten Eigenwert wird zur zweiten Hauptkomponente und so weiter.

Die **erklärte Varianz** jeder Hauptkomponente entspricht dem zugehörigen Eigenwert. Der Anteil der erklärten Varianz berechnet sich als:

$$\text{Erklärte Varianz}_i = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}$$

wobei λ_i der i-te Eigenwert ist. Die Summe aller Eigenwerte entspricht der Gesamtvarianz der ursprünglichen Daten.

In der PCA taucht oft der Begriff Loading auf:

Die **Loadings** sind die Komponenten der Eigenvektoren (Hauptkomponenten). Sie zeigen an, wie stark jede ursprüngliche Variable zur jeweiligen Hauptkomponente beiträgt. Hohe Loadings (betragsmäßig) bedeuten einen starken Einfluss der Variable auf die Hauptkomponente.

Durch die Normierung der Eigenvektoren liegen die *Loadings* im Bereich von –1 bis 1.

Beispiel: Berechnung und Interpretation der Loadings

Angenommen, eine PCA wird auf einem standardisierten Datensatz mit vier Variablen zur Fahrzeugbewertung durchgeführt. Die Eigenwertzerlegung der Korrelationsmatrix ergibt für die ersten beiden Hauptkomponenten:

Eigenwerte: $\lambda_1 = 2.8$, $\lambda_2 = 0.9$

Eigenvektoren (= Standard-Loadings):

$$\mathbf{v}_1 = \begin{pmatrix} 0.33 \\ 0.35 \\ 0.31 \\ -0.18 \end{pmatrix}, \quad \mathbf{v}_2 = \begin{pmatrix} -0.32 \\ -0.26 \\ -0.16 \\ -0.90 \end{pmatrix} \tag{D.3}$$

Loadings-Matrix: Die Loadings sind direkt die Komponenten der Eigenvektoren $l_{ij} = v_{ij}$:

Variable	PC1 Loading	PC2 Loading
Preis	0.33	−0.32
PS-Leistung	0.35	−0.26
Gewicht	0.31	−0.16
Verbrauch (L/100km)	−0.18	−0.90

Interpretation: PC1 hat hohe positive Loadings für Preis (0.35), PS-Leistung (0.35) und Gewicht (0.31), aber ein negatives Loading für Verbrauch (-0.18). Diese Komponente könnte als „Fahrzeuggröße & Leistung“ interpretiert werden. Hohe Werte auf PC1 repräsentieren große, teure und leistungsstarke Fahrzeuge mit tendenziell niedrigerem Verbrauch.

PC2 wird stark vom Verbrauch dominiert (-0.90), während die anderen Variablen moderate negative Loadings haben. Diese Komponente könnte als „Kraftstoffeffizienz“ interpretiert werden, wobei niedrige Werte auf PC2 (negative Scores) auf Fahrzeuge mit hohem Verbrauch hindeuten.

Hinweis: Die Beträge der Loadings zeigen die Stärke des Beitrags jeder Variable zur jeweiligen Hauptkomponente an.

Der Abstand entlang der i-ten Hauptkomponenten vom Ursprung zu einem standardisierten Punkt x_{std} ist der Betrag aus dem i-ten Score. Dabei ist der i-te Score das Skalarprodukt aus der i-ten Hauptkomponenten und dem normierten Datenpunkt:

Die **Scores** sind die Koordinaten der transformierten Datenpunkte in der neuen PCA-Basis. Für einen Datenpunkt $\mathbf{x}$ (standardisiert: $\mathbf{x}_{\text{std}}$) und die i-te Hauptkomponente $\mathbf{v}_i$ berechnet sich der Score als

$$\text{Score}_i = \mathbf{x}_{\text{std}}^T \mathbf{v}_i = \sum_{j=1}^{n} \frac{x_j - \mu_j}{\sigma_j} v_{ij}$$

Die Scores beschreiben die Position eines Datenpunktes entlang der Richtungen maximaler Varianz.

Scores sind die Koordinaten der Projektionen der standardisierten Daten auf die Hauptkomponenten.

Möchte man den Abstand eines Punktes vom Ursprung nach Projektion mit k-Hauptkomponenten ist dies

$$\sqrt{\sum_{i=1}^{k} (x_{\text{std}}^T \cdot v_i)^2} = \sqrt{\sum_{i=1}^{k} \text{Score}_i^2}.$$

D.3.1 Dimensionsreduktion

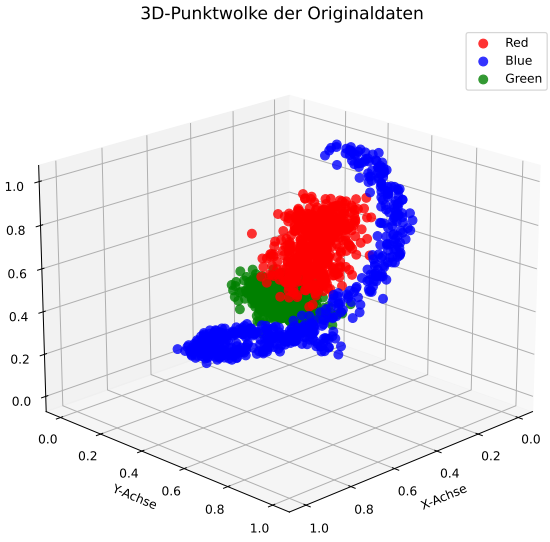
Für die Dimensionsreduktion werden nur die ersten k Hauptkomponenten beibehalten, wobei $k < n$ ist. Die Wahl von k erfolgt oft über die kumulative erklärte Varianz:

$$\text{Kumulative Varianz} = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{j=1}^{n} \lambda_j}$$

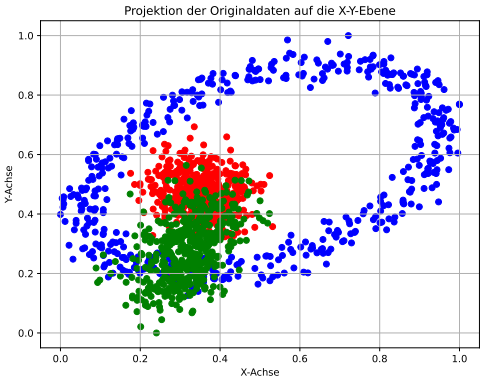
Ein häufig verwendeter Schwellenwert ist 95%, d.h. man wählt k so, dass mindestens 95% der ursprünglichen Varianz erhalten bleibt.

Die Wahl von 95% ist eine Faustregel aus der Statistik. In der Praxis variiert dieser Wert je nach Anwendung: Bildkompression oft 90%, Bioinformatik manchmal 99%, explorative Datenanalyse oft 80-90%.

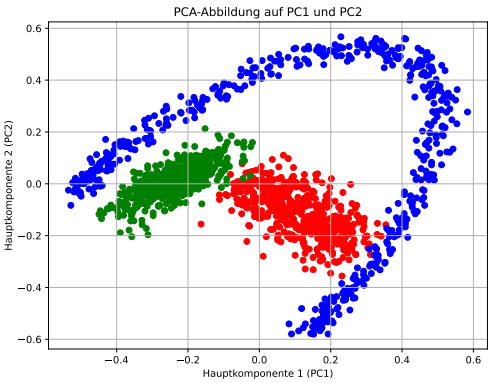
Durch die Beschränkung auf die ersten k Hauptkomponenten kann die Dimensionalität des Datensatzes erheblich verringert werden, was die Visualisierung, Analyse und Modellierung vereinfacht.



(a)



(b)



(c)

Abbildung D.1: Hauptkomponentenanalyse: (a) zeigt die ursprüngliche 3D-Punktwolke, (b) zeigt die "naive" Projektion auf die X-Y-Ebene durch Weglassen der Z-Koordinate (c) zeigt die Projektion auf der 1. und 2. Hauptkomponente - hier bleibt am meisten Information erhalten.

D.4 Zusammenfassung

Dieses Kapitel hat die Hauptkomponentenanalyse als eine der fundamentalsten Techniken der multivariaten Datenanalyse und des maschinellen Lernens vorgestellt. Ausgehend von den Grundlagen der linearen Algebra, insbesondere dem Konzept des Basiswechsels und der Eigenwertzerlegung, wurde gezeigt, wie PCA eine optimale Basis findet, die an die Datenstruktur angepasst ist. Ein Schlüsselaspekt ist die Transformation der zentrierten und standardisierten Daten in einen neuen Koordinatenraum, in dem die Hauptkomponenten orthogonal zueinander stehen und die Varianz in absteigender Reihenfolge erklären. Die mathematische Grundlage bildet die Eigenwertzerlegung der Kovarianzmatrix, wobei die Eigenvektoren die Richtungen maximaler Varianz und die Eigenwerte das Maß der erklärten Varianz darstellen.

Die praktischen Anwendungen der PCA erstrecken sich über verschiedene Bereiche: In der Datenvisualisierung ermöglicht sie die Darstellung hochdimensionaler Daten in zwei- oder dreidimensionalen Räumen, bei der Datenkompression wird durch Dimensionsreduktion bei Erhaltung der wichtigsten Information eine effiziente Speicherung erreicht, und bei der Rauschreduktion werden unwichtige Komponenten, die oft Rauschen repräsentieren, systematisch entfernt.

Das *Eigenfaces-Verfahren* war ein Meilenstein der Gesichtserkennung und eine der ersten erfolgreichen Anwendungen der PCA in der Bilderkennung ([51]).

Trotz ihrer Mächtigkeit unterliegt die PCA verschiedenen Annahmen und Einschränkungen, die ihre Anwendbarkeit begrenzen. Die Linearitätsannahme beschränkt PCA auf die Erfassung linearerZusammenhänge zwischen Variablen, während nichtlineare Strukturen nicht adäquat abgebildet werden können. Die Interpretation von Varianz als Informationsmaß kann problematisch sein, wenn wichtige Informationen in Richtungen geringer Varianz liegen. Darüber hinaus sind die resultierenden Hauptkomponenten oft schwer interpretierbar, da sie Linearkombinationen aller ursprünglichen Variablen darstellen. Schließlich können unterschiedliche Skalierungen der Variablen die Ergebnisse erheblich beeinflussen, weshalb eine sorgfältige Vorverarbeitung durch Standardisierung unerlässlich ist.

Klassische PCA-Mathoden setzen voraus, dass die Zusammenhänge linear sind. Es gibt Weiterentwicklungen - z.B. Kernel-PCA - die versuchen, diese Einschränkungen zu vermeiden. Diese sind aber sehr rechenintensiv.

Zusammenfassend lässt sich festhalten, dass die PCA ein unverzichtbares, mathematisch fundiertes Verfahren für jeden Datenanalysten ist, der hochdimensionale Datenstrukturen verstehen und die wesentlichen Informationen extrahieren möchte.

Verteilungen E

E.1 Normal-Verteilung

Die Normal-Verteilung ist die wichtigste und am häufigsten verwendete stetige Wahrscheinlichkeitsverteilung in der Statistik, die für alle reellen Zahlen definiert ist. Diese universelle Bedeutung verdankt sie dem zentralen Grenzwertsatz und ihrer Eigenschaft, natürliche Variationen in unzähligen Phänomenen zu beschreiben. Die Verteilung wird durch zwei Parameter charakterisiert: den Mittelwert μ (Lokationsparameter) und die Standardabweichung σ (Skalenparameter), die ihre Lage und Streuung bestimmen.

Auch als Gauß-Verteilung bekannt, obwohl sie bereits 1733 von Abraham de Moivre entdeckt wurde.

Die Wahrscheinlichkeitsdichtefunktion der **Normal-Verteilung** ist gegeben durch:

$$f(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad \text{für } -\infty < x < \infty$$

wobei $\mu \in \mathbb{R}$ der Mittelwert und $\sigma > 0$ die Standardabweichung ist.

Die charakteristische Glockenkurve entsteht durch die quadratische Funktion im Exponenten.

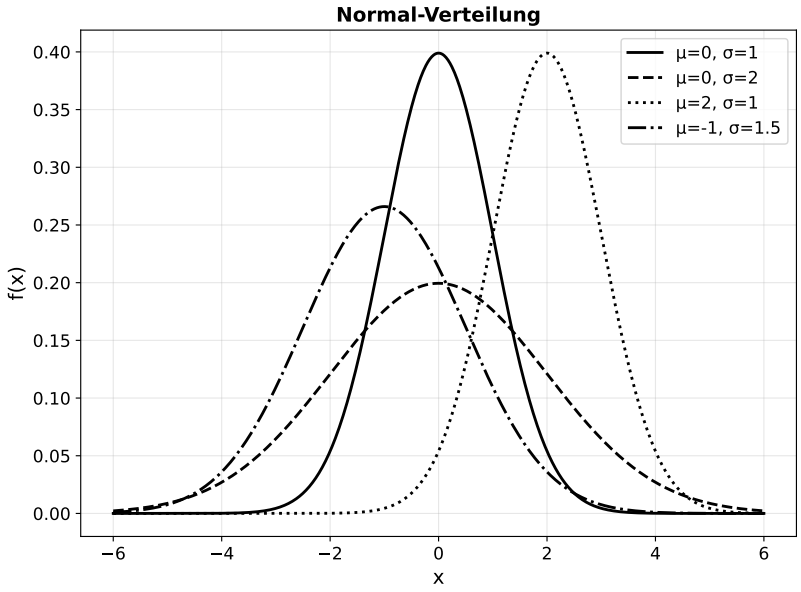


Abbildung E.1: Normal-Verteilung mit unterschiedlichen Parametern.

E.1.1 Statistische Eigenschaften

Die 68-95-99,7-Regel besagt, dass etwa 68% der Werte innerhalb einer Standardabweichung vom Mittelwert liegen.

Die Normal-Verteilung zeichnet sich durch ihre charakteristische Glockenkurve und fundamentale mathematische Eigenschaften aus. Die Verteilung ist perfekt symmetrisch um den Mittelwert μ, wodurch Median, Modus und Mittelwert identisch sind. Die Standardnormalverteilung mit $\mu = 0$ und $\sigma = 1$ dient als Referenz durch die Z-Transformation:

$$Z = \frac{X - \mu}{\sigma}.$$

Eine bemerkenswerte Eigenschaft ist die Stabilität unter linearen Transformationen:

$$\text{Ist} X \sim N(\mu, \sigma^2), \text{ dann ist } aX + b \sim N(a\mu + b, a^2\sigma^2).$$

Die Summe normalverteilter Zufallsvariablen ist ebenfalls normalverteilt. Der zentrale Grenzwertsatz erklärt, warum die Normalverteilung so häufig in der Natur auftritt: Die Summe vieler unabhängiger Zufallseinflüsse konvergiert gegen eine Normalverteilung.

Francis Galton demonstrierte dies mit seinem Galton-Brett, bei dem Kugeln durch zufällige Kollisionen eine Normalverteilung bilden.

E.1.2 Typische Anwendungsgebiete

Die Normal-Verteilung findet universelle Anwendung in nahezu allen quantitativen Wissenschaften. In den **Naturwissenschaften und Messtechnik** bildet sie die Grundlage für Fehlerrechnung und Messunsicherheit. Messfehler, experimentelle Abweichungen und natürliche Schwankungen folgen häufig einer Normalverteilung.

In der **Qualitätskontrolle und Fertigungstechnik** werden Normalverteilungen zur statistischen Prozesskontrolle verwendet. Fertigungstoleranzen und Qualitätsmerkmale werden typischerweise als normalverteilt angenommen. Das **Finanzwesen** modelliert Aktienrenditen und Zinssätze häufig als normalverteilt, obwohl dies nur näherungsweise zutrifft.

Die Finanzkrise 2008 verdeutlichte die Grenzen normalverteilungsbasierter Risikomodelle.

Psychologie und Sozialwissenschaften nutzen die Normalverteilung zur Modellierung menschlicher Eigenschaften wie Intelligenzquotienten und Persönlichkeitsdimensionen. In der **Medizin** werden biologische Parameter wie Körpergröße, Gewicht und Laborwerte analysiert. Das **Bildungswesen** modelliert Prüfungsergebnisse und standardisierte Testscores als normalverteilt.

E.2 Log-Normal-Verteilung

Die Log-Normal-Verteilung ist eine stetige Wahrscheinlichkeitsverteilung, die ausschließlich für positive Werte definiert ist und deren natürlicher Logarithmus einer Normalverteilung folgt. Diese fundamentale Beziehung zur Normalverteilung macht sie zur natürlichen Wahl für die Modellierung von Größen, die durch multiplikative Prozesse entstehen. Die Verteilung wird durch zwei Parameter charakterisiert: μ (Mittelwert des Logarithmus) und σ (Standardabweichung des Logarithmus), die ihre Form und statistischen Eigenschaften bestimmen.

Erstmals systematisch von Francis Galton 1879 zur Modellierung biologischer Größenvariationen beschrieben.

Die Wahrscheinlichkeitsdichtefunktion der **Log-Normal-Verteilung** ist gegeben durch:

$$f(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln x-\mu)^2}{2\sigma^2}} \quad \text{für } x > 0$$

wobei $\mu \in \mathbb{R}$ der Mittelwert und $\sigma > 0$ die Standardabweichung des zugrundeliegenden Logarithmus sind.

Wenn X log-normalverteilt ist, dann ist $\ln(X)$ normalverteilt mit Parametern μ und σ.

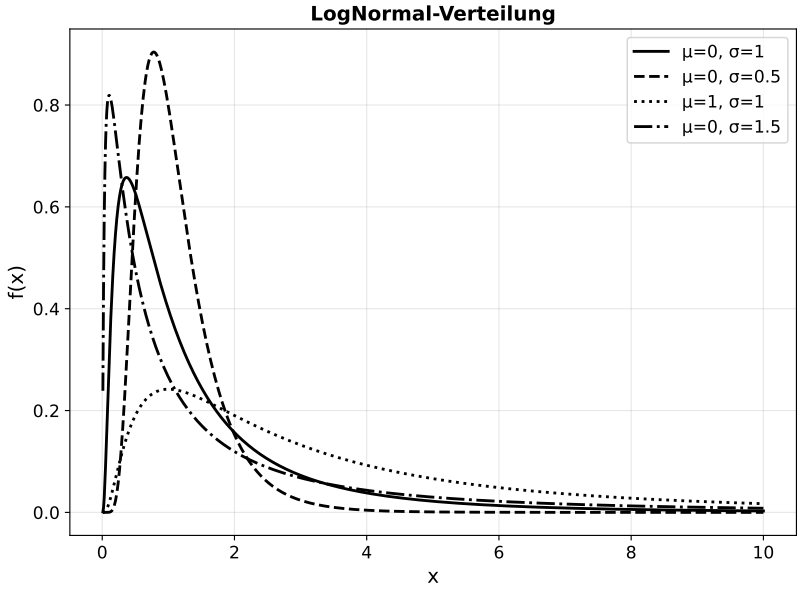


Abbildung E.2: Log-Normal-Verteilung mit unterschiedlichen Parametern.

E.2.1 Statistische Eigenschaften

Die Log-Normal-Verteilung zeichnet sich durch ihre charakteristische rechtsschiefe Form und ihre enge Verbindung zu multiplikativen Prozessen aus. Die Verteilung ist immer

rechtsschief, wobei die Schiefe mit zunehmendem σ stärker wird. Der Modus liegt bei $e^{\mu-\sigma^2}$, während der Median bei e^{μ} liegt. Eine bemerkenswerte Eigenschaft ist die multiplikative Stabilität: Das Produkt log-normalverteilter Variablen ist ebenfalls log-normalverteilt.

Die logarithmische Transformation $Y = \ln(X)$ überführt eine log-normalverteilte Variable in eine normalverteilte Variable, was die Anwendung aller normalverteilungsbasierten statistischen Methoden ermöglicht.

Der zentrale Grenzwertsatz in multiplikativer Form erklärt, warum das Produkt vieler positiver Zufallsvariablen gegen eine Log-Normal-Verteilung konvergiert.

E.2.2 Typische Anwendungsgebiete

Das **Finanzwesen und Kapitalmärkte** verwendet die Log-Normal-Verteilung als Grundlage für viele Bewertungsmodelle. Das Black-Scholes-Modell nimmt an, dass Aktienkurse einer geometrischen Brownschen Bewegung folgen, was zu log-normalverteilten Preisen führt. **Einkommen und Vermögen** folgen in vielen Ländern annähernd einer Log-Normal-Verteilung, bekannt als Gibrat-Gesetz.

Das *Gibrat-Gesetz* (Gesetz des proportionellen Wachstums, Robert Gibrat, 1931) besagt, dass die relative Wachstumsrate einer Größe unabhängig von ihrer aktuellen Größe ist.
Mathematisch gilt:

$$X_{t+1} = X_t \cdot \varepsilon_t, \quad \varepsilon_t > 0$$

wobei ε_t eine Zufallsvariable ist. Durch die wiederholte Multiplikation entsteht ein multiplikativer Prozess, dessen Logarithmus normalverteilt ist – somit ist X_t *lognormalverteilt*.

In der **Medizin und Pharmakokinetik** werden Arzneimittelkonzentrationen im Blut, Clearance-Raten und biologische Halbwertszeiten modelliert. **Umweltwissenschaften** verwenden Log-Normal-Verteilungen für Schadstoffkonzentrationen und Partikelgrößenverteilungen. **Materialwissenschaften** modellieren Partikelgrößenverteilungen und Oberflächenrauheiten. Das **Versicherungswesen** nutzt sie zur Modellierung von Schadenshöhen, und **Internet-Technologien** zeigen log-normale Muster bei Dateigrößen und Netzwerk-Traffic.

E.3 Exponential-Verteilung

Die Exponential-Verteilung ist eine fundamentale stetige Wahrscheinlichkeitsverteilung, die ausschließlich für nichtnegative Werte definiert ist und das Verhalten von Wartezeiten zwischen unabhängigen Ereignissen beschreibt. Diese Eigenschaft macht sie zur natürlichen Wahl für die Modellierung von Zwischenankunftszeiten, Ausfallzeiten und anderen zeitbasierten Prozessen mit konstanter Rate. Die Verteilung wird durch einen einzigen positiven Parameter λ (Ratenparameter) charakterisiert.

Entwickelt von Siméon Denis Poisson (1837) und später von Agner Krarup Erlang (1909) zur Modellierung von Telefonanrufen systematisiert.

Die Wahrscheinlichkeitsdichtefunktion der **Exponential-Verteilung**

ist gegeben durch:

$$f(x;\lambda) = \lambda e^{-\lambda x} \quad \text{für } x \geq 0$$

wobei $\lambda > 0$ der Ratenparameter ist.

Alternativ wird häufig die Parametrisierung mit dem Skalenparameter $\theta = 1/\lambda$ verwendet.

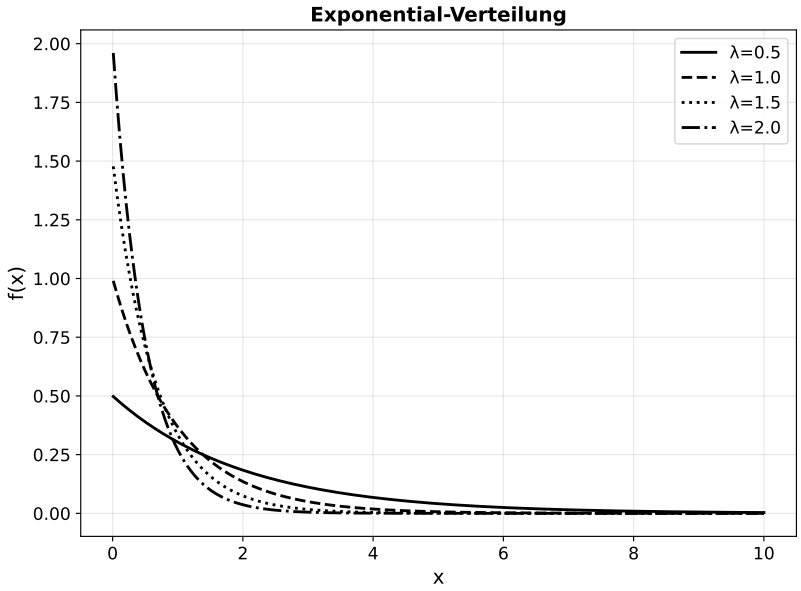


Abbildung E.3: Exponential-Verteilung mit unterschiedlichen Parametern.

E.3.1 Statistische Eigenschaften

Die Exponential-Verteilung zeichnet sich durch ihre einzigartige Gedächtnislosigkeit und ihre fundamentale Rolle in der Warteschlangentheorie aus. Die Gedächtnislosigkeit bedeutet: Die Wahrscheinlichkeit, dass ein Ereignis in den nächsten t Zeiteinheiten eintritt, ist unabhängig davon, wie lange bereits gewartet wurde. Diese Eigenschaft macht die Exponentialverteilung zur einzigen stetigen Verteilung mit konstanter Hazard-Rate.

Eine wichtige Beziehung besteht zur Poisson-Verteilung: Wenn die Anzahl der Ereignisse in einem festen Zeitintervall Poisson-verteilt ist, dann sind die Zwischenankunftszeiten exponentialverteilt. Das Minimum mehrerer unabhängiger exponentialverteilter Variablen ist wieder exponentialverteilt mit der Summe der Ratenparameter.

E.3.2 Typische Anwendungsgebiete

Das 'M' steht für Markovsch und impliziert Exponentialverteilung aufgrund der Gedächtnislosigkeit.

Warteschlangentheorie und Servicemanagement bilden das klassische Anwendungsgebiet der Exponentialverteilung. In M/M/1-Warteschlangen sind sowohl Zwischenankunfts- als auch Bedienzeiten exponentialverteilt. **Zuverlässigkeitstechnik** verwendet die Exponentialverteilung zur Modellierung der Nutzungsphase von Produktlebenszyklen, wo die Ausfallrate konstant ist.

Radioaktiver Zerfall und Kernphysik nutzen die Exponentialverteilung zur Beschreibung von Zerfallszeiten instabiler Atomkerne. **Kommunikationstechnik und Netzwerke** modellieren Paketankünfte in Computernetzwerken, Anrufzeiten in Telefonnetzen und Nachrichtenintervalle. Das **Finanzwesen und Risikomanagement** nutzt Exponentialverteilungen für Zeitintervalle zwischen extremen Marktereignissen, Kreditausfällen und Liquiditätskrisen. Die **Epidemiologie** verwendet exponentialverteilte Modelle für Inkubationszeiten bei Infektionskrankheiten und Intervalle zwischen Krankheitsausbrüchen.

E.4 Beta-Verteilung

Erstmals 1676 von Isaac Newton untersucht, aber erst 1839 von Eugène Charles Catalan systematisch beschrieben.

Die Beta-Verteilung ist eine äußerst flexible stetige Wahrscheinlichkeitsverteilung, die ausschließlich auf dem Intervall $[0, 1]$ definiert ist. Diese Eigenschaft macht sie zur natürlichen Wahl für die Modellierung von Wahrscheinlichkeiten, Anteilen, Proportionen und anderen normierten Größen. Die Verteilung wird durch zwei positive Parameter α und β charakterisiert, die ihre Form und statistischen Eigenschaften bestimmen.

Die Wahrscheinlichkeitsdichtefunktion der **Beta-Verteilung** ist gegeben durch:

$$f(x; \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)} \quad \text{für } 0 \leq x \leq 1$$

wobei $B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ die Beta-Funktion darstellt.

Die Beta-Funktion wurde von Leonhard Euler eingeführt.

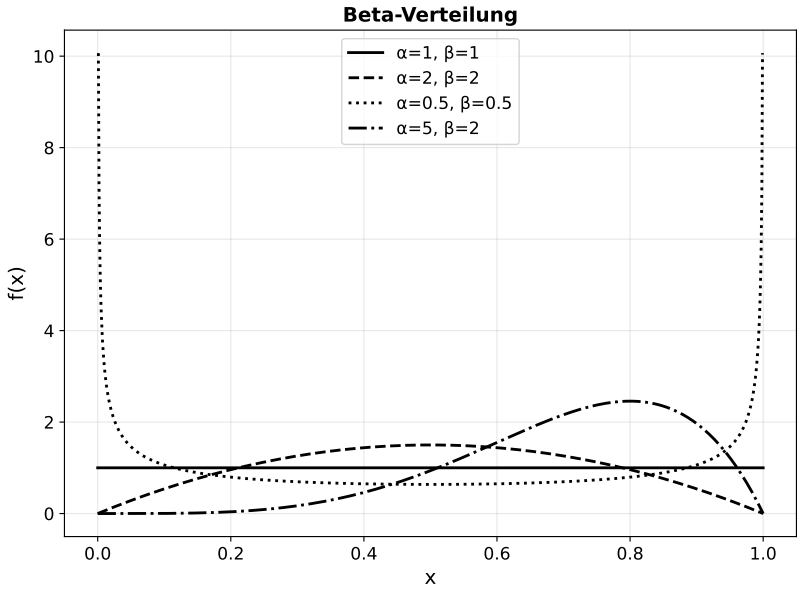


Abbildung E.4: Beta-Verteilung mit unterschiedlichen Parametern.

E.4.1 Statistische Eigenschaften

Die Beta-Verteilung zeichnet sich durch ihre außergewöhnliche Formflexibilität aus. Für Symmetrie gilt $\alpha = \beta$, wodurch die Verteilung symmetrisch um $x = 0{,}5$ wird. Bei Rechtsschiefe mit $\alpha < \beta$ entsteht eine rechtsschiefe Verteilung, bei Linksschiefe mit $\alpha > \beta$ eine linksschiefe Verteilung. Eine besondere U-Form entsteht für $\alpha, \beta < 1$. Im Spezialfall Uniform mit $\alpha = \beta = 1$ reduziert sich die Beta-Verteilung zur Uniformverteilung.

Diese Flexibilität in Form, Schiefe und Kurtosis ermöglicht es der Beta-Verteilung, eine breite Palette empirischer Verteilungen von Anteilsdaten zu approximieren.

E.4.2 Typische Anwendungsgebiete

Erfolgsraten und Wahrscheinlichkeiten bilden einen zentralen Anwendungsbereich der Beta-Verteilung. Sie dient zur Modellierung unbekannter Erfolgswahrscheinlichkeiten in der Bayesschen Statistik und fungiert als konjugierte Prior-Verteilung für die Binomialverteilung. **Marktanteile** verschiedener Unternehmen oder Produktkategorien folgen häufig Beta-Verteilungen, da sie natürlicherweise zwischen null und eins beschränkt sind.

Thomas Bayes verwendete bereits 1763 ähnliche Konzepte.

Qualitätskontrolle in der industriellen Produktion nutzt Beta-Verteilungen zur Modellierung von Ausschussraten und Fehlerquoten in IT-Systemen. Das **Finanzwesen** profitiert bei der Modellierung von Portfolioanteilen und Recovery-Raten bei Kreditausfällen. **Medizin und Biologie** verwenden Beta-Verteilungen zur Modellierung von Heilungsraten, normierten Überlebenswahrscheinlichkeiten und Genfrequenzen. **Projektmanagement** nutzt sie zur Darstellung von Fortschrittsgraden und Zielerreichungsgraden. **Sportwissenschaft** wendet Beta-Verteilungen auf Trefferquoten und Gewinnwahrscheinlichkeiten an.

E.5 Gamma-Verteilung

Die Gamma-Verteilung ist eine äußerst vielseitige stetige Wahrscheinlichkeitsverteilung, die ausschließlich für positive Werte definiert ist und eine natürliche Verallgemeinerung der Exponentialverteilung darstellt. Diese fundamentale Eigenschaft macht sie zur idealen Wahl für die Modellierung von Wartezeiten bis zum k-ten Ereignis, Lebensdauern mit variablen Ausfallraten und anderen positiven kontinuierlichen Größen. Die Verteilung wird durch zwei positive Parameter charakterisiert: den Formparameter α und den Ratenparameter β.

Benannt nach der Gamma-Funktion von Leonhard Euler (1729).

Die Wahrscheinlichkeitsdichtefunktion der **Gamma-Verteilung** ist gegeben durch:

$$f(x;\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \quad \text{für } x > 0$$

wobei $\alpha > 0$ der Formparameter, $\beta > 0$ der Ratenparameter und $\Gamma(\alpha)$ die Gamma-Funktion ist.

Alternative Parametrisierung: $f(x;\alpha,\theta) = \frac{1}{\Gamma(\alpha)\theta^\alpha} x^{\alpha-1} e^{-x/\theta}$ mit Skalenparameter $\theta = 1/\beta$.

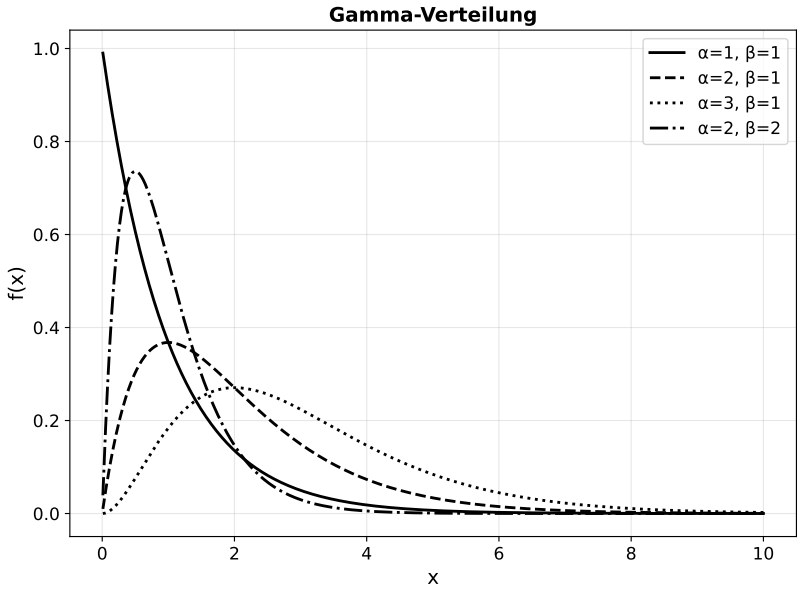


Abbildung E.5: Gamma-Verteilung mit unterschiedlichen Parametern.

E.5.1 Statistische Eigenschaften

Die Gamma-Verteilung zeichnet sich durch ihre außergewöhnliche Formflexibilität aus. Für $\alpha < 1$ entsteht eine monoton fallende Dichtefunktion mit einer Singularität bei $x = 0$. Bei $\alpha = 1$ reduziert sich die Verteilung zur Exponentialverteilung mit konstanter Hazard-Rate. Für $\alpha > 1$ ergibt sich eine eingipflige Verteilung mit einem Maximum bei $(\alpha - 1)/\beta$. Bei großen Werten von α nähert sich die Verteilung der Normalverteilung an.

Eine fundamentale Eigenschaft ist die Additivität: Die Summe unabhängiger Gamma-verteilter Variablen mit gleichem Ratenparameter ist wieder Gamma-verteilt mit der Summe der Formparameter. Die Skalierungseigenschaft führt dazu, dass cX für Gamma-verteiltes X und $c > 0$ wieder Gamma-verteilt ist mit modifiziertem Ratenparameter.

E.5.2 Typische Anwendungsgebiete

Warteschlangentheorie und Prozessmodellierung verwenden die Gamma-Verteilung zur Modellierung der Wartezeit bis zum k-ten Ereignis in Poisson-Prozessen. **Zuverlässigkeitstechnik mit variablen Ausfallraten** nutzt die Gamma-Verteilung zur Modellierung von Lebensdauern, wenn die

Ausfallrate nicht konstant ist. Im Gegensatz zur Exponentialverteilung kann die Gamma-Verteilung sowohl steigende als auch fallende Ausfallraten beschreiben.

Bayessche Statistik und Inferenz verwendet die Gamma-Verteilung extensiv als konjugierte Prior-Verteilung für Präzisionsparameter und Ratenparameter. Das **Finanzwesen und Risikomanagement** modelliert aggregierte Verluste, Zeitintervalle zwischen Marktschocks und Volatilitätsprozesse. **Meteorologie und Klimawissenschaften** verwenden Gamma-Verteilungen zur Modellierung von Niederschlagsmengen, da diese nur positive Werte annehmen können und häufig rechtsschiefe Verteilungen zeigen. **Medizinische Statistik und Epidemiologie** nutzen Gamma-Verteilungen zur Modellierung von Überlebenszeiten, wenn die Hazard-Rate nicht konstant ist.

E.6 Gumbel-Verteilung

Die Gumbel-Verteilung ist eine fundamentale stetige Wahrscheinlichkeitsverteilung aus der Familie der Extremwertverteilungen, die für alle reellen Zahlen definiert ist und das asymptotische Verhalten von Maxima unabhängiger, identisch verteilter Zufallsvariablen beschreibt. Diese fundamentale Eigenschaft macht sie zur natürlichen Wahl für die Modellierung extremer Ereignisse wie Naturkatastrophen, Rekordwerte und seltene Grenzfälle. Die Verteilung wird durch zwei Parameter charakterisiert: den Lokationsparameter μ und den Skalenparameter β.

Benannt nach dem deutsch-amerikanischen Mathematiker Emil Julius Gumbel (1891-1966), der sie zur Analyse von Hochwasserdaten entwickelte.

Die Wahrscheinlichkeitsdichtefunktion der **Gumbel-Verteilung** ist gegeben durch:

$$f(x;\mu,\beta) = \frac{1}{\beta} e^{-\frac{x-\mu}{\beta}} e^{-e^{-\frac{x-\mu}{\beta}}} \quad \text{für } -\infty < x < \infty$$

wobei $\mu \in \mathbb{R}$ der Lokationsparameter und $\beta > 0$ der Skalenparameter ist.

Die charakteristische Doppel-Exponential-Form verleiht der Verteilung ihre asymmetrischen Eigenschaften.

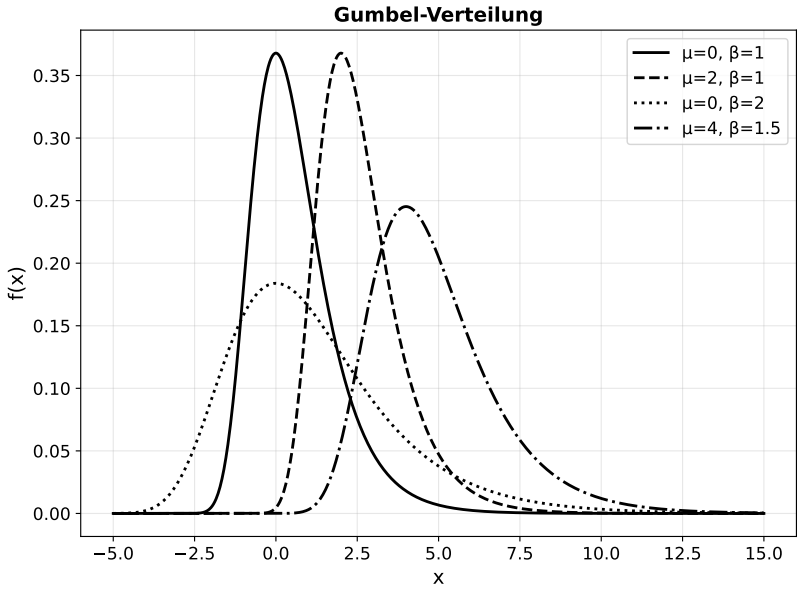


Abbildung E.6: Gumbel-Verteilung mit unterschiedlichen Parametern.

E.6.1 Statistische Eigenschaften

Die Gumbel-Verteilung zeichnet sich durch ihre charakteristische asymmetrische Form und ihre fundamentale Rolle in der Extremwerttheorie aus. Die Verteilung ist rechtsschief mit einem langen rechten Schwanz, der extreme Werte wahrscheinlicher macht als bei einer symmetrischen Verteilung. Der Modus liegt bei μ, während der Median bei $\mu - \beta \ln(\ln(2)) \approx \mu - 0{,}367\beta$ liegt. Die charakteristische Euler-Mascheroni-Konstante $\gamma \approx 0{,}5772$ erscheint im Erwartungswert: $E[X] = \mu + \beta\gamma$.

Eine fundamentale Eigenschaft ist die Stabilität unter Maximierung: Das Maximum von Gumbel-verteilten Variablen ist wieder Gumbel-verteilt mit modifizierten Parametern. Eine wichtige Beziehung besteht zur Exponentialverteilung: Wenn Y exponentialverteilt ist, dann ist $X = \mu - \beta \ln(Y)$ Gumbel-verteilt.

E.6.2 Typische Anwendungsgebiete

Hydrologie und Wasserwirtschaft verwenden die Gumbel-Verteilung als Standardwerkzeug für die Hochwasseranalyse und die Bestimmung von Bemessungshochwässern. Jährliche Höchstwasserstände, maximale Niederschlagsmengen und extreme Abflussspitzen folgen häufig Gumbel-Verteilungen.

Das niederländische Rijkswaterstaat nutzt Gumbel-Analysen zur Dimensionierung der Deltawerke.

Strukturtechnik und Bauwesen nutzen Gumbel-Verteilungen zur Modellierung extremer Lasten wie Windgeschwindigkeiten, Schneelasten und seismischen Beschleunigungen.

Klimatologie und Meteorologie modellieren extreme Temperaturen und Hitzewellen mit Gumbel-Verteilungen. **Finanzwesen und Risikomanagement** verwenden Gumbel-Verteilungen zur Modellierung extremer Marktverluste, Tail-Risiken und systemischer Finanzrisiken. **Versicherungswesen** nutzt Gumbel-Verteilungen für die Bewertung von Katastrophenrisiken, maximalen jährlichen Schadenshöhen und Rückversicherungsberechnungen. **Umweltwissenschaften und Ökologie** verwenden Gumbel-Verteilungen zur Analyse von Schadstoffspitzenwerten, extremen Umweltbelastungen und ökologischen Stressereignissen. **Seismologie und Erdbebentechnik** nutzen Gumbel-Verteilungen zur Modellierung maximaler jährlicher Erdbebenmagnitudes und seismischer Beschleunigungen.

E.7 Weibull-Verteilung

Benannt nach dem schwedischen Mathematiker Waloddi Weibull, der sie 1951 zur Modellierung von Materialermüdung einführte.

Die Weibull-Verteilung ist eine vielseitige stetige Wahrscheinlichkeitsverteilung, die ausschließlich für nicht-negative Werte definiert ist. Diese Eigenschaft macht sie zur bevorzugten Wahl für die Modellierung von Lebensdauern, Ausfallzeiten und anderen positiven kontinuierlichen Größen. Die Verteilung wird durch zwei positive Parameter charakterisiert: den Formparameter k und den Skalenparameter λ, die ihre Form und statistischen Eigenschaften bestimmen.

Die Weibull-Verteilung ist eng verwandt mit der Exponentialverteilung als Spezialfall.

Die Wahrscheinlichkeitsdichtefunktion der **Weibull-Verteilung** ist gegeben durch:

$$f(x;k,\lambda) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k} \quad \text{für } x \geq 0$$

wobei $k > 0$ der Formparameter und $\lambda > 0$ der Skalenparameter ist.

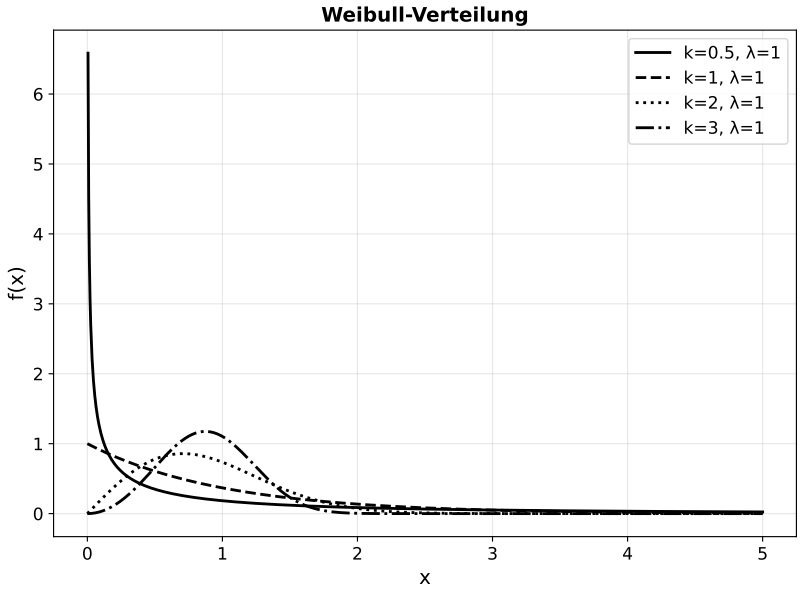


Abbildung E.7: Weibull-Verteilung mit unterschiedlichen Parametern.

E.7.1 Statistische Eigenschaften

Die Weibull-Verteilung zeichnet sich durch ihre außergewöhnliche Formflexibilität aus. Für $k < 1$ entsteht eine monoton fallende Dichtefunktion mit einer abnehmenden Ausfallrate, die typisch für Frühausfälle oder Kinderkrankheiten ist. Bei $k = 1$ reduziert sich die Verteilung zur Exponentialverteilung mit konstanter Ausfallrate, dem charakteristischen Merkmal zufälliger Ausfälle. Für $k > 1$ ergibt sich eine eingipflige Verteilung mit steigender Ausfallrate, die Verschleißausfälle oder Alterungsprozesse beschreibt. Der Spezialfall $k = 2$ entspricht der Rayleigh-Verteilung , während $k = 3{,}6$ eine gute Approximation der Normalverteilung darstellt.

Die Rayleigh-Verteilung ist die Verteilung der Beträge zweidimensionaler Vektoren mit unabhängigen normalverteilten Komponenten.

Diese Flexibilität in der Modellierung verschiedener Ausfallmuster macht die Weibull-Verteilung zu einem Standardwerkzeug in der Zuverlässigkeitsanalyse und Lebensdauermodellierung.

E.7.2 Typische Anwendungsgebiete

Zuverlässigkeitstechnik und Qualitätskontrolle bilden das Herzstück der Weibull-Anwendungen. Sie dient zur Modellierung der Lebensdauer von Maschinen, elektronischen Bauteilen, Motoren und anderen technischen Systemen. Die Verteilung ermöglicht es, verschiedene Ausfallmechanismen

Die *NASA* nutzt die Weibull-Analyse, um die Zuverlässigkeit und Lebensdauer von Komponenten in Gasturbinen zu bewerten, z. B. zur Analyse von Schaufelbruch und Ermüdungsversagen ([52]).

zu charakterisieren: Frühausfälle durch Fertigungsfehler, zufällige Ausfälle durch externe Einflüsse und Verschleißausfälle durch Alterung.

Die Weibull-Verteilung ist nützlich wegen ihrer Flexibilität zur Beschreibung unterschiedlicher *Ausfallraten* (steigend, konstant, fallend).

Materialwissenschaften verwenden die Weibull-Verteilung extensiv zur Charakterisierung der Bruchfestigkeit von spröden Materialien wie Keramik, Glas und Verbundwerkstoffen. **Windenergie** nutzt die Weibull-Verteilung zur Modellierung von Windgeschwindigkeitsverteilungen an potenziellen Standorten für Windkraftanlagen. **Medizinische Forschung** wendet Weibull-Modelle in der Überlebensanalyse und Epidemiologie an. Das **Versicherungswesen** nutzt Weibull-Verteilungen zur Modellierung von Schadenszeitpunkten und Lebensdauern in der Lebensversicherung. **Hydrologie und Klimatologie** verwenden die Weibull-Verteilung zur Analyse extremer Wetterereignisse wie Hochwasser, Dürreperioden oder Sturmereignissen. **Wirtschaftswissenschaften** nutzen Weibull-Modelle zur Analyse von Unternehmensinsolvenzen, Produktlebenszyklen und Kundenabwanderungsraten.

Literaturverzeichnis

[1] Arthur G. Stephenson u. a. „Lessons learned from the loss of the Mars Climate Orbiter“. In: *IEEE Aerospace and Electronic Systems Magazine* 19.3 (2004), S. 13–18. DOI: 10.1109/MAES.2004.1281733 (siehe S. 10).

[2] Thomas C. Redman. „Seizing Opportunity in Data Quality“. In: *MIT Sloan Management Review* (2016). Study based on research by Experian plc and consultants James Price and Martin Spratt (siehe S. 10).

[3] Alon Halevy, Peter Norvig und Fernando Pereira. „The Unreasonable Effectiveness of Data“. In: *IEEE Intelligent Systems* 24.2 (2009). Google Research Essay, S. 8–12. DOI: 10.1109/MIS.2009.36 (siehe S. 14).

[4] Julia Angwin u. a. *Machine Bias: Risk Assessments in Criminal Sentencing*. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing. ProPublica-Analyse von COMPAS. 2016 (siehe S. 15).

[5] U.S. Securities and Exchange Commission und Commodity Futures Trading Commission. *Findings Regarding the Market Events of May 6, 2010*. Techn. Ber. Joint SEC and CFTC report on the “Flash Crash” of May 6, 2010. U.S. Securities, Exchange Commission (SEC) und Commodity Futures Trading Commission (CFTC), Sep. 2010 (siehe S. 21).

[6] K. A. Mohammed u. a. „The Effects of Data Quality on Machine Learning Performance“. In: *arXiv preprint* (2022). Available online via arXiv:2207.14529 (siehe S. 27).

[7] Forrester Research. *Benefits To Organizations With Advanced Data Literacy Levels*. Techn. Ber. Data Culture & Literacy Survey 2023. Forrester Research, 2023 (siehe S. 40).

[8] IBM. *The Impact of Bad Data + How Observability is the Solution*. Blog Post. IBM, Apr. 2025. URL: https://www.ibm.com/think/insights/observability-data-benefits (besucht am 07. 09. 2025) (siehe S. 71).

[9] Svetlana Ulianova. *Cardiovascular Disease dataset*. Accessed: 2025-08-28. 2019. URL: https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset (siehe S. 90, 159, 194, 196, 225, 232).

[10] Pragmatic Institute. „Overcoming the 80/20 Rule in Data Science“. In: *Pragmatic Institute Resources* (Feb. 2024). Accessed: 3. November 2025 (siehe S. 99).

[11] John W. Tukey. *Exploratory Data Analysis*. Reading, MA: Addison-Wesley, 1977, S. 688 (siehe S. 101, 245).

[12] Kes Ward. *An Introduction to Robust Statistics*. Blog post, STOR-i Student Sites, Lancaster University. Accessed: 3. November 2025. Juni 2020. URL: https://www.lancaster.ac.uk/stor-i-student-sites/kim-ward/2020/06/30/an-introduction-to-robust-statistics/ (siehe S. 102).

[13] Boris Iglewicz und David Hoaglin. *How to Detect and Handle Outliers*. Hrsg. von Edward F. Mykytka. Bd. 16. The ASQC Basic References in Quality Control: Statistical Techniques. American Society for Quality Control. ASQC Quality Press, 1993 (siehe S. 103).

[14] Hans R. Künsch. „Obituary: Frank Hampel, 1941–2018“. In: *IMS Bulletin* (Dez. 2018). Accessed: 3. November 2025 (siehe S. 103).

[15] Frank R. Hampel u. a. *Robust Statistics: The Approach Based on Influence Functions*. Wiley Series in Probability and Statistics. New York: John Wiley & Sons, 1986, S. 502 (siehe S. 104, 115).

[16] Frank R. Hampel u. a. *Robust Statistics: The Approach Based on Influence Functions*. ResearchGate. PDF available on ResearchGate. 1986. URL: https://www.researchgate.net/publication/346630632_Robust_Statistics_The_Approach_Based_on_Influence_Functions (besucht am 15. 09. 2025) (siehe S. 104, 115).

[17] Max Kuhn und Kjell Johnson. *Feature Engineering and Selection: A Practical Approach for Predictive Models*. Chapman & Hall/CRC Data Science Series. Boca Raton, FL: Taylor & Francis, 2019 (siehe S. 105, 109).

[18] Paulo Cortez u. a. *Wine Quality*. Donated on 10/6/2009. UCI Machine Learning Repository, 2009. DOI: 10.24432/C56S3T (siehe S. 106, 118, 119, 125).

[19] P. Cortez u. a. „Modeling wine preferences by data mining from physicochemical properties“. In: *Decis. Support Syst.* 47 (2009), S. 547–553 (siehe S. 106).

[20] Max Kuhn und Kjell Johnson. *Feature Engineering and Selection: A Practical Approach for Predictive Models*. Online version. 2019. URL: https://bookdown.org/max/FES/ (besucht am 14. 09. 2025) (siehe S. 109).

[21] J. C. Farman, B. G. Gardiner und J. D. Shanklin. „Large losses of total ozone in Antarctica reveal seasonal ClO_x/NO_x interaction“. In: *Nature* 315.6016 (1985). Das Paper, das die Entdeckung des Ozonlochs publizierte, S. 207–210. DOI: 10.1038/315207a0 (siehe S. 111).

[22] Gavin. *What did NASA know? and when did they know it?* https://www.realclimate.org/index.php/archives/2017/12/what-did-nasa-know-and-when-did-they-know-it/. Zugriff am 3. November 2025. Dez. 2017 (siehe S. 111).

[23] NASA. *Lesson 641: Mars Climate Orbiter – An Imperial / Metric Units Mishap*. https://llis.nasa.gov/lesson/641. Zugriff am 3. November 2025. 2016 (siehe S. 112).

[24] European Commission. *Authorised Economic Operator Guidelines*. Revision 6. 2016. URL: https://taxation-customs.ec.europa.eu/system/files/2017-03/aeo_guidelines_en.pdf (besucht am 15. 09. 2025) (siehe S. 115).

[25] Kenneth P. Burnham und David R. Anderson. *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. 2. Aufl. New York: Springer, 2002, S. 488 (siehe S. 121).

[26] wordsforthewise. *Lending Club Loan Data*. Complete dataset with approximately 30 million credit decisions. 2024. URL: https://www.kaggle.com/datasets/wordsforthewise/lending-club (besucht am 07. 09. 2025) (siehe S. 134).

[27] IEEE DataPort. *Time Series Dataset for DDoS (TSD-DDoS) Attack Detection*. https://ieee-dataport.org/documents/time-series-dataset-ddostsd-ddos-attack-detection. Dataset for time series-based detection of TCP-based flooding attacks. 2024 (siehe S. 138).

[28] Mark J. Nigrini. *Forensic Analytics: Methods and Techniques for Forensic Accounting Investigations*. 1st. John Wiley & Sons, 2011, S. 480 (siehe S. 145).

[29] Association of Certified Fraud Examiners. *Fraud Examiners Manual*. 2022 Edition. Austin, TX: Association of Certified Fraud Examiners, 2022 (siehe S. 152).

[30] American Institute of Certified Public Accountants, Business Valuation and Forensic & Litigation Services Section. *Forensic Accounting – Fraud Investigations*. Practice Aid 07-1. New York, NY: American Institute of Certified Public Accountants, 2007 (siehe S. 152).

[31] *Anti-bribery management systems – Requirements with guidance for use*. Updated 2025. Geneva, Switzerland, 2016 (siehe S. 152).

[32] Teuvo Kohonen. *Self-Organizing Maps*. 3rd. Bd. 30. Springer Series in Information Sciences. Berlin, Heidelberg: Springer, 2001 (siehe S. 187).

[33] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. 3rd. Upper Saddle River, NJ: Prentice Hall, 2008 (siehe S. 189).

[34] Guido Deboeck und Teuvo Kohonen, Hrsg. *Visual Explorations in Finance with Self-Organizing Maps*. London: Springer, 1998 (siehe S. 205).

[35] Alfred Ultsch. „Self-organizing neural networks for visualization and classification“. In: *Information and Classification: Concepts, Methods and Applications*. Hrsg. von Otto Opitz, Berthold Lausen und Rüdiger Klar. Berlin, Heidelberg: Springer, 1993, S. 307–313 (siehe S. 210).

[36] Markus M. Breunig u. a. „LOF: Identifying Density-Based Local Outliers“. In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. ACM. Dallas, TX, USA, 2000, S. 93–104. DOI: 10.1145/342009.335388 (siehe S. 215, 220).

[37] Charu C. Aggarwal. *Outlier Analysis*. 2nd. Cham, Switzerland: Springer, 2017 (siehe S. 215).

[38] Fabian Pedregosa u. a. „Scikit-learn: Machine learning in Python“. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830 (siehe S. 220).

[39] John C. Gower. „A general coefficient of similarity and some of its properties“. In: *Biometrics* 27.4 (1971), S. 857–871. DOI: 10.2307/2528823 (siehe S. 221).

[40] Fei Tony Liu, Kai Ming Ting und Zhi-Hua Zhou. „Isolation Forest“. In: *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM)*. Piscataway, NJ: IEEE, 2008, S. 413–422. DOI: 10.1109/ICDM.2008.17 (siehe S. 227).

[41] Fei Tony Liu, Kai Ming Ting und Zhi-Hua Zhou. „Isolation-based Anomaly Detection“. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 6.1 (2012), 3:1–3:39. DOI: 10.1145/2133360.2133363 (siehe S. 227, 228, 230, 233).

[42] Ludwig Fahrmeir u. a. *Statistik: Der Weg zur Datenanalyse*. 9., überarb. u. erg. Auflage. Umfassende Darstellung der deskriptiven und induktiven Statistik sowie explorativen Datenanalyse. Berlin, Heidelberg: Springer, 2023 (siehe S. 242, 247).

[43] Justin Matejka und George Fitzmaurice. „Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing“. In: *CHI '17: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (2017). Introduces the Datasaurus Dozen - datasets with identical statistical properties but different visual patterns, S. 1290–1294. DOI: 10.1145/3025453.3025912 (siehe S. 244).

[44] *Shannon's "A Mathematical Theory of Communication"*. https://www.historyofinformation.com/detail.php?entryid=860. History of Information. 1948. (Besucht am 14. 09. 2025) (siehe S. 245).

[45] Prem S. Mann. *Introductory Statistics*. 9th edition. Einführung in die angewandte Statistik für Studenten ohne starken mathematischen Hintergrund. Wiley, 2019 (siehe S. 247).

[46] Gerald Keller und Brian Warrack. *Statistics for Management and Economics*. 11th edition. Anwendungsorientierte Statistik für Wirtschafts- und Managementbereiche. Cengage Learning, 2019 (siehe S. 247).

[47] Lothar Sachs. *Angewandte Statistik: Anwendung statistischer Methoden*. 11., überarb. u. aktualisierte Auflage. Lehrbuch und Nachschlagewerk für anwendungsorientierte Leser. Berlin, Heidelberg: Springer-Verlag, 2004, S. LXXVI, 889 (siehe S. 247).

[48] Horst Rinne. *Taschenbuch der Statistik*. 3., überarb. und erw. Auflage. Nachschlagewerk und kommentierte Formelsammlung für Wirtschafts- und Sozialwissenschaften. Frankfurt am Main: Harri Deutsch, 2003, S. 612 (siehe S. 247).

[49] Leonhard Held. *Methoden der statistischen Inferenz: Likelihood und Bayes*. Heidelberg: Spektrum Akademischer Verlag, 2008, S. XII, 304 (siehe S. 249).

[50] Ian T. Jolliffe. *Principal Component Analysis*. 2nd. Springer Series in Statistics. New York: Springer, 2002 (siehe S. 269).

[51] Matthew A. Turk und Alex P. Pentland. „Eigenfaces for Recognition". In: *Journal of Cognitive Neuroscience* 3.1 (1991). Original presentation: "Face Recognition Using Eigenfaces", CVPR 1991, S. 71–86. DOI: 10.1162/jocn.1991.3.1.71 (siehe S. 275).

[52] Stephen G. Johnson und John B. Cushing. *Reliability Analysis of Gas Turbine Engine Components Using Weibull Statistics*. NASA Technical Memorandum NASA/TM-2013-217851. NASA NTRS 20130013703. NASA Glenn Research Center, 2013 (siehe S. 289).

Alphabetical Index

Prof. Dr. Andreas Igl

Josef Gruber

In einer Welt, in der Künstliche Intelligenz und Large Language Models die Zukunft prägen, bleibt eines unverändert entscheidend: die Qualität der Daten.

Dieses Handbuch zeigt praxisnah, wie Datenqualität gemessen, gesichert und verbessert werden kann – in Datenbanken ebenso wie in alltäglichen Dokumenten. Als eines der weltweit ersten Werke stellt es konkrete, sofort einsetzbare Werkzeuge in Form von Prompts vor. Diese Prompts können online bezogen und auf bereitgestellten Datensätzen direkt ausprobiert werden. So wird Datenqualitätsmanagement greifbar, interaktiv und KI-gestützt.

Ein unverzichtbarer Leitfaden für alle, die Daten nicht nur nutzen, sondern wirklich verstehen und verbessern wollen.

BRAINBOARD

Das Buch ist im Rahmen der Zusammenarbeit mit BRAINBOARD entstanden – einer Initiative zur Förderung datengetriebener Exzellenz und innovativer KI-Anwendungen. Weitere Informationen und ergänzende Materialien finden Sie unter www.brainboard.de

9 783000 848322