

Computational Intelligence

LEHRBUCH

Thomas A. Runkler

# Data Mining

Modelle und Algorithmen intelligenter  
Datenanalyse

*2. Auflage*



Springer Vieweg

---

# **Computational Intelligence**

## **Herausgegeben von**

Prof. Dr. Wolfgang Bibel, Scheidegg

Prof. Dr. Rudolf Kruse, Otto-von-Guericke-Universität Magdeburg

Prof. Dr. Bernhard Nebel, Albert-Ludwigs-Universität Freiburg

Die Reihe „Computational Intelligence“ wird herausgegeben von den Professoren Wolfgang Bibel, Rudolf Kruse und Bernhard Nebel.

Weitere Bände siehe <http://www.springer.com/series/12572>

---

Thomas A. Runkler

# Data Mining

Modelle und Algorithmen intelligenter  
Datenanalyse

2., aktualisierte Auflage



Springer Vieweg

Thomas A. Runkler  
München  
Deutschland

Computational Intelligence

ISBN 978-3-8348-1694-8

ISBN 978-3-8348-2171-3 (eBook)

DOI 10.1007/978-3-8348-2171-3

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Springer VS

© Springer Fachmedien Wiesbaden 2010, 2015

Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürfen. Der Verlag, die Autoren und die Herausgeber gehen davon aus, dass die Angaben und Informationen in diesem Werk zum Zeitpunkt der Veröffentlichung vollständig und korrekt sind. Weder der Verlag noch die Autoren oder die Herausgeber übernehmen, ausdrücklich oder implizit, Gewähr für den Inhalt des Werkes, etwaige Fehler oder Äußerungen.

Gedruckt auf säurefreiem und chlorfrei gebleichtem Papier

Springer Fachmedien Wiesbaden ist Teil der Fachverlagsgruppe Springer Science+Business Media  
([www.springer.com](http://www.springer.com))

---

## Vorwort

Die Information in der Welt verdoppelt sich etwa alle 20 Monate. Zu den wichtigsten Datenquellen gehören geschäftliche und industrielle Prozesse, Texte und strukturierte Datenbanken, Bild- und Videodaten, sowie Daten aus Physik, Biologie oder Medizin. Ziel der Datenanalyse ist es, solche Daten zu nutzen, um relevante Informationen zu finden, verfügbare Informationen zu strukturieren, neue Erkenntnisse zu gewinnen, Ursachen und Wirkungen zu erkennen, zukünftige Entwicklungen vorherzusagen oder optimale Entscheidungen zu treffen. Zur Sammlung, Vorverarbeitung, Analyse und Nutzung von Daten werden viele unterschiedliche Modelle und Algorithmen aus den Bereichen Statistik, maschinelles Lernen, Mustererkennung, Systemtheorie oder künstliche Intelligenz eingesetzt. Die Nutzung wertvoller Daten mit solchen Datenanalysemethoden wird als „Data Mining“ bezeichnet.

Dieses Buch gibt eine strukturierte Einführung in die wichtigsten Methoden und Algorithmen des Data Mining. Es vermittelt die wichtigsten Konzepte moderner Datenanalyse und befähigt die Leserin oder den Leser, für konkrete Anwendungen geeignete Data-Mining-Methoden auszuwählen und erfolgreich in eigenen Projekten einzusetzen. Das Buch richtet sich an Ingenieure, Informatiker und Mathematiker in Forschung und Lehre, an Studierende dieser Fachgebiete, aber auch an Praktiker, die mit großen Datensätzen arbeiten. Die Gliederung orientiert sich am typischen schrittweisen Ablauf von Datenanalyse-Projekten. Zum Verständnis werden lediglich grundlegende Mathematiken vorausgesetzt. Der Stoff dieses Buches basiert auf der langjährigen Vorlesung „Data Mining and Knowledge Discovery“ an der Fakultät für Informatik der Technischen Universität München. Das Buch enthält Ergebnisse aus zahlreichen Forschungs- und Entwicklungsprojekten bei Siemens Corporate Technology in München. Die erste Fassung dieses Buches erschien im Jahr 2000 unter dem Titel „Information Mining“. Eine erste Aktualisierung und umfassende Erweiterung erschien 2010 unter dem Titel „Data Mining“. Eine englische Fassung mit dem Titel „Data Analytics“ erschien 2012. Das vorliegende Buch ist als Übersetzung und Überarbeitung der englischen Fassung von 2012 entstanden.

Ich danke allen, die zu diesem Buch und den vorangegangenen Fassungen beigetragen haben, besonders auch den Studierenden, Rezessenten und Lesern für die vielen Hinweise auf Fehler, Widersprüche und Verbesserungsvorschläge, die in dieses Buch eingeflossen sind, und nicht zuletzt meinen Partnerinnen und Partnern bei Springer Vieweg für die unkomplizierte und konstruktive Zusammenarbeit.

München, Juli 2015

Thomas A. Runkler

---

# Inhaltsverzeichnis

<b>1</b>	<b>Einführung</b>	1
1.1	Daten	1
1.2	Data Mining, Data Analytics und Knowledge Discovery	2
Literatur		3
<b>2</b>	<b>Daten und Relationen</b>	5
2.1	Der Iris-Datensatz	5
2.2	Maßskalen	6
2.3	Mengen- und Matrixdarstellung	10
2.4	Relationen	11
2.5	Unähnlichkeitsmaße	12
2.6	Ähnlichkeitsmaße	14
2.7	Sequenzrelationen	16
2.8	Abtastung und Quantisierung	18
Literatur		22
<b>3</b>	<b>Datenvorverarbeitung</b>	23
3.1	Fehlerarten	23
3.2	Behandlung fehlerhafter Daten	26
3.3	Filterung	27
3.4	Datentransformationen	32
3.5	Datenintegration	35
Literatur		36
<b>4</b>	<b>Datenvisualisierung</b>	37
4.1	Diagramme	37
4.2	Hauptkomponentenanalyse	39
4.3	Mehrdimensionale Skalierung	43
4.4	Sammon-Abbildung	47
4.5	Auto-Assoziator	51
4.6	Histogramme	51

4.7	Spektralanalyse .....	54
	Literatur .....	58
<b>5</b>	<b>Korrelation</b> .....	59
5.1	Lineare Korrelation .....	59
5.2	Korrelation und Kausalität .....	61
5.3	Chi-Quadrat-Test auf Unabhängigkeit .....	63
	Literatur .....	65
<b>6</b>	<b>Regression</b> .....	67
6.1	Lineare Regression .....	67
6.2	Lineare Regression mit nichtlinearer Substitution .....	71
6.3	Robuste Regression .....	72
6.4	Neuronale Netze .....	72
6.5	Radiale Basisfunktionen .....	76
6.6	Kreuzvalidierung .....	78
6.7	Merkmalsselektion .....	80
	Literatur .....	82
<b>7</b>	<b>Prognose</b> .....	83
7.1	Endliche Zustandsautomaten .....	83
7.2	Rekurrente Modelle .....	85
7.3	Autoregressive Modelle .....	86
	Literatur .....	87
<b>8</b>	<b>Klassifikation</b> .....	89
8.1	Klassifikationskriterien .....	89
8.2	Naiver Bayes Klassifikator .....	93
8.3	Lineare Diskriminanzanalyse .....	96
8.4	Supportvektormaschine .....	98
8.5	Nächster-Nachbar-Klassifikator .....	100
8.6	Lernende Vektorquantisierung .....	101
8.7	Entscheidungsbäume .....	102
	Literatur .....	106
<b>9</b>	<b>Clustering</b> .....	109
9.1	Clusterpartitionen .....	109
9.2	Sequenzielles Clustering .....	111
9.3	Prototypbasiertes Clustering .....	114
9.4	Fuzzy-Clustering .....	116
9.5	Relationales Clustering .....	121
9.6	Clustertendenz .....	125
9.7	Clustervalidität .....	126

9.8      Selbstorganisierende Karte .....	127
Literatur .....	129
<b>Anhang: Optimierungsverfahren .....</b>	<b>131</b>
<b>Lösungen der Übungsaufgaben .....</b>	<b>135</b>
<b>Sachverzeichnis .....</b>	<b>139</b>

---

# Symbolverzeichnis

$\forall x \in X$	für jedes $x$ aus $X$
$\exists x \in X$	es existiert ein $x$ aus $X$ so, dass
$\Rightarrow$	wenn ... dann ...
$\Leftrightarrow$	genau dann, wenn
$b$	
$\int_a^b f dx$	Integral von $f$ von $x = a$ bis $x = b$
$\frac{\partial f}{\partial x}$	partielle Ableitung von $f$ nach $x$
$\wedge$	Konjunktion
$\vee$	Disjunktion
$\cap$	Durchschnitt
$\cup$	Vereinigung
$\neg$	Komplement
$\setminus$	Mengendifferenz
$\subset, \subseteq$	Inklusion
$\cdot$	Produkt, inneres Produkt, Skalarprodukt
$\times$	Vektorprodukt, Kreuzprodukt
$\{\}$	leere Menge
$[x, y]$	geschlossenes Intervall von $x$ bis $y$
$(x, y], [x, y)$	halboffene Intervalle von $x$ bis $y$
$(x, y)$	offenes Intervall von $x$ bis $y$
$ x $	Absolutbetrag von $x$
$ X $	Kardinalität der Menge $X$
$\ x\ $	Norm von $x$
$\lfloor x \rfloor$	kleinste ganze Zahl $a \geq x$
$\lceil x \rceil$	größte ganze Zahl $a \leq x$
$\binom{n}{m}$	Vektor mit den Komponenten $n$ und $m$ , Binomialkoeffizient
$\infty$	unendlich
$a \ll b$	$a$ ist vernachlässigbar klein gegenüber $b$
$a \gg b$	$b$ ist vernachlässigbar klein gegenüber $a$
$\alpha(t)$	zeitlich veränderliche Lernrate

$\operatorname{argmin} X$	Index des Minimums von $X$
$\operatorname{argmax} X$	Index des Maximums von $X$
$\arctan x$	Arcus Tangens von $x$
$\operatorname{artanh} x$	inverser hyperbolischer Tangens von $x$
$c_{ij}$	Kovarianz zwischen den Merkmalen $i$ und $j$
$CE(U)$	Klassifikationsentropie von $U$
$\operatorname{cov} X$	Kovarianzmatrix von $X$
$d(a, b)$	Abstand zwischen $a$ und $b$
$\operatorname{eig} X$	Eigenvektoren und Eigenwerte der Matrix $X$
$F_c$	Fouriercosinustransformierte
$F_s$	Fouriersinustransformierte
$h(X)$	Hopkins-Index von $X$
$H(a, b)$	Hamming-Abstand zwischen $a$ und $b$
$H(Z)$	minimaler Hyperwürfel oder Entropie von $Z$
$H(Z \mid a)$	Entropie von $Z$ unter der Bedingung $a$
$\inf X$	Infimum von $X$
$\lambda$	Eigenwert, Lagrange-Variable
$L(a, b)$	Edit-Abstand zwischen $a$ und $b$
$\lim_{x \rightarrow y}$	Grenzwert für $x$ gegen $y$
$\log_b x$	Logarithmus von $x$ zur Basis $b$
$\max X$	Maximum von $X$
$\min X$	Minimum von $X$
$a \bmod b$	$a$ Modulo $b$
$N(\mu, \sigma)$	Normalverteilung mit Mittelwert $\mu$ und Standardabweichung $\sigma$
$\text{NaN}$	undefinierter Wert (engl. <i>Not a Number</i> )
$PC(U)$	Partitionskoeffizient von $U$
$\mathbb{R}$	Menge der reellen Zahlen
$\mathbb{R}^+$	Menge der positiven reellen Zahlen
$r$	Radius
$s$	Standardabweichung
$s_{ij}$	Korrelation zwischen den Merkmalen $i$ und $j$
$\sup X$	Supremum von $X$
$\tanh x$	hyperbolischer Tangens von $x$
$u_{ik}$	Zugehörigkeit des $k$ -ten Vektors zum $i$ -ten Cluster
$X$	Menge oder Matrix $X$
$\bar{x}$	Mittelwert von $X$
$X^T, x^T$	Transponierte der Matrix $X$ oder des Vektors $x$
$x_k$	$k$ -ter Vektor aus $X$
$x^{(i)}$	$i$ -te Komponente aus $X$
$x_k^{(i)}$	$i$ -te Komponente des $k$ -ten Vektors aus $X$
$x$	Skalar oder Vektor $x$
$x(t)$	Zeitsignal
$x(j2\pi f)$	Spektrum

## Zusammenfassung

Dieses Buch behandelt Modelle und Algorithmen für die Analyse von Daten, zum Beispiel Daten aus industriellen und geschäftlichen Prozessen, Text und strukturierte Daten, Bilddaten oder biomedizinische Daten. Es werden die Begriffe Datenanalyse, Data Mining, Knowledge Discovery sowie die KDD- und CRISP-DM-Prozesse eingeführt. Typische Datenanalyseprojekte lassen sich in mehrere Phasen gliedern: Vorbereitung, Vorverarbeitung, Analyse und Nachbereitung. Die einzelnen Kapitel dieses Buches behandeln die wichtigsten Methoden der Datenvorverarbeitung und -analyse: Daten und Relationen, Datenvorverarbeitung, Visualisierung, Korrelation, Regression, Prognose, Klassifikation und Clustering.

## 1.1 Daten

Der Fokus dieses Buches ist die Analyse großer Datenmengen, zum Beispiel:

- Industrielle Prozessdaten: Zur Automatisierung und Steuerung industrieller Prozesse werden immer mehr Daten erfasst, gespeichert und verarbeitet. Die Informationstechnik fließt in alle Ebenen der Automatisierung ein und betrifft somit Signale von Sensoren und Aktuatoren in der Feldebene ebenso wie Steuerungs- und Regelungssignale in der Steuerungsebene, Bedienungs- und Beobachtungsdaten in der Prozessleitebene sowie Planungsdaten und Kennzahlen in der Betriebsleit- und Unternehmensebene. Auf allen diesen Ebenen lassen sich mit Data Mining Potenziale zur Prozessoptimierung und Steigerung der Wettbewerbsfähigkeit erschließen.
- Geschäftsdaten: Daten aus Geschäftsprozessen werden analysiert, um diese Prozesse besser zu verstehen und zu optimieren. Hierzu gehören Daten zu Kunden, Portfolio, Vertriebs- und Marketingprozessen, Service, Preisbildung, Finanzen, Risiken oder

Compliance. In der Warenkorbanalyse wird beispielsweise ausgewertet, welche Produkte in welchen Stückzahlen mit welchen anderen Produkten zusammen gekauft wurden, etwa im Supermarkt. Die Auswertung hat das Ziel, Produkte optimal zu platzieren oder die Preisgestaltung zu optimieren. Ein anderes wichtiges Anwendungsbereich ist die datenbasierte Kundensegmentierung, die für gezielte Kundenangebote und Werbemaßnahmen genutzt werden kann.

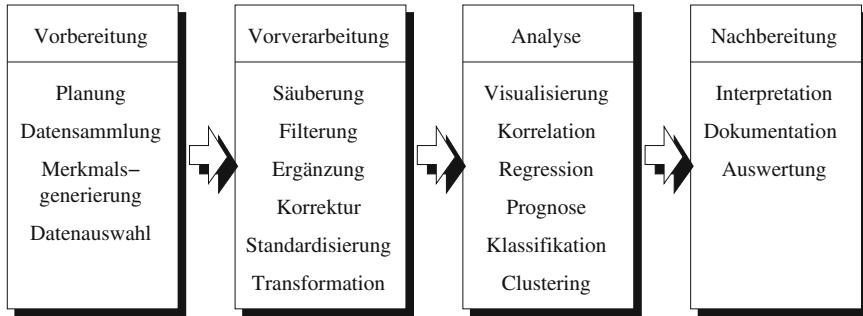
- Textdaten und strukturierte Daten: Numerische Daten stehen seit vielen Jahren im Mittelpunkt der Datenanalyse. Heute gewinnen Text und strukturierte Daten zunehmend an Bedeutung: Textdokumente, elektronische Nachrichten (z.B. E-Mail und Nachrichten in sozialen Medien), Web-Dokumente oder vernetzte Datenbanken. Text und nicht-numerische Informationen werden gefiltert, durchsucht, Informationen extrahiert und strukturiert. Strukturierte Daten verwenden definierte Ordnungskriterien wie Felder oder Objektstrukturen und werden oft mit semantischen und ontologischen Modellen beschrieben.
- Bild- und Videodaten: Durch die zunehmende Verfügbarkeit von preisgünstigen elektronischen Kameras (etwa in Smartphones) und satellitenbasierte Bildquellen stehen große Mengen von zweidimensionalen und dreidimensionalen Bild- und Videodaten für die Analyse zur Verfügung. Mit Data-Mining-Methoden werden Objekte gesucht und erkannt, Szenen analysiert und klassifiziert, und Bilddaten mit anderen Informationen in Beziehung gesetzt.
- Biomedizinische Daten: Labordaten helfen, biologische und medizinische Prozesse zu analysieren, zu verstehen und zu nutzen, z. B. bei der DNA- oder Protein-Analyse.

---

## 1.2 Data Mining, Data Analytics und Knowledge Discovery

Der Begriff „Data Mining“ reicht bis in die 1980er Jahre zurück [3]. Data Mining hat das Ziel, Wissen aus Daten zu extrahieren [1]. Unter Wissen verstehen wir interessante Muster, die allgemein gültig sind, nicht trivial, neu, nützlich und verständlich. Inwieweit diese Eigenschaften auf die erkannten Muster zutreffen, muss im Kontext der Anwendung beurteilt werden, und zwar in der Regel gemeinsam mit Anwendungsexperten. Durch die Einbeziehung der Experten entsteht ein rückgekoppelter Prozess, der oft wiederholt durchlaufen wird, bis ein zufriedenstellendes Ergebnis erreicht ist. Der Begriff „Data Analytics“ kam in den 2000er Jahren auf [2, 6], definiert als die Anwendung von Computersystemen auf die Analyse großer Datenmengen zur Entscheidungsunterstützung.

Datenanalyse ist ein sehr interdisziplinäres Gebiet, in das Aspekte aus vielen verschiedenen wissenschaftlichen Disziplinen eingeflossen sind, z. B. Statistik, maschinelles Lernen, Mustererkennung, Systemtheorie oder künstliche Intelligenz. Typische Datenanalyseprojekte können in mehrere Phasen gegliedert werden. Daten werden erfasst, ausgewählt, gesäubert, gefiltert, visualisiert, analysiert, und die Analyseergebnisse werden schließlich interpretiert und genutzt. Der *KDD-Prozess* (engl. *Knowledge Discovery in*



**Abb. 1.1** Der Data-Mining-Prozess

*Databases*) [1] umfasst die sechs Phasen Selektion, Vorverarbeitung, Transformation, Data Mining, Interpretation und Evaluierung. Der *CRISP-DM-Prozess* (engl. *Cross Industry Standard Process for Data Mining*) [5] umfasst die sechs Phasen Domänenverständnis, Datenverständnis, Datenaufbereitung, Modellierung, Evaluierung und Einsatz. In diesem Buch betrachten wir ein vereinfachtes Modell mit vier Phasen: Vorbereitung, Vorverarbeitung, Analyse und Nachbereitung (Abb. 1.1). Der Schwerpunkt dieses Buchs liegt in der Datenvorverarbeitung und Datenanalyse. Die Kapitel sind nach den wichtigsten Methoden der Vorverarbeitung und Analyse gegliedert: Daten und Relationen, Datenvorverarbeitung, Datenvisualisierung, Korrelation, Regression, Prognose, Klassifikation und Clustering.

Data Mining wird heute durch eine zunehmende Anzahl von Software-Tools unterstützt, z. B. KNIME, MATLAB, SPSS, SAS, STATISTICA, TIBCO Spotfire, R, Rapid Miner, Tableau, QlikView, oder WEKA. Solche Software-Tools werden in diesem Buch weder beschrieben, verglichen noch empfohlen. Für einen Überblick über aktuelle Software-Tools verweisen wir auf [4, 7].

---

## Literatur

- U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining*. AAAI Press, Menlo Park, 1996.
- R. Kohavi, N. J. Rothleider, and E. Simoudis. Emerging trends in business analytics. *Communications of the ACM*, 45(8):345–48, 2002.
- M. C. Lovell. Data mining. *Review of Economics and Statistics*, 65(1):1–11, 1983.
- R. Mikut and M. Reischl. Data mining tools. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(5):431–443, 2011.
- C. Shearer. The CRISP-DM model: The new blueprint for data mining. *J Data Warehousing*, 5(4):13–22, 2000.
- S. Tyagi. Using data analytics for greater profits. *Journal of Business Strategy*, 24(3):12–14, 2003.
- L. Zhang, A. Stoffel, M. Behrisch, S. Mittelstadt, T. Schreck, R. Pompl, S. Weber, H. Last, and D. Keim. Visual analytics for the big data era — a comparative review of state-of-the-art commercial systems. In *IEEE Conference on Visual Analytics Science and Technology*, pages 173–182, Seattle, 2012.

## Zusammenfassung

Die Grundkonzepte der Datenanalyse werden anhand des bekannten Iris-Datensatzes eingeführt. Datenskalen (nominal, ordinal, Intervall, proportional) müssen berücksichtigt werden, weil bestimmte mathematische Operationen nur für gewisse Skalen geeignet sind. Numerische Daten können als Mengen, Vektoren oder Matrizen repräsentiert werden. Viele Datenanalyseverfahren basieren auf Unähnlichkeitsmaßen (z. B. Matrixnormen, Lebesgue/Minkowski-Normen) oder Ähnlichkeitsmaßen (z. B. Cosinus, Überlapp, Dice, Jaccard, Tanimoto). Sequenzen können mit Sequenzrelationen analysiert werden (z. B. Hamming, Levenshtein/Edit-Abstand). Aus kontinuierlichen analogen Signalen können Daten durch Abtastung und Quantisierung extrahiert werden. Die Nyquist-Bedingung ermöglicht eine Abtastung ohne Informationsverlust.

---

## 2.1 Der Iris-Datensatz

Zur Einführung der Grundkonzepte der Datenanalyse betrachten wir einen der bekanntesten historischen Referenz-Datensätze: den *Iris-Datensatz* [1]. Der Iris-Datensatz wurde 1935 von dem amerikanischen Botaniker Edgar Anderson erstellt, der die geographischen Unterschiede der Schwertlilien (*Iris*) untersuchte. Die von Anderson erhobenen Daten über Schwertlilien auf der Gaspé-Halbinsel in Quebec (Kanada) wurden erstmals von Sir Ronald Aylmer Fisher 1936 als Beispiel für die Diskriminanzanalyse multivariater Daten (siehe Kap. 8) verwendet [4] und entwickelten sich später zu einem der meistverwendeten Referenzdatensätze in Statistik und Datenanalyse. Der Iris-Datensatz umfasst Messdaten von 150 Irispflanzen, davon jeweils 50 Borsten-Schwertlilien (*Iris Setosa*), Virginia-Schwertlilien (*Iris Virginica*) und verschiedenfarbige Schwertlilien (*Iris Versicolor*). Der Datensatz enthält von jeder dieser 150 Pflanzen die Länge und Breite eines

Kelchblatts sowie die Länge und Breite eines Blütenblatts (in Zentimetern). Den kompletten Datensatz zeigt Tab. 2.1. Inzwischen sind einige unterschiedliche Variationen dieses Datensatzes verwendet und veröffentlicht worden, so dass bei Vergleichen überprüft werden sollte, welche Variante des Iris-Datensatzes verwendet wurde [2]. Der Iris-Datensatz und viele andere bekannte Referenzdatensätze sind z. B. über die *Machine Learning Data Base* der Universität von Kalifornien (Irvine, UCI) verfügbar.

In der Datenanalyse bezeichnen wir jede der 150 Irispflanzen als ein *Objekt*, jede der 3 Pflanzenarten als eine *Klasse* und jede der 4 Messgrößen als ein *Merkmal*. Typische Fragen, die uns bei der Analyse dieser Daten interessieren, sind:

- Welche dieser Daten enthalten möglicherweise Messfehler oder falsche Klassenzuordnungen?
- Welcher Fehler wird durch die Rundung der Daten auf 0,1 Zentimeter verursacht?
- Wie stark ist der Zusammenhang zwischen Länge und Breite einer Blüte?
- Zwischen welchen Merkmalen besteht der am stärksten ausgeprägte Zusammenhang?
- Keine der Pflanzen in diesem Datensatz hat eine Kelchbreite von 1,8 Zentimetern. Welche Kelchlänge würden wir bei einer solchen Pflanze erwarten?
- Zu welcher Pflanzenart gehört vermutlich eine Iris mit einer Kelchbreite von 1,8 Zentimetern?
- Sind in den drei betrachteten Arten auch Unterarten enthalten, die aus den Daten ersichtlich werden?

In diesem Buch werden wir zahlreiche Methoden kennenlernen, solche und ähnliche Fragestellungen zu beantworten. Zum Verständnis dieser Methoden ist es notwendig, zunächst einige grundlegende Eigenschaften von Daten und deren Relationen zueinander formal zu definieren.

---

## 2.2 Maßskalen

Numerische Informationen können unterschiedliche Bedeutung haben, auch wenn sie durch die gleichen numerischen Daten repräsentiert werden. Je nach semantischer Bedeutung können bestimmte mathematische Operationen zulässig oder unzulässig sein. Für die semantische Bedeutung numerischer Daten wurden von Stevens [7] vier unterschiedliche Skalen vorgeschlagen (Tab. 2.2). Für nominalskalierte Daten (unterste Zeile) sind nur Test auf Gleichheit und Ungleichheit zulässig. Beispiele für nominalskalierte Daten sind Namen von Personen oder Indizes von Objekten. Daten eines nominalskalierten Merkmals können durch den *Modus* (oder *Modalwert*), also dem am häufigsten vorkommenden Wert, repräsentiert werden. Für ordinalskalierte Daten (zweite Zeile von unten) sind die Ordnungsrelationen  $>$  bzw.  $<$  gültig. Auf jedem Skalen-Niveau sind auch die Operationen und statistischen Maße aller niedrigen Skalen-Niveaus gültig. Für ordinalskalierte Daten

**Tab. 2.1** Iris-Datensatz [1]

Setosa				Versicolor				Virginica			
Kelchblatt		Blütenblatt		Kelchblatt		Blütenblatt		Kelchblatt		Blütenblatt	
Länge	Breite	Länge	Breite	Länge	Breite	Länge	Breite	Länge	Breite	Länge	Breite
5.1	3.5	1.4	0.2	7	3.2	4.7	1.4	6.3	3.3	6	2.5
4.9	3	1.4	0.2	6.4	3.2	4.5	1.5	5.8	2.7	5.1	1.9
4.7	3.2	1.3	0.2	6.9	3.1	4.9	1.5	7.1	3	5.9	2.1
4.6	3.1	1.5	0.2	5.5	2.3	4	1.3	6.3	2.9	5.6	1.8
5	3.6	1.4	0.2	6.5	2.8	4.6	1.5	6.5	3	5.8	2.2
5.4	3.9	1.7	0.4	5.7	2.8	4.5	1.3	7.6	3	6.6	2.1
4.6	3.4	1.4	0.3	6.3	3.3	4.7	1.6	4.9	2.5	4.5	1.7
5	3.4	1.5	0.2	4.9	2.4	3.3	1	7.3	2.9	6.3	1.8
4.4	2.9	1.4	0.2	6.6	2.9	4.6	1.3	6.7	2.5	5.8	1.8
4.9	3.1	1.5	0.1	5.2	2.7	3.9	1.4	7.2	3.6	6.1	2.5
5.4	3.7	1.5	0.2	5	2	3.5	1	6.5	3.2	5.1	2
4.8	3.4	1.6	0.2	5.9	3	4.2	1.5	6.4	2.7	5.3	1.9
4.8	3	1.4	0.1	6	2.2	4	1	6.8	3	5.5	2.1
4.3	3	1.1	0.1	6.1	2.9	4.7	1.4	5.7	2.5	5	2
5.8	4	1.2	0.2	5.6	2.9	3.6	1.3	5.8	2.8	5.1	2.4
5.7	4.4	1.5	0.4	6.7	3.1	4.4	1.4	6.4	3.2	5.3	2.3
5.4	3.9	1.3	0.4	5.6	3	4.5	1.5	6.5	3	5.5	1.8
5.1	3.5	1.4	0.3	5.8	2.7	4.1	1	7.7	3.8	6.7	2.2
5.7	3.8	1.7	0.3	6.2	2.2	4.5	1.5	7.7	2.6	6.9	2.3
5.1	3.8	1.5	0.3	5.6	2.5	3.9	1.1	6	2.2	5	1.5
5.4	3.4	1.7	0.2	5.9	3.2	4.8	1.8	6.9	3.2	5.7	2.3
5.1	3.7	1.5	0.4	6.1	2.8	4	1.3	5.6	2.8	4.9	2
4.6	3.6	1	0.2	6.3	2.5	4.9	1.5	7.7	2.8	6.7	2
5.1	3.3	1.7	0.5	6.1	2.8	4.7	1.2	6.3	2.7	4.9	1.8
4.8	3.4	1.9	0.2	6.4	2.9	4.3	1.3	6.7	3.3	5.7	2.1
5	3	1.6	0.2	6.6	3	4.4	1.4	7.2	3.2	6	1.8
5	3.4	1.6	0.4	6.8	2.8	4.8	1.4	6.2	2.8	4.8	1.8
5.2	3.5	1.5	0.2	6.7	3	5	1.7	6.1	3	4.9	1.8
5.2	3.4	1.4	0.2	6	2.9	4.5	1.5	6.4	2.8	5.6	2.1
4.7	3.2	1.6	0.2	5.7	2.6	3.5	1	7.2	3	5.8	1.6

**Tab. 2.1** (Fortsetzung)

Setosa				Versicolor				Virginica			
Kelchblatt		Blütenblatt		Kelchblatt		Blütenblatt		Kelchblatt		Blütenblatt	
Länge	Breite	Länge	Breite	Länge	Breite	Länge	Breite	Länge	Breite	Länge	Breite
4.8	3.1	1.6	0.2	5.5	2.4	3.8	1.1	7.4	2.8	6.1	1.9
5.4	3.4	1.5	0.4	5.5	2.4	3.7	1	7.9	3.8	6.4	2
5.2	4.1	1.5	0.1	5.8	2.7	3.9	1.2	6.4	2.8	5.6	2.2
5.5	4.2	1.4	0.2	6	2.7	5.1	1.6	6.3	2.8	5.1	1.5
4.9	3.1	1.5	0.2	5.4	3	4.5	1.5	6.1	2.6	5.6	1.4
5	3.2	1.2	0.2	6	3.4	4.5	1.6	7.7	3	6.1	2.3
5.5	3.5	1.3	0.2	6.7	3.1	4.7	1.5	6.3	3.4	5.6	2.4
4.9	3.6	1.4	0.1	6.3	2.3	4.4	1.3	6.4	3.1	5.5	1.8
4.4	3	1.3	0.2	5.6	3	4.1	1.3	6	3	4.8	1.8
5.1	3.4	1.5	0.2	5.5	2.5	4	1.3	6.9	3.1	5.4	2.1
5	3.5	1.3	0.3	5.5	2.6	4.4	1.2	6.7	3.1	5.6	2.4
4.5	2.3	1.3	0.3	6.1	3	4.6	1.4	6.9	3.1	5.1	2.3
4.4	3.2	1.3	0.2	5.8	2.6	4	1.2	5.8	2.7	5.1	1.9
5	3.5	1.6	0.6	5	2.3	3.3	1	6.8	3.2	5.9	2.3
5.1	3.8	1.9	0.4	5.6	2.7	4.2	1.3	6.7	3.3	5.7	2.5
4.8	3	1.4	0.3	5.7	3	4.2	1.2	6.7	3	5.2	2.3
5.1	3.8	1.6	0.2	5.7	2.9	4.2	1.3	6.3	2.5	5	1.9
4.6	3.2	1.4	0.2	6.2	2.9	4.3	1.3	6.5	3	5.2	2
5.3	3.7	1.5	0.2	5.1	2.5	3	1.1	6.2	3.4	5.4	2.3
5	3.3	1.4	0.2	5.7	2.8	4.1	1.3	5.9	3	5.1	1.8

**Tab. 2.2** Maßskalen

Skala	Operation		Beispiel	Statistisches Maß
Proportional	.	/	273° K, 21 Jahre	Verallgemeinerter Mittelwert
Intervall	+	-	20° C, 2020 n. Chr.	Mittelwert
Ordinal	>	<	Sehr gut, gut, befriedigend	Median
Nominal	=	≠	Müller, Meier, Schulz	Modus

sind also die Operationen  $=$ ,  $\neq$ ,  $>$  und  $<$  gültig, und somit auch die Kombinationen „größer oder gleich“ ( $\geq$ ) und „kleiner oder gleich“ ( $\leq$ ). Die Relation „kleiner oder gleich“ ( $\leq$ ) definiert eine *totale Ordnung*, so dass für alle  $x, y, z$  gilt  $(x \leq y) \wedge (y \leq x) \Rightarrow (x = y)$  (Antisymmetrie),  $(x \leq y) \wedge (y \leq z) \Rightarrow (x \leq z)$  (Transitivität) und  $(x \leq y) \vee (y \leq x)$

(Totalität). Beispiele für ordinalskalierte Daten sind Schulnoten. Daten eines ordinalskalierten Merkmals können durch den *Median* repräsentiert werden, also den Wert für den (ungefähr) so viele größere wie kleinere Werte vorliegen. Der *Mittelwert* ist für ordinalskalierte Daten nicht zulässig. Es ist also nicht sinnvoll, von einem Noten-Mittelwert von 3,0 zu sprechen. Für intervallskalierte Daten (dritte Zeile von unten) sind Addition und Subtraktion gültig. Intervallskalierte Merkmale haben kontextspezifisch definierte Nullpunkte. Beispiele sind Jahreszahlen nach Christus oder Temperaturen in Grad Celsius oder Grad Fahrenheit. Es also nicht sinnvoll zu sagen, dass  $40^\circ\text{C}$  doppelt so warm ist wie  $20^\circ\text{C}$ . Daten eines intervallskalierten Merkmals, z. B. die Daten  $X = \{x_1, \dots, x_n\}$ , können durch den (arithmetischen) *Mittelwert*

$$\bar{x} = \frac{1}{n} \sum_{k=1}^n x_k \quad (2.1)$$

repräsentiert werden. Für proportionalskalierte Daten (oberste Zeile) sind Multiplikation und Division gültig. Beispiele für proportionalskalierte Merkmale sind Zeitdifferenzen (z. B. Alter) oder Temperaturen auf der Kelvin-Skala. Daten eines proportionalskalierten Merkmals können durch den *verallgemeinerten Mittelwert*

$$m_\alpha(X) = \sqrt[\alpha]{\frac{1}{n} \sum_{k=1}^n x_k^\alpha} \quad (2.2)$$

$\alpha \in \mathbb{R}$ , repräsentiert werden. Der verallgemeinerte Mittelwert enthält die Sonderfälle Minimum ( $\alpha \rightarrow -\infty$ ), harmonischer Mittelwert ( $\alpha = -1$ ), geometrischer Mittelwert ( $\alpha \rightarrow 0$ ), arithmetischer Mittelwert ( $\alpha = 1$ ), quadratischer Mittelwert ( $\alpha = 2$ ) und Maximum ( $\alpha \rightarrow \infty$ ).

Die Messdaten aus dem Iris-Datensatz sind proportionalskalierte Daten. So kann zum Beispiel aus der Länge und Breite von Kelch- und Blütenblättern durch Multiplikation näherungsweise die Fläche der Kelch- und Blütenblätter bestimmt werden. Es sind daher alle statistischen Maße Modus, Median, Mittelwert und verallgemeinerter Mittelwert zulässig. Wir berechnen diese für die Blütenblattbreite (viertes Merkmal). Der Datensatz enthält Blütenblattbreiten zwischen 0,1 und 2,5 Zentimetern. Die am häufigsten vorkommende Blütenblattbreite ist 0,2 Zentimeter, und zwar bei 29 Pflanzen. Der Modalwert der Blütenblattbreite ist also 0,2 Zentimeter. Zur Berechnung des Median wird die Anzahl der Pflanzen mit den Blütenblattbreiten 0,1 Zentimeter, 0,2 Zentimeter, 0,3 Zentimeter usw. aufsummiert, bis die Hälfte der Objekte (75) erfasst ist. Dies ist im linken Teil von Tab. 2.3 dargestellt. Der Median der Blütenblattbreite ist also 1,3 Zentimeter. Alternativ kann die Summierung auch in absteigender Reihenfolge (rechter Teil von Tab. 2.3) erfolgen. Dieser Algorithmus hat die zeitliche Komplexität  $O(n \log n)$ . Allerdings (und das ist selbst in der Wissenschaft wenig bekannt) kann der Median mit Hilfe sogenannter Selektionsalgorithmen effizient in linearer Zeit berechnet werden [3]. Der Mittelwert der Blütenblattbreite ist schließlich ca. 1,19933.

Viele Methoden, die in diesem Buch beschrieben sind, gehen von intervall- oder proportionalskalierten Daten aus und verwenden das entsprechende maßtheoretisch zulässige

**Tab. 2.3** Berechnung des Medians der Blütenblattbreite (Iris)

Wert	Anzahl	kumulierte Summe	Wert	Anzahl	kumulierte Summe
0,1	5	5	2,5	3	3
0,2	29	34	2,4	3	6
0,3	7	41	2,3	8	14
0,4	7	48	2,2	3	17
0,5	1	49	2,1	6	23
0,6	1	50	2	6	29
0,7	0	50	1,9	5	34
0,8	0	50	1,8	12	46
0,9	0	50	1,7	2	48
1	7	57	1,6	4	52
1,1	3	60	1,5	12	64
1,2	5	65	1,4	8	72
1,3	10(13)	75	1,3	3(13)	75

mathematische Instrumentarium. Es werden jedoch auch zahlreiche Methoden zur Analyse von nominal- und ordinalskalierten Daten beschrieben. Diese erfolgt meist mit Hilfe von Relationen bzw. relationalen Analysemethoden.

## 2.3 Mengen- und Matrixdarstellung

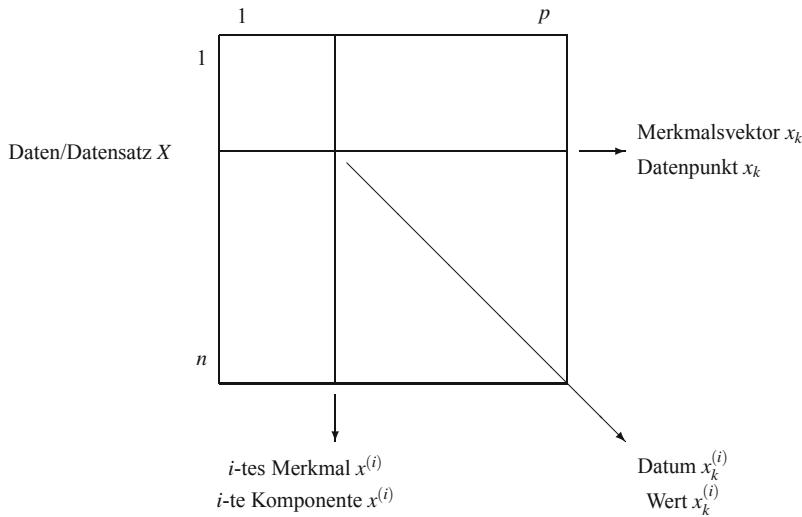
Jeder numerischer Merkmalsdatensatz lässt sich als Menge

$$X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p \quad (2.3)$$

schreiben. Ein solcher Datensatz enthält  $n \in \{1, 2, \dots\}$  Elemente. Jedes Element ist ein  $p$ -dimensionaler reellwertiger Merkmalsvektor,  $p \in \{1, 2, \dots\}$ . Für  $p = 1$  heißt  $X$  *skalarer Datensatz*. Außer der Mengenschreibweise wird häufig auch die Matrixschreibweise

$$X = \begin{pmatrix} x_1^{(1)} & \cdots & x_1^{(p)} \\ \vdots & \ddots & \vdots \\ x_n^{(1)} & \cdots & x_n^{(p)} \end{pmatrix} \quad (2.4)$$

verwendet. Die Vektoren  $x_1, \dots, x_n$  sind also *Zeilenvektoren*. Mathematisch nicht ganz sauber werden die Menge  $X$  und die Matrix  $X$  häufig als äquivalente Repräsentationen eines Datensatzes benutzt. Abbildung 2.1 zeigt die üblichen Bezeichnungen der Elemente von Datenmatrizen. Jede *Zeile* in der Datenmatrix entspricht einem *Element* der Datenmenge und wird als *Merkmalsvektor* oder *Datenpunkt*  $x_k$  bezeichnet,  $k = 1, \dots, n$ . Jede



**Abb. 2.1** Matrixschreibweise eines Datensatzes

*Spalte* in der Datenmatrix entspricht einer *Komponente* aller Elemente der Datenmenge und wird als *i-tes Merkmal* oder *i-te Komponente*  $x^{(i)}$  bezeichnet,  $i = 1, \dots, p$ . Zeilen und Spalten werden in diesem Buch durch tiefgestellte Indizes für Zeilen und eingeklammerte hochgestellte Indizes für Spalten unterschieden. In der Literatur finden sich auch alternative Schreibweisen, zum Beispiel  $x(k, .)$  und  $x(., i)$ . Ein einzelnes *Matrixelement* entspricht *einer Komponente eines Elements* der Datenmenge und wird *Datum* oder *Wert*  $x_k^{(i)}$  genannt,  $k = 1, \dots, n, i = 1, \dots, p$ .

Der Iris-Datensatz lässt sich als eine solche Datenmatrix mit 150 Zeilen und 4 Spalten darstellen. Jede Zeile dieser Matrix entspricht einer der 150 Pflanzen, und jede Spalte der Matrix entspricht einer der 4 Messgrößen. Die Iris-Datenmatrix kann also durch Untereinanderhängen der drei Teile in Tab. 2.1 erzeugt werden. Die Klasseninformation (Setosa, Versicolor, Virginica) kann als zusätzliches fünftes (ordinalskaliertes) Merkmal in einer solchen Datenmatrix interpretiert werden.

## 2.4 Relationen

Ohne Bezug auf numerische Merkmale schreiben wir eine Menge von (abstrakten) Elementen als

$$O = \{o_1, \dots, o_n\} \quad (2.5)$$

Häufig lassen sich solche Objekte  $o_k, k = 1, \dots, n$ , nicht sinnvoll mit Merkmalsvektoren repräsentieren, so dass konventionelle merkmalsbasierte Datenanalysemethoden nicht verwendet werden können. Stattdessen lässt sich oft eine *Relation* aller Paare von Objekten

quantifizieren und als quadratische Matrix

$$R = \begin{pmatrix} r_{11} & \cdots & r_{1n} \\ \vdots & \ddots & \vdots \\ r_{n1} & \cdots & r_{nn} \end{pmatrix} \in \mathbb{R}^{n \times n} \quad (2.6)$$

schreiben. Durch die Relationswerte  $r_{ij}$ ,  $i, j = 1, \dots, n$ , lassen sich Grade der Ähnlichkeit, Unähnlichkeit, Vereinbarkeit, Unvereinbarkeit, Nähe oder Abstand zwischen den Objekten  $o_i$  und  $o_j$  quantifizieren.  $R$  heißt *symmetrisch*, wenn  $r_{ij} = r_{ji}$  für alle  $i, j = 1, \dots, n$ . Die Werte in  $R$  können manuell definiert oder aus Merkmalen berechnet werden. Wenn sinnvolle numerische Merkmale  $X$  verfügbar sind, dann kann  $R$  durch eine geeignete Funktion  $f: \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}$  berechnet werden. Zum Beispiel könnte eine Relationsmatrix für die Iris-Daten manuell durch einen Botaniker spezifiziert werden, der die Ähnlichkeit jedes Paares von Pflanzen begutachtet und dann numerisch quantifiziert, oder sie könnte mit einer geeigneten Funktion  $f$  aus den Längen und Breiten der Kelch- und Blütenblätter berechnet werden. Die folgenden beiden Abschnitte behandeln die in der Datenanalyse wichtigsten Klassen von Relationen: Unähnlichkeiten und Ähnlichkeiten.

## 2.5 Unähnlichkeitsmaße

Eine Funktion  $d$  heißt *Unähnlichkeitsmaß* oder *Distanzmaß* wenn für alle  $x, y \in \mathbb{R}^p$  gilt

$$d(x, y) = d(y, x) \quad (2.7)$$

$$d(x, y) = 0 \Leftrightarrow x = y \quad (2.8)$$

$$d(x, z) \leq d(x, y) + d(y, z) \quad (2.9)$$

Aus diesen Axiomen folgt

$$d(x, y) \geq 0 \quad (2.10)$$

Eine Klasse von Unähnlichkeitsmaßen ist definiert durch eine *Norm*  $\|\cdot\|$  von  $x - y$ , also

$$d(x, y) = \|x - y\| \quad (2.11)$$

Eine Funktion  $\|\cdot\|: \mathbb{R}^p \rightarrow \mathbb{R}^+$  heißt Norm genau dann, wenn

$$\|x\| = 0 \Leftrightarrow x = (0, \dots, 0) \quad (2.12)$$

$$\|a \cdot x\| = |a| \cdot \|x\| \quad \forall a \in \mathbb{R}, x \in \mathbb{R}^p \quad (2.13)$$

$$\|x + y\| \leq \|x\| + \|y\| \quad \forall x, y \in \mathbb{R}^p \quad (2.14)$$

Zum Beispiel ist die häufig benutzte sogenannte *hyperbolische Norm*

$$\|x\|_h = \prod_{i=1}^p x^{(i)} \quad (2.15)$$

keine Norm gemäß obiger Definition, denn die Bedingung (2.12) wird verletzt durch  $x = (0, 1) \neq (0, 0)$  mit  $\|x\|_h = \|(0, 1)\|_h = 0$ , bzw. Bedingung (2.13) wird verletzt durch  $x = (1, 1)$  und  $a = 2$  mit  $\|a \cdot x\|_h = \|2 \cdot (1, 1)\|_h = \|(2, 2)\|_h = 4 \neq |a| \cdot \|x\|_h = |2| \cdot \|(1, 1)\|_h = 2$ .

Zu den häufig verwendeten Normen gehören die *Matrixnormen* und die Lebesgue- oder *Minkowski-Normen*. Die Matrixnorm ist definiert als

$$\|x\|_A = \sqrt{x A x^T} \quad (2.16)$$

mit einer Matrix  $A \in \mathbb{R}^{n \times n}$ . Wichtige Sonderfälle von Matrixnormen sind die *Euklidische Norm*

$$A = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \quad (2.17)$$

die *Frobenius-* oder *Hilbert-Schmidt-Norm*

$$A = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix} \quad (2.18)$$

die *Diagonalmatrix* mit merkmalsspezifischen Gewichten

$$A = \begin{pmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_p \end{pmatrix} \quad (2.19)$$

und die *Mahalanobis-Norm*

$$A = \text{cov}^{-1} X = \left( \frac{1}{n-1} \sum_{k=1}^n (x_k - \bar{x})^T (x_k - \bar{x}) \right)^{-1} \quad (2.20)$$

Die Mahalanobis-Norm verwendet die Inverse der Kovarianzmatrix des Datensatzes  $X$  (siehe Kap. 5). Sie passt die Gewichtung der einzelnen Komponenten an die beobachtete Statistik an und berücksichtigt auch Korrelationen zwischen Merkmalen.

Die Lebesgue- oder Minkowski-Norm ist definiert als

$$\|x\|_\alpha = \sqrt[\alpha]{\sum_{j=1}^p |x^{(j)}|^\alpha} \quad (2.21)$$

und entspricht dem verallgemeinerten Mittelwert (2.2) bis auf einen konstanten Faktor  $\sqrt[n]{n}$ . Wichtige Sonderfälle der Lebesgue- oder Minkowski-Norm sind die Infimum-Norm ( $\alpha \rightarrow -\infty$ )

$$\|x\|_{-\infty} = \min_{j=1, \dots, p} x^{(j)} \quad (2.22)$$

die *Manhattan-* oder *City-Block-Norm* ( $\alpha = 1$ )

$$\|x\|_1 = \sum_{j=1}^p |x^{(j)}| \quad (2.23)$$

die *Euklidische Norm* ( $\alpha = 2$ ) als einziger Schnittpunkt zwischen Matrixnormen und Lebesgue-/Minkowski-Normen

$$\|x\|_2 = \sqrt{\sum_{j=1}^p (x^{(j)})^2} \quad (2.24)$$

und die Supremum-Norm ( $\alpha \rightarrow \infty$ )

$$\|x\|_\infty = \max_{j=1, \dots, p} |x^{(j)}| \quad (2.25)$$

Ein weiteres häufig verwendetes Unähnlichkeitsmaß ist der *Hamming-Abstand* [5]

$$d_H(x, y) = \sum_{i=1}^p \rho(x^{(i)}, y^{(i)}) \quad (2.26)$$

mit der diskreten Metrik

$$\rho(x, y) = \begin{cases} 0 & \text{falls } x = y \\ 1 & \text{sonst} \end{cases} \quad (2.27)$$

Der Hamming-Abstand ist also die Anzahl der unterschiedlichen Merkmalswerte. Für binärwerte Merkmale ist der Hamming-Abstand gleich dem Manhattan- oder City-Block-Abstand (2.23),  $d_H(x, y) = \|x - y\|_1$ . Der Hamming-Abstand ist jedoch keine Norm, denn Bedingung (2.13) ist verletzt. Eine Variante des Hamming-Abstands ersetzt (2.27) durch andere Funktionen  $\rho$ , um die Ähnlichkeit zwischen Merkmalen zu quantifizieren. Sind die Merkmale beispielsweise (nominalskalierte) Teile einer Maschine, dann könnten ähnlichere Teile zu geringeren Werten von  $\rho$  führen als weniger ähnliche Teile.

---

## 2.6 Ähnlichkeitsmaße

Eine Funktion  $s$  heißt *Ähnlichkeitsmaß* wenn für alle  $x, y \in \mathbb{R}^p$  gilt

$$s(x, y) = s(y, x) \quad (2.28)$$

$$s(x, y) \leq s(x, x) \quad (2.29)$$

$$s(x, y) \geq 0 \quad (2.30)$$

Die Funktion  $s$  heißt *normalisiertes Ähnlichkeitsmaß*, wenn zusätzlich gilt

$$s(x, x) = 1 \quad (2.31)$$

Jedes Unähnlichkeitsmaß  $d$  kann genutzt werden, um ein Ähnlichkeitsmaß  $s$  zu definieren (und umgekehrt), etwa mit einer monoton fallenden positiven Funktion  $f$  mit  $f(0) = 1$  wie beispielsweise

$$s(x, y) = \frac{1}{1 + d(x, y)} \quad (2.32)$$

Die Beispiele aus dem vorherigen Abschnitt werden jedoch meist als Unähnlichkeit  $d$  verwendet und die Beispiele in diesem Abschnitt meist als Ähnlichkeit  $s$ .

Betrachten wir zunächst Ähnlichkeiten zwischen *binären* Merkmalsvektoren. Zwei binäre Merkmalsvektoren können als ähnlich betrachtet werden, wenn viele ihrer Einsen übereinstimmen. Die Anzahl der übereinstimmenden Einsen kann mit dem Skalarprodukt berechnet werden. Bei der Verallgemeinerung für beliebige positive reellwertige Merkmale werden Ähnlichkeitsmaße als unterschiedlich normalisierte Skalarprodukte definiert:

- Kosinus

$$s(x, y) = \frac{\sum_{i=1}^p x^{(i)} y^{(i)}}{\sqrt{\sum_{i=1}^p (x^{(i)})^2 \sum_{i=1}^p (y^{(i)})^2}} \quad (2.33)$$

- Überlapp

$$s(x, y) = \frac{\sum_{i=1}^p x^{(i)} y^{(i)}}{\min \left( \sum_{i=1}^p (x^{(i)})^2, \sum_{i=1}^p (y^{(i)})^2 \right)} \quad (2.34)$$

- Dice

$$s(x, y) = \frac{2 \sum_{i=1}^p x^{(i)} y^{(i)}}{\sum_{i=1}^p (x^{(i)})^2 + \sum_{i=1}^p (y^{(i)})^2} \quad (2.35)$$

- Jaccard (auch Tanimoto)

$$s(x, y) = \frac{\sum_{i=1}^p x^{(i)} y^{(i)}}{\sum_{i=1}^p (x^{(i)})^2 + \sum_{i=1}^p (y^{(i)})^2 - \sum_{i=1}^p x^{(i)} y^{(i)}} \quad (2.36)$$

Für Nullvektoren sind diese Ausdrücke undefiniert, also muss die Ähnlichkeit für diesen Fall explizit definiert werden, z. B. als null. Die Kosinus-Ähnlichkeit ist invariant bezüglich einer (positiven) Skalierung der Merkmalsvektoren und betrachtet daher die relative Verteilung der Merkmalswerte.

$$s(c \cdot x, y) = s(x, y) \quad (2.37)$$

$$s(x, c \cdot y) = s(x, y) \quad (2.38)$$

für alle  $x, y \in \mathbb{R}^p$  und  $c > 0$ . Beispiel: Gegeben sei ein Kuchenrezept mit den Zutaten 1 Ei, 100 g Zucker, 200 g Butter und 300 g Mehl sowie ein zweites Kuchenrezept mit den Zutaten 2 Eier, 200 g Zucker, 400 g Butter und 600 g Mehl. Offensichtlich ist die Zusammensetzung beider Kuchen gleich. Das zweite Rezept ist lediglich für die doppelte Menge ausgelegt. Das Kosinus-Maß ist das einzige der genannten Ähnlichkeitsmaße, das für dieses Beispiel erwartungsgemäß eine Ähnlichkeit von eins liefert.

Die in diesem Abschnitt beschriebenen skalarproduktbasierten Ähnlichkeitsmaße quantifizieren die Ähnlichkeit zwischen Merkmalsvektoren (Zeilen der Datenmatrix). Die in Kap. 5 beschriebenen Korrelationsmaße quantifizieren die Ähnlichkeit zwischen Merkmalen (Spalten der Datenmatrix). Durch Transponieren (Vertauschen von Zeilen und Spalten) der Datenmatrix können also Ähnlichkeitsmaße auch zur Korrelationsanalyse und Korrelationsmaße zur Berechnung von Ähnlichkeiten verwendet werden.

## 2.7 Sequenzrelationen

Die vorigen beiden Abschnitte behandelten Unähnlichkeits- und Ähnlichkeitsmaße für Merkmalsvektoren, deren Elemente sich auf bestimmte Merkmale beziehen. In diesem Abschnitt werden *Sequenzen* von Merkmalswerten oder -vektoren betrachtet, beispielsweise tägliche Temperaturmessungen, Text (Sequenzen von alphanumerischen Zeichen) oder Sequenzen von Webseiten, die von einem Benutzer besucht werden. Solche Sequenzen können zwar grundsätzlich als Merkmalsvektoren betrachtet werden, es ist aber oft sinnvoller, den sequenziellen Charakter der Daten explizit zu berücksichtigen, dass also jedes Element der Sequenz das gleiche Merkmal beschreibt, dass die Reihenfolge eine Rolle spielt und dass die Sequenzen unterschiedliche Längen haben können.

Zum Vergleich einzelner Sequenzelemente wird eine Funktion  $\rho$  verwendet, z. B. die Ungleichheitsfunktion (2.27) des Hamming-Abstands (2.26). Der Hamming-Abstand könnte als Sequenzrelation verwendet werden, aber nur für Paare von gleich langen Sequenzen. Um die Relation zwischen Sequenzen unterschiedlicher Länge zu berechnen, könnten neutrale Elemente (wie Nullen oder Leerzeichen) an die kürzere Sequenz angehängt werden. Je nach Positionierung der Teilesequenzen könnte jedoch eine niedrigere Hamming-Distanz erzielt werden, wenn die neutralen Elemente vorangestellt oder gar optimal eingefügt werden. Dies ist die Idee des *Levenshtein-* oder *Edit-Abstands* [6], der die minimale Anzahl an Editieroperationen (Einfügen, Löschen oder Ändern eines

Für $s = 1, \dots, i$	$L_{s0} = s$
Für $t = 1, \dots, j$	$L_{0t} = t$
Für $s = 1, \dots, i$	Für $t = 1, \dots, j$
	$L_{st} = \min\{L_{s-1,t} + 1,$
	$L_{s,t-1} + 1,$
	$L_{s-1,t-1} + \rho(x^{(s)}, y^{(t)})\}$

**Abb. 2.2** Iterative Berechnung des Edit-Abstands  $L_{ij}$ 

Sequenzelement) bestimmt, um die eine Sequenz in die andere zu überführen. Wir schreiben  $L_{ij}(x, y)$  als Edit-Abstand zwischen den ersten  $i$  Elementen von  $x$  und den ersten  $j$  Elementen von  $y$  und definieren den Edit-Abstand rekursiv als

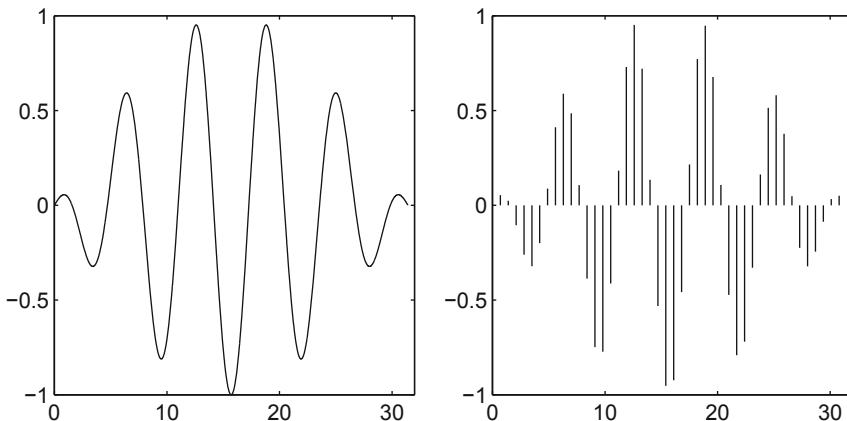
$$L_{ij} = \begin{cases} i & j = 0 \\ j & i = 0 \\ \min\{L_{i-1,j} + 1, L_{i,j-1} + 1, L_{i-1,j-1} + \rho(x^{(i)}, y^{(j)})\} & \text{sonst} \end{cases} \quad (2.39)$$

Die ersten beiden Fälle der Fallunterscheidung bestimmen die Abstände zu leeren Sequenzen und lösen die Rekursion auf. Die drei Argumente des Minimum-Operators im dritten Fall der Fallunterscheidung entsprechen den drei möglichen Editieroperationen Einfügen, Löschen und Ändern. Zur Berechnung von  $L_{ij}$  müssen alle Werte von  $L_{st}$ ,  $s = 1, \dots, i$ ,  $t = 1, \dots, j$  berechnet werden. Die oben angegebene rekursive Berechnung des Edit-Abstands ist nicht effizient, denn viele Werte von  $L_{st}$  werden mehrfach berechnet. Eine effiziente iterative Implementierung des Edit-Abstands, die jeden Wert  $L_{st}$  nur einmal berechnet, ist in Abb. 2.2 gezeigt. Dazu werden zunächst die Abstände der Leerwörter als  $L_{s0} = s$ ,  $s = 1, \dots, i$ , und  $L_{0t} = t$ ,  $t = 1, \dots, j$ , initialisiert und anschließend nacheinander die neuen Abstände  $L_{st}$  aus den bereits berechneten Abständen  $L_{s-1,t}$ ,  $L_{s,t-1}$  und  $L_{s-1,t-1}$  berechnet. Der Algorithmus in Abb. 2.2 berechnet  $L$  zeilenweise. Stattdessen kann  $L$  auch spaltenweise oder diagonal berechnet werden. Abbildung 2.3 zeigt die Berechnung des Edit-Abstands zwischen den alphanumerischen Zeichensequenzen CAESAR und CLEOPATRA. Jedes Matrixelement wird berechnet als Minimum des oberen Nachbarn plus eins, des linken Nachbarn plus eins und des Nachbarn links oben plus dem entsprechenden Zeichenabstand. Der resultierende Edit-Abstand lässt sich als letzten Matrixeintrag rechts unten ablesen und ist in diesem Beispiel gleich 5. Es sind also mindestens 5 Editieroperationen notwendig, um das Wort CAESAR in das Wort CLEOPATRA umzuwandeln, und umgekehrt. Abbildung 2.4 zeigt fünf solche Editieroperationen: Zeichen 2 ändern, Zeichen 4 ändern, Zeichen 5 einfügen, Zeichen 7 einfügen und Zeichen 9 einfügen.

		C	L	E	O	P	A	T	R	A
	0	1	2	3	4	5	6	7	8	9
C	1	0	1	2	3	4	5	6	7	8
A	2	1	1	2	3	4	4	5	6	7
E	3	2	2	1	2	3	4	5	6	7
S	4	3	3	2	2	3	4	5	6	7
A	5	4	4	3	3	3	3	4	5	6
R	6	5	5	4	4	4	4	4	4	5

**Abb. 2.3** Berechnung des Edit-Abstands zwischen den Sequenzen CAESAR und CLEOPATRA

C	A	E	S		A		R	A
0	1	0	1	1	0	1	0	1

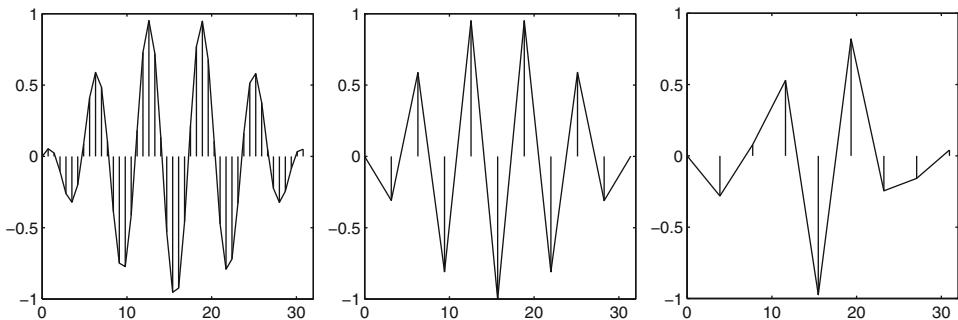
 $\Rightarrow 5$ 
**Abb. 2.4** Operationen zur Umwandlung der Sequenz CAESAR in die Sequenz CLEOPATRA**Abb. 2.5** Originalsignal  $x(t)$  und abgetastete Zeitreihe  $x_k$ 

## 2.8 Abtastung und Quantisierung

Im vorigen Abschnitt wurden diskrete endliche Sequenzen von Merkmalswerten oder Merkmalsvektoren betrachtet. Häufig kann eine solche Sequenz durch *Abtastung* eines kontinuierlichen Signals  $x(t)$  mit einer Abtastperiode  $T$  gewonnen werden, z. B. wenn alle 10 Minuten die Raumtemperatur gemessen wird.

$$x_k = x(k \cdot T), \quad k = 1, \dots, n \quad (2.40)$$

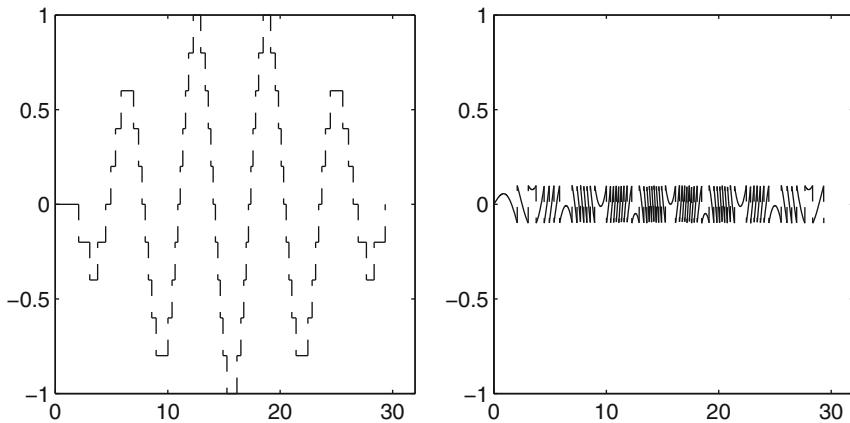
Abbildung 2.5 zeigt ein Beispiel für ein solches Signal  $x(t)$  und eine entsprechend abgetastete Zeitreihe  $x_k$  (vertikale Balken). Die Zeitreihe enthält nur einzelne Werte des kontinuierlichen Signals, aber keine Information über die unendlich vielen Punkte des



**Abb. 2.6** Abgetastete Zeitreihen und Rekonstruktionen durch Polygonzüge für unterschiedliche Abtastperioden

Signals zwischen Paaren von Abtastpunkten. Es wird also nur ein Teil der im Signal enthaltenen Information erfasst. Je größer die Abtastperiode, desto größer ist der Informationsverlust, bis aus der abgetasteten Zeitreihe schließlich keinerlei nutzbare Information mehr zu entnehmen ist. Je kleiner die Abtastperiode, desto genauer wird das kontinuierliche Signal abgebildet, allerdings steigt dadurch die Anzahl der Daten und somit der notwendige Speicherbedarf und Rechenaufwand. Gesucht ist also zu einem gegebenen kontinuierlichen Signal eine möglichst große Abtastperiode, mit der trotz Abtastung möglichst alle relevanten Informationen erhalten bleiben.

Jedes stetige kontinuierliche Signal lässt sich als Summe von periodischen Signalen unterschiedlicher Frequenz darstellen (siehe Kap. 4). Ein Signal heißt *bandbegrenzt*, wenn die höchste vorkommende Frequenz  $f_{\max}$  endlich ist, so dass für das Fourier-Spektrum gilt  $|x(j2\pi f)| = 0$  für  $f > f_{\max}$ . Wenn ein solches Signal mit einer Abtastperiode von weniger als  $T_s = 1/(2 \cdot f_{\max})$  bzw. einer Abtastfrequenz von mehr als  $f_s = 2 \cdot f_{\max}$  abgetastet wird, so lässt sich das Originalsignal aus der (unendlichen) Zeitreihe verlustfrei rekonstruieren (Satz von Shannon). Die Bedingung  $T \leq T_s$  oder  $f \geq f_s$  (mit  $f = 1/T$ ) wird *Nyquist-Bedingung* genannt. Abbildung 2.6 zeigt drei Zeitreihen, die mit unterschiedlichen Abtastperioden aus dem Originalsignal  $x(t)$  in Abb. 2.5 (links) gewonnen wurden. Für die linke Zeitreihe gilt  $T < 1/(2 \cdot f_{\max})$ , die Nyquist-Bedingung ist also erfüllt, und der aus der Zeitreihe entstandene Polygonzug stimmt gut mit der Kurve  $x(t)$  überein. Für die mittlere Zeitreihe gilt  $T = 1/(2 \cdot f_{\max})$ , die Abtastperiode ist also an der Grenze der Nyquist-Bedingung, und der entsprechende Polygonzug gibt die Charakteristika von  $x(t)$  hinreichend gut wieder. Bei einer Verschiebung der Abtastzeitpunkte um eine halbe Periodenlänge bei gleicher Abtastperiode würden allerdings alle Abtastwerte in die Nulldurchgänge fallen und eine (informationslose) Zeitreihe mit Nullwerten liefern. Es ist also empfehlenswert, eine Abtastperiode etwas unterhalb der Nyquist-Bedingung zu wählen. Für die rechte Zeitreihe gilt  $T > 1/(2 \cdot f_{\max})$ , das heißt die Nyquist-Bedingung ist nicht erfüllt, und der entsprechende Polygonzug stimmt nur sehr schlecht mit  $x(t)$  überein. Häufig ist die Abtastung zum Zeitpunkt der Datenanalyse bereits erfolgt und kein



**Abb. 2.7** Quantisierte Zeitreihe und Quantisierungsfehler

Zugriff auf die Originalsignale mehr möglich, so dass nicht überprüft werden kann, ob die Nyquist-Bedingung eingehalten wurde. Es ist oft empfehlenswert, diesen Aspekt mit den Datenlieferanten zu besprechen.

Der erste Teil dieses Abschnitts behandelte die Diskretisierung im Zeitbereich, die sogenannte Abtastung. Im folgenden Teil wird die Diskretisierung im Wertebereich betrachtet, die sogenannte *Quantisierung*. Quantisierung wird angewendet auf analoge Werte, die mit endlicher Auflösung in digitale Werte übersetzt werden, sowie auf digitale Werte, deren Auflösung reduziert wird, um Speicherplatz zu sparen oder um die Übertragungsgeschwindigkeit zu erhöhen. Die Quantisierung analoger Werte bildet das kontinuierliche Intervall  $[x_{\min}, x_{\max}]$  auf die Menge der diskreten Werte  $\{x_1, \dots, x_q\}$  ab, wobei  $q$  die Anzahl der Quantisierungsstufen bezeichnet. Die Quantisierungsstufen  $x_1, \dots, x_q$  sind nicht mit den Merkmalswerten von Zeitreihen oder von verschiedenen Objekten zu verwechseln. Jeder quantisierte Wert kann beispielsweise als Binärzahl mit  $b = \lceil \log_2 q \rceil$  bit dargestellt werden. Zu jedem kontinuierlichen Wert  $x \in [x_{\min}, x_{\max}]$  kann ein entsprechender quantisierter Wert  $x_k \in \{x_1, \dots, x_q\}$  bzw. ein Index  $k \in \{1, \dots, q\}$  durch „kaufmännisches Runden“ bestimmt werden.

$$\frac{\frac{x_{k-1}+x_k}{2} - x_1}{x_q - x_1} \leq \frac{x - x_{\min}}{x_{\max} - x_{\min}} < \frac{\frac{x_k+x_{k+1}}{2} - x_1}{x_q - x_1} \quad (2.41)$$

Die Quantisierung führt in der Regel zu einem Quantisierungsfehler. Abbildung 2.7 (links) zeigt die Quantisierung  $x_q(t)$  der Zeitreihe aus Abb. 2.5 (links) für  $q = 11$  äquidistante Quantisierungsstufen  $\{-1, -0.8, \dots, 1\}$ . Die Quantisierungsstufen erscheinen als waagerechte Linien (Treppenstufen). Abbildung 2.7 (rechts) zeigt den Quantisierungsfehler  $e(t) = x(t) - x_q(t)$ . Die Quantisierungsstufen sind hier äquidistant,  $x_i = x_1 + (i - 1) \cdot \Delta x$ ,  $i = 1, \dots, q$ ,  $\Delta x = (x_q - x_1)/(q - 1)$ , also schwankt der Quantisierungsfehler im Intervall  $e(t) \in [-\Delta x/2, \Delta x/2]$ . Durch die Quantisierung entsteht also ein additiver Fehler

mit dem maximalen Betrag  $|e| \leq \Delta x/2$ . Um den Quantisierungsfehler möglichst klein zu halten, sollten die Quantisierungsstufen möglichst fein sein, das heißt  $\Delta x$  möglichst klein bzw.  $q$  möglichst groß.

Eine Binärzahl  $b$  bit kann ganze Zahlen zwischen  $x_1 = 0$  und  $x_q = 2^b - 1$  darstellen. Der relative Quantisierungsfehler lässt sich dann mit  $|e/(x_q - x_1)| \leq 100\%/(2^b - 1) \approx 100\%/2^{b+1}$  abschätzen. Für typische Werte von  $b \geq 8$  kann dieser Quantisierungsfehler meist vernachlässigt werden, falls die Grenzen  $x_{\min}$  und  $x_{\max}$  geeignet gewählt werden. Falls der Wertebereich  $x_{\max} - x_{\min}$  jedoch deutlich höher gewählt wurde als die eigentliche Variation der Daten, so können die Daten konstant (und daher irrelevant) erscheinen, und an den Quantisierungsgrenzen können große Sprünge auftreten, die die Information verfälschen. Wird zum Beispiel die Raumtemperatur mit 8 bit und den Grenzen  $[x_{\min}, x_{\max}] = [0^\circ \text{C}, 1000^\circ \text{C}]$  dargestellt, dann entsprechen Temperaturen zwischen etwa  $13.72^\circ \text{C}$  und  $17.65^\circ \text{C}$  dem Wert 4 und Temperaturen zwischen etwa  $17.65^\circ \text{C}$  und  $21.57^\circ \text{C}$  dem Wert 5. Bei normalen Raumtemperaturverhältnissen treten also nur diese beiden Werte 4 und 5 auf. Die einzige in den quantisierten Werten enthaltene Information ist dann die Unter- oder Überschreitung der recht willkürlichen Temperaturgrenze  $17.65^\circ \text{C}$ .

### Übungsaufgaben

**2.1.** In einem Kuchenrezept steht „Den **Kuchen 45** min bei **180° C** backen“. Geben Sie die Skalen der fettgedruckten Daten an.

**2.2.** Welche Werte liefert das geeignete statistische Maß für  $X = \{1, 2, 3, 4, 4\}$ , wenn die Daten a) nominalskaliert, b) ordinalskaliert, c) intervallskaliert sind?

**2.3.** Berechnen Sie den a) Euklidischen, b) City-Block-, c) Hamming- und d) Edit-Abstand zwischen  $(0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1)$  und  $(1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0)$ .

**2.4.** Welche der folgenden Funktionen sind Ähnlichkeits- oder Unähnlichkeitsmaße?

$$f(x, y) = \begin{cases} 1 & \text{falls } x = y \\ 0 & \text{sonst} \end{cases} \quad (2.42)$$

$$f(x, y) = \frac{x \cdot y^T}{\sqrt{x \cdot x^T \cdot y \cdot y^T}} \quad (2.43)$$

$$f(x, y) = \tanh \left( (x - y) \cdot (x - y)^T \right) \quad (2.44)$$

$$f(x, y) = \cos \left( (x - y) \cdot (x - y)^T \right) \quad (2.45)$$

## Literatur

1. E. Anderson. The Irises of the Gaspe Peninsula. *Bull. of the American Iris Society*, 59:2–5, 1935.
2. J. C. Bezdek, J. M. Keller, R. Krishnapuram, L. I. Kuncheva, and N. R. Pal. Will the real Iris data please stand up? *IEEE Transactions on Fuzzy Systems*, 7(3):368–369, 1999.
3. M. Blum, R. W. Floyd, V. Pratt, R. Rivest, and R. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7:488–461, 1973.
4. R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.
5. R. W. Hamming. Error detecting and error correcting codes. *The Bell System Technical Journal*, 26(2):147–160, April 1950.
6. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.
7. S. S. Stevens. On the theory of scales of measurement. *Science*, 103(2684):677–680, 1946.

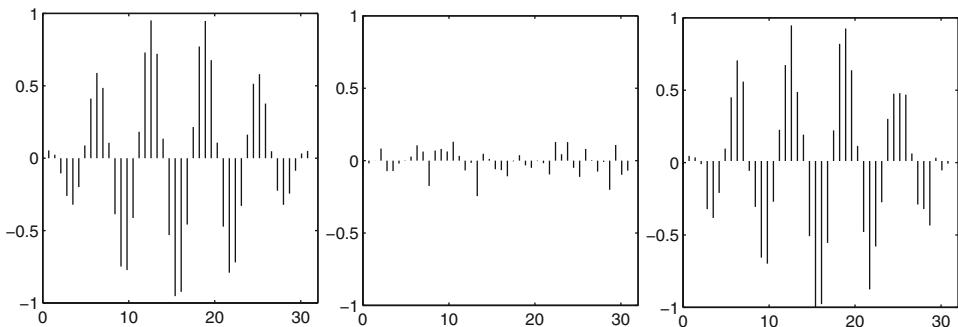
## Zusammenfassung

Daten aus realen Anwendungen enthalten meist Fehler und Rauschen, müssen skaliert und transformiert werden, und müssen oft aus unterschiedlichen und möglicherweise heterogenen Informationsquellen integriert werden. Fehler in Daten können systematischer oder zufälliger Natur sein. Systematische Fehler lassen sich oft korrigieren. Ausreißer sollten erkannt, entfernt oder korrigiert werden. Ausreißer und Rauschen können durch Filtern reduziert werden. Es wird ein Überblick über verschiedene Filtermethoden mit unterschiedlichen Eigenschaften und Komplexitäten gegeben: gleitende Maße und diskrete lineare Filter mit endlicher oder unendlicher Impulsantwort. Merkmale mit unterschiedlichen Wertebereichen werden meist standardisiert oder transformiert.

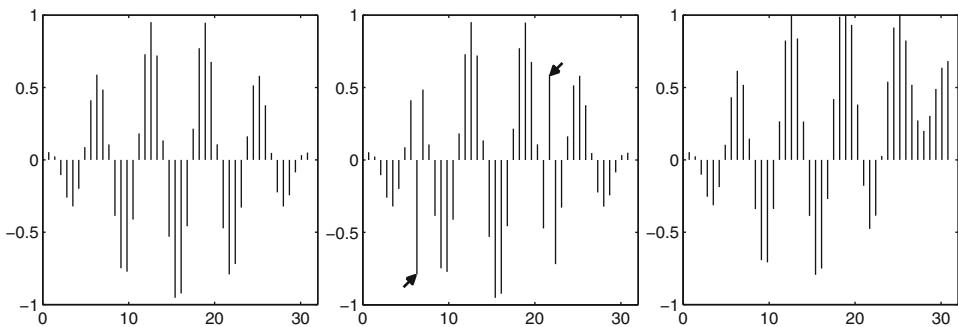
---

## 3.1 Fehlerarten

Daten enthalten oft Fehler, die die Analyseergebnisse verfälschen können. Fehler in Daten oder Messwerten lassen sich unterteilen in *zufällige* und *systematische* Fehler. Zu den zufälligen Fehlern gehören zum Beispiel Mess- und Übertragungsfehler. Diese können als *additives Rauschen* modelliert werden. Abbildung 3.1 (links) zeigt den aus Abb. 2.5 (rechts) bekannten Datensatz. Um einen entsprechenden verrauschten Datensatz nachzubilden, wird mit einem (Quasi-)Zufallsgenerator Gaußsches Rauschen mit Mittelwert 0 und Standardabweichung 0.1 (also mit einer sogenannten  $N(0, 0.1)$ -Verteilung) erzeugt und zu den Daten addiert. Abbildung 3.1 (Mitte) zeigt ein Beispiel für solche Rauschdaten, und Abb. 3.1 (rechts) zeigt die Daten, die entstehen, wenn zu den Originaldaten (links) die Rauschdaten (Mitte) addiert werden. Die Originaldaten und die verrauschten Daten erscheinen sehr ähnlich. Die Verfälschung der Analyseergebnisse durch moderates Rauschen kann oft vernachlässigt werden.



**Abb. 3.1** Originaldaten, Gaußsches Rauschen und verrauschte Daten



**Abb. 3.2** Originaldaten, Ausreißer und Drift

Eine andere Art von Fehlern sind *Ausreißer*. Ausreißer sind einzelne Daten, die stark von den übrigen Daten abweichen. Sie können durch zufällige oder systematische Effekte verursacht sein, z. B. durch ungewöhnlich große Messfehler oder Paketverluste in der Datenübertragung. Bei der manuellen Datenerfassung können Ausreißer durch Eintragung in falsche Felder oder durch Schreibfehler verursacht werden, z. B. durch falsche Position von Dezimaltrennzeichen. Fehlerhafte Dezimaltrennzeichen können auch deterministisch sein, z. B. wenn Daten zwischen Systemen ausgetauscht werden, die unterschiedliche Dezimaltrennzeichen . und , verwenden, wodurch beispielsweise  $1.234 \cdot 10^3$  in  $1,234$  (also  $1.234 \cdot 10^0$ ) verwandelt und somit um einen Faktor von 1000 verfälscht wird.

Systematische Fehler können durch fehlerhafte Formeln bei der Berechnung von Merkmalen verursacht werden, durch falsche Kalibrierung von Messgeräten, durch falsche Skalierung oder durch Drifteffekte. Solche systematischen Fehler können korrigiert werden, falls die Systematik der Fehler bekannt ist.

Abbildung 3.2 zeigt erneut den schon oben verwendeten Datensatz  $X$  (links), dessen Verfälschung durch zwei einzelne Ausreißer (Mitte) sowie durch einen systematischen Drifteffekt (rechts). Die beiden Ausreißer in Abb. 3.2 (Mitte) sind kaum zu erkennen (und daher mit Pfeilen markiert), können die Analyseergebnisse aber signifikant verfälschen.

Ausreißerbehandlung ist daher ein sehr wichtiger Schritt in der Datenvorverarbeitung. Im Gegensatz zu den beiden sogenannten *lokalen* Ausreißern in diesem Beispiel können sogenannte *globale* Ausreißer daran erkannt werden, dass sie stark von der Verteilung der übrigen Daten abweichen. Solche globale Ausreißer können z. B. mit der *2-Sigma-Regel* erkannt werden, die einen Wert  $x_k^{(i)}$  als Ausreißer klassifiziert, sobald er vom Mittelwert des Merkmals um mehr als zwei Mal die Standardabweichung („Sigma“) abweicht.

$$x_k^{(i)} \text{ ist Ausreißer} \Leftrightarrow |x_k^{(i)} - \bar{x}^{(i)}| > 2 \cdot s^{(i)} \quad (3.1)$$

wobei  $\bar{x}^{(i)}$  der Mittelwert des  $i$ -ten Merkmals (2.1) und  $s^{(i)}$  die entsprechende Standardabweichung ist.

$$s^{(i)} = \sqrt{\frac{1}{n-1} \sum_{k=1}^n (x_k^{(i)} - \bar{x}^{(i)})^2} = \sqrt{\frac{1}{n-1} \left( \sum_{k=1}^n (x_k^{(i)})^2 - n (\bar{x}^{(i)})^2 \right)} \quad (3.2)$$

Entsprechend zur 2-Sigma-Regel nach (3.1) lässt sich natürlich auch allgemeiner eine  $m$ -Sigma-Regel definieren. Die Ausreißer in Abb. 3.2 (Mitte) können nicht nach dieser Regel erkannt werden, da sie lediglich lokale Ausreißer sind und den Wertebereich der übrigen Daten nicht verlassen. In Zeitreihen können lokale Ausreißer mit differenziellen Regeln erkannt werden, in denen die Änderung eines Wertes zum Folgewert betrachtet wird. Eine Alternative sind Filtermethoden (siehe weiter unten), die (lokale und globale) Ausreißer nicht nur erkennen, sondern auch reduzieren.

Die Ausreißerbehandlung sollte mit Vorsicht durchgeführt werden, denn Datensätze enthalten oft ungewöhnliche aber korrekte Daten („Exoten“), die wertvolle Information tragen und daher nicht entfernt werden sollten. Wurde zum Beispiel in einer Produktionsanlage selten ein Produkt mit einer ungewöhnlich hohen oder niedrigen Qualität produziert, so würden die entsprechenden Produktionsdaten bei der Ausreißerbehandlung möglicherweise entfernt, obwohl sie für die Analyse des Einflusses verschiedener Produktionsparameter auf die Produktqualität sehr wertvoll sein können. Ein anderes Beispiel sind Erdbeben in seismischen Daten. Exoten sind ohne weitere Informationen (z. B. von Domänenexperten) nicht von fehlerhaften Daten unterscheidbar.

*Ungültige* Daten lassen sich daran erkennen, dass ihre Merkmale außerhalb des zulässigen Wertebereichs liegen. Sie können durch Vergleich mit den zulässigen Merkmalsgrenzen  $x_{\min}^{(i)}$  und  $x_{\max}^{(i)}$ ,  $i = 1, \dots, p$ , erkannt werden.

$$x_k^{(i)} \text{ ist ungültig} \Leftrightarrow (x_k^{(i)} < x_{\min}^{(i)}) \vee (x_k^{(i)} > x_{\max}^{(i)}) \quad (3.3)$$

Merkmalsgrenzen können gegeben sein durch das Vorzeichen (Preis, Temperatur, Zeit), den Messbereich des Aufnehmers, die Zeit der Betrachtung oder durch den Bereich physikalisch sinnvoller Werte. *Konstante* Merkmale

$$x_k^{(i)} = x_l^{(i)} \quad \forall k, l = 1, \dots, n \quad (3.4)$$

enthalten keine nutzbare Information, können die Datenanalyse aber verfälschen und sollten daher entfernt werden.

## 3.2 Behandlung fehlerhafter Daten

Fehlerhafte Daten (wie Ausreißer, ungültige oder fehlende Daten) können auf verschiedene Weise gehandhabt werden:

1. Fehlerliste: Die Daten bleiben unverändert, jedoch werden die Indizes der fehlerhaften Daten in einer separaten Liste gespeichert, die in jedem Zugriff auf die Daten berücksichtigt wird.
2. Fehlerwert: Die ungültigen Daten werden durch einen speziellen Fehlerwert  $x_k^{(i)} = \text{NaN}$  ersetzt. NaN steht für engl. *Not a Number* (keine Zahl) und wird z. B. im 64-Bit-IEEE-Fließkommaformat als \$7FFFFFFF kodiert.
3. Korrektur oder Schätzung einzelner Merkmalswerte: Nur einzelne Merkmalswerte werden korrigiert (falls ungültig) oder geschätzt (falls fehlend). Hierfür kommen verschiedene Vorgehensweisen in Frage:
  - a. Ersetzen durch den Mittelwert, Median, Minimum oder Maximum der korrekten Werte des entsprechenden Merkmals.
  - b. Ersetzen durch den nächsten Nachbarn  $x_k^{(i)} = x_j^{(i)}$  aus

$$\|x_j - x_k\|_{-i} = \min_{l \in \{1, \dots, n\}} \|x_l - x_k\|_{-i} \quad (3.5)$$

wobei  $\|\cdot\|_{-i}$  Merkmal  $i$  sowie ungültige oder fehlende Daten ignoriert.

- c. lineare Interpolation bei Zeitreihen mit äquidistanten Zeitschritten

$$x_k^{(i)} = \frac{x_{k-1}^{(i)} + x_{k+1}^{(i)}}{2} \quad (3.6)$$

- d. lineare Interpolation bei Zeitreihen mit nicht äquidistanten Zeitschritten

$$x_k^{(i)} = \frac{x_{k-1}^{(i)} \cdot (t_{k+1} - t_k) + x_{k+1}^{(i)} \cdot (t_k - t_{k-1})}{t_{k+1} - t_{k-1}} \quad (3.7)$$

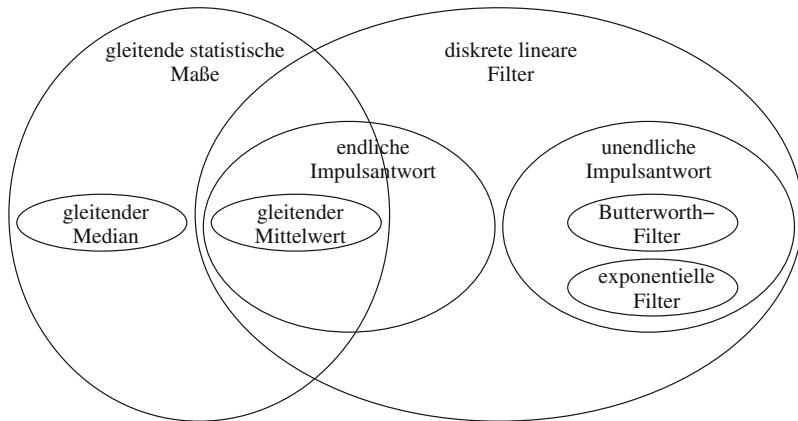
- e. nichtlineare Interpolation, zum Beispiel durch Splines [1]

- f. modellbasierte Schätzung durch Regression (siehe Kap. 6)

- g. Filterung (siehe weiter unten)

4. Entfernen ganzer Merkmalsvektoren: Alle Merkmalsvektoren, die mindestens einen fehlerhaften Merkmalswert enthalten, werden entfernt.
5. Entfernen ganzer Merkmale: Alle Merkmale, die mindestens einen ungültigen Merkmalswert enthalten, werden entfernt.

Bei der Auswahl der geeigneten Methode spielt es eine große Rolle, wie viele Daten insgesamt verfügbar und wie viele davon fehlerhaft sind. Sind nur wenige Daten vorhanden und es ist schwierig, zusätzliche Daten zu beschaffen, dann lohnt sich oft der Aufwand, fehlerhafte Daten zu korrigieren und fehlende Daten zu ergänzen. Sind dagegen ausreichend korrekte Daten verfügbar und soll eine hohe Datenqualität erreicht werden, dann ist es oft sinnvoll, zweifelhafte Merkmalsvektoren oder Merkmale komplett zu entfernen.



**Abb. 3.3** Einige wichtige Familien von Filtern

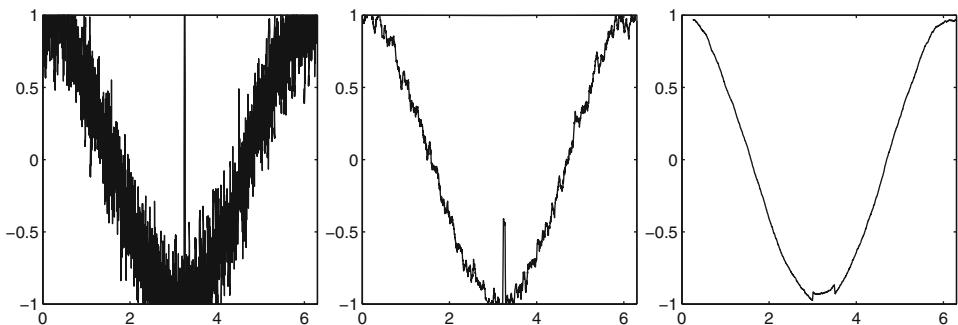
### 3.3 Filterung

Im vorigen Abschnitt wurde beschrieben, wie sich einzelne fehlerhafte Daten korrigieren lassen. Die in diesem Abschnitt behandelten Filtermethoden betrachten Zeitreihendaten und verändern in der Regel *alle* Daten einer Zeitreihe. Ziel ist, nicht nur Ausreißer, sondern auch Rauschen zu entfernen. Abbildung 3.3 zeigt die Familien von Filtermethoden, die hier beschrieben werden.

Zu den verbreiteten Filtermethoden gehören die gleitenden statistischen Maße. Hierbei wird für jeden Wert  $x_k$ ,  $k = 1, \dots, n$ , der Zeitreihe die unmittelbare Nachbarschaft dieses Werts betrachtet und ein statistisches Maß der Daten in dieser Nachbarschaft als gefilterten Wert  $y_k$  ausgegeben. Symmetrische gleitende statistische Maße ungerader Ordnung  $q \in \{3, 5, 7, \dots\}$  betrachten die Nachbarschaftsmengen  $w_{kq} = \{x_{k-(q-1)/2}, \dots, x_{k+(q-1)/2}\}$ , also  $x_k$ , die  $(q-1)/2$  vorangegangenen und die  $(q-1)/2$  nachfolgenden Werte. Symmetrische gleitende statistische Maße sind für die nachträgliche Filterung einsetzbar, wenn die gesamte Zeitreihe bereits vorliegt und im Nachhinein gefiltert werden soll (Offline-Betrieb). Asymmetrische gleitende statistische Maße ganzzahliger Ordnung  $q \in \{2, 3, 4, \dots\}$  betrachten die Nachbarschaftsmengen  $w_{kq} = \{x_{k-q+1}, \dots, x_k\}$ , also  $x_k$  und die  $q-1$  vorangegangenen Werte. Asymmetrische gleitende statistische Maße sind auch für den Online-Betrieb geeignet und liefern den Ausgabewert  $y_k$ , sobald der Eingabewert  $x_k$  vorliegt.

Als statistisches Maß wird häufig der Mittelwert verwendet. Der symmetrische (3.8) bzw. asymmetrische (3.9) gleitende Mittelwert mit Intervallbreite  $q$  lautet

$$y_k = \frac{1}{q} \sum_{i=k-\frac{q-1}{2}}^{k+\frac{q-1}{2}} x_i \quad (3.8)$$



**Abb. 3.4** Original- und mit gleitendem Mittelwert gefilterte Daten,  $q = 21$  und  $q = 201$

$$y_k = \frac{1}{q} \sum_{i=k-q+1}^k x_i \quad (3.9)$$

Abbildung 3.4 (links) zeigt eine Zeitreihe  $X = \{x_1, \dots, x_n\}$ , die aus einer mit Rauschen und einem starken Ausreißer versetzten Kosinusfunktion gewonnen wurde. Die beiden anderen Diagramme zeigen die mit asymmetrischem gleitendem Mittelwert gefilterten Zeitreihen für  $q = 21$  (Mitte) und  $q = 201$  (rechts). In beiden gefilterten Zeitreihen wurde das Rauschen wesentlich verringert, für  $q = 201$  wurde das Rauschen sogar fast vollständig beseitigt. Die Amplitude des Ausreißers wurde von 2 auf etwa 0.5 ( $q = 21$ ) bzw. 0.1 ( $q = 201$ ) verringert. Eine Erhöhung der Intervallbreite  $q$  führt in der Regel zu einer stärkeren Filterung, allerdings gehen durch die Filterung  $q - 1$  Datenwerte verloren. Die Intervallbreite sollte also deutlich kleiner sein als die Länge der Zeitreihe ( $q \ll n$ ).

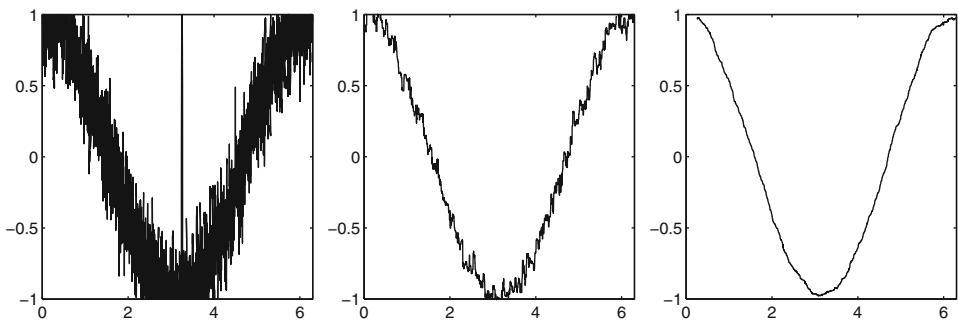
Im vorigen Kapitel wurde der Median als statistisches Maß für ordinal-, intervall- und proportionalskalierte Daten beschrieben. Der symmetrische bzw. asymmetrische gleitende Median mit Intervallbreite  $q$  lautet  $m_{kq} \in w_{kq} = \{x_{k-(q-1)/2}, \dots, x_{k+(q-1)/2}\}$  oder  $w_{kq} = \{x_{k-q+1}, \dots, x_k\}$  mit

$$|\{x_i \in w_{kq} \mid x_i < m_{kq}\}| = |\{x_i \in w_{kq} \mid x_i > m_{kq}\}| \quad (3.10)$$

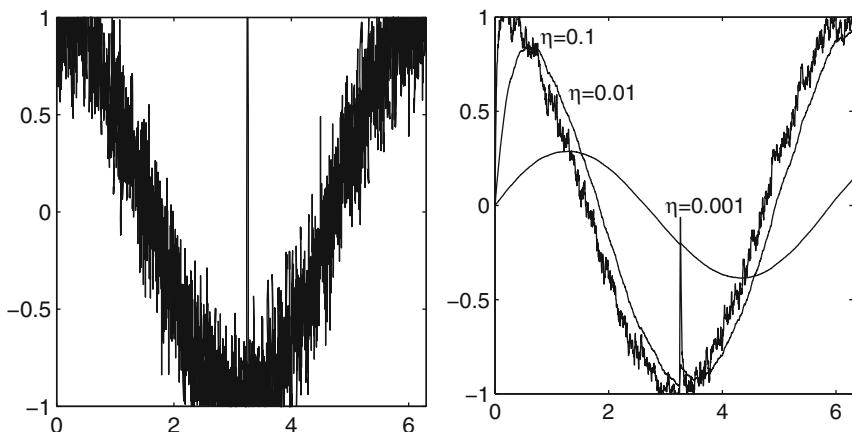
wobei  $|\cdot|$  die Mengenkardinalität (Anzahl der Elemente) bezeichnet. Abbildung 3.5 zeigt das gleiche Experiment wie Abb. 3.4, nur mit gleitendem Median statt gleitendem Mittelwert. Für die gleiche Intervallbreite verringern der gleitende Mittelwert und der gleitende Median das Rauschen etwa gleich gut, der gleitende Median reduziert den Ausreißer jedoch wesentlich besser als der gleitende Mittelwert.

Die dritte Familie der hier dargestellten Filtermethoden ist das *exponentielle Filter*. Ein exponentielles Filter geht von langsamem Änderungen der zu filternden Zeitreihe aus. Daher sollte jeder Wert  $y_k$  des Filterausgangs dem vorhergehenden Wert  $y_{k-1}$  ähnlich sein, bis auf einen Korrekturwert, der als Bruchteil  $\eta \in [0, 1]$  des vorhergehenden Filterfehlers  $x_{k-1} - y_{k-1}$  berechnet wird. Ein exponentielles Filter ist also definiert durch

$$y_k = y_{k-1} + \eta \cdot (x_{k-1} - y_{k-1}), \quad k = 1, \dots, n-1 \quad (3.11)$$

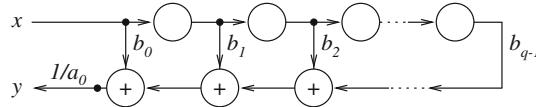


**Abb. 3.5** Original- und mit gleitendem Median gefilterte Daten,  $q = 21$  und  $q = 201$



**Abb. 3.6** Original- und exponentiell gefilterte Daten,  $\eta = 0.1, 0.01$  und  $0.001$

wobei der Filterausgang meist als  $y_0 = 0$  initialisiert wird. In den aktuellen Wert des Filterausgangs  $y_k$  fließen die vorhergehenden Werte des Filterausgangs  $y_{k-i}$ ,  $i = 1, \dots, k-1$ , mit dem Faktor  $(1 - \eta)^i$  ein, d.h. das Filter zeigt ein exponentielles Vergessen der vorangegangenen Werte des Filterausgangs, daher der Name „exponentielles Filter“. Für  $\eta = 0$  verharrt das exponentielle Filter auf dem Initialwert  $y_k = y_0 = 0$ . Für  $\eta = 1$  reproduziert es einfach den vorangegangenen Wert des Filtereingangs  $y_k = x_{k-1}$ . Abbildung 3.6 (rechts) zeigt die Ausgabe von exponentiellen Filtern für den Beispieldatensatz aus Abb. 3.4 und 3.5 (links) für die Parameterwerte  $\eta = 0.1, 0.01$  und  $0.001$ . Je niedriger der Wert von  $\eta$ , desto stärker wird das Rauschen reduziert. Der Ausreißer wird deutlich weniger unterdrückt als mit einem Median-Filter. Für kleine Werte von  $\eta$  ergibt sich ein zeitlicher Versatz zwischen Ein- und Ausgangsdaten. Für  $\eta = 0.001$  kann der Filterausgang dem Filtereingang nicht mehr folgen. Bei einem exponentiellen Filter muss der Wert des Parameters  $\eta$  also sorgfältig gewählt werden. Er muss klein genug sein, um einen



**Abb. 3.7** Datenflussgraph eines FIR-Filters

ausreichenden Filtereffekt zu erzielen, muss aber groß genug sein, um die wesentlichen Charakteristika der Originaldaten zu erhalten.

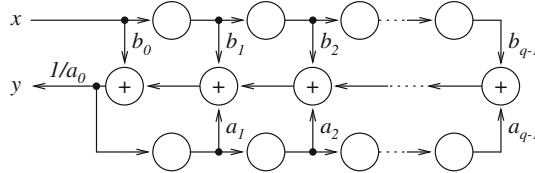
Der gleitende Mittelwert und das exponentielle Filter sind Sonderfälle des allgemeinen Familien der *diskreten linearen Filter*. Ein asymmetrisches diskretes lineares Filter der Ordnung  $q = 1, 2, \dots$  wird beschrieben durch

$$\sum_{i=0}^{q-1} a_i \cdot y_{k-i} = \sum_{i=0}^{q-1} b_i \cdot x_{k-i} \quad (3.12)$$

mit den Filterkoeffizienten  $a_0, \dots, a_{q-1}, b_0, \dots, b_{q-1} \in \mathbb{R}$  [3]. Durch Umformung erhalten wir die Ausgabe eines asymmetrischen diskreten linearen Filters als

$$y_k = \sum_{i=0}^{q-1} \frac{b_i}{a_0} \cdot x_{k-i} - \sum_{i=1}^{q-1} \frac{a_i}{a_0} \cdot y_{k-i} \quad (3.13)$$

Zur Vereinfachung betrachten wir hier nur den asymmetrischen Fall. Durch Änderung der Indizierung lassen sich die Gleichungen für ein symmetrisches lineares Filter leicht herleiten. Die Eigenschaften eines diskreten linearen Filters werden durch die Koeffizientenvektoren  $a = (a_0, \dots, a_{q-1})$  und  $b = (b_0, \dots, b_{q-1})$  bestimmt, wobei  $a_0 \neq 0$ . Für  $a_1 = \dots = a_{q-1} = 0$  ist die Filterausgabe  $y_k$  nur von den Eingaben  $x_{k-q+1}, \dots, x_k$  abhängig und ist unabhängig von den vorhergehenden Ausgaben  $y_{k-q+1}, \dots, y_{k-1}$ , so dass eine Änderung am Eingang  $x_k$  nur den aktuellen Wert des Ausgangs  $y_k$  und die Folgewerte  $y_{k+1}, \dots, y_{k+q-1}$  beeinflusst und dann vollständig vergessen wird. Für  $a_1 = \dots = a_{q-1} = 0$  sprechen wir daher von einer *endlichen Impulsantwort*, engl. *Finite Impulse Response (FIR)*. Andernfalls sprechen wir von einer *unendlichen Impulsantwort*, engl. *Infinite Impulse Response (IIR)*. Abbildung 3.7 zeigt den Datenflussgraphen eines FIR-Filters. Jeder Kreis mit einem Pluszeichen (+) entspricht einer Addition der Knoteneingangs-werte, jedes Kantengewicht entspricht einer Multiplikation mit diesem Gewichtswert, und jeder leere Kreis entspricht einer Zeitverzögerung von einem Schritt, d.h. in jedem Schritt wird der aktuelle Wert gespeichert und im nächsten Schritt wieder ausgegeben, ein sogenanntes *Register*. Der Datenflussgraph eines FIR-Filters beschreibt eine Sequenz von  $q$  Phasen, in denen jeweils eine Multiplikation, eine Addition und eine Speicherung durchgeführt wird. Abbildung 3.8 zeigt den Datenflussgraphen eines IIR-Filters. In jeder Phase des IIR-Filters werden zwei Multiplikationen, eine Addition von drei Summanden und die Speicherung von zwei Werten durchgeführt. Sogenannte *Signalprozessoren* sind spezialisierte Hardware-Einheiten, die einzelne FIR- oder IIR-Phasen in wenigen Zeitschritten durchführen können und somit sehr schnelles Filtern ermöglichen.



**Abb. 3.8** Datenflussgraph eines IIR-Filters

Der gleitende Mittelwert und das exponentielle Filter sind Sonderfälle von diskreten linearen Filtern. Durch Einsetzen der Koeffizientenvektoren  $a = (1, \eta - 1)$  und  $b = (0, \eta)$  in (3.13) erhalten wir

$$y_k = \eta \cdot x_{k-1} - (\eta - 1) \cdot y_{k-1} = y_{k-1} + \eta(x_{k-1} - y_{k-1}) \quad (3.14)$$

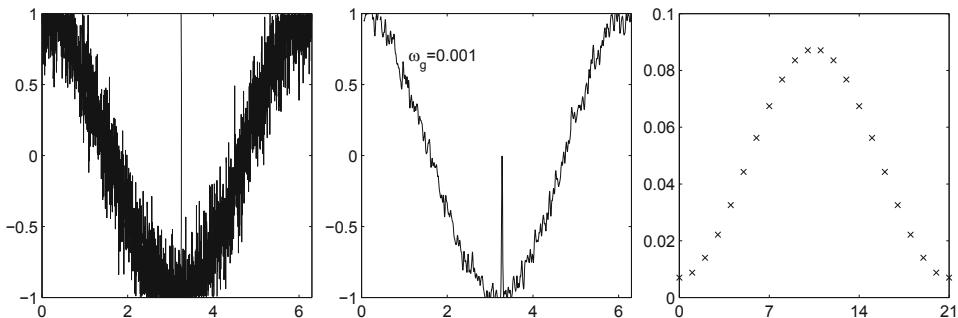
was der Gleichung des exponentiellen Filters (3.11) entspricht. Für  $\eta < 1$  ist  $a_2 > 0$  und das exponentielle Filter daher ein IIR-Filter. Durch Einsetzen der Koeffizientenvektoren  $a = (1)$  und  $b = (\underbrace{\frac{1}{q}, \dots, \frac{1}{q}}_{q \text{ mal}})$  in (3.13) erhalten wir

$$y_k = \sum_{i=0}^{q-1} \frac{1}{q} \cdot x_{k-i} = \frac{1}{q} \sum_{i=k-q+1}^k x_i \quad (3.15)$$

was der Gleichung des (asymmetrischen) gleitenden Mittelwerts (3.9) entspricht. Der gleitende Mittelwert ist also ein FIR-Filter. Der gleitende Mittelwert der Ordnung  $q = 2$  wird auch *FIR-Tiefpass erster Ordnung* genannt, was darauf hindeutet, dass bevorzugt Signalbestandteile mit tiefen Frequenzen durchgelassen werden. Es wurden viele andere Arten von Tiefpassen, Hochpassen und anderen Filtertypen entwickelt. Zur Entwicklung von FIR- und IIR-Filters mit bestimmten Eigenschaften werden Tafeln von Filterkoeffizienten sowie spezielle Software-Werkzeuge zum Filterentwurf eingesetzt.

Abbildung 3.9 zeigt den Beispieldatensatz aus Abb. 3.4–3.6 (links), die Ausgabe eines FIR-Tiefpasses der Ordnung 21 (Mitte), sowie die entsprechenden Filterparameter  $b_0, \dots, b_{21}$  für  $a = (1)$  (rechts). Die Filterausgabe ähnelt stark der des gleitenden Mittelwerts in Abb. 3.4 (Mitte). Alle Filterkoeffizienten  $b$  sind positiv und liegen auf einer symmetrischen glockenförmigen Kurve. Den größten Wert hat der mittlere Koeffizient  $b_{10}$ , und die Summe der Koeffizienten beträgt  $b_0 + \dots + b_{21} = 1$ .

Eine bekannte Familie von FIR/IIR-Filters sind die sogenannten *Butterworth-Filter* [2]. Ein Butterworth-Tiefpass erster Ordnung mit Grenzfrequenz  $\omega_g = 0.5$  hat die Filterkoeffizienten  $a = (1)$  und  $b = (0.5, 0.5)$ , ist also ein FIR-Filter und entspricht einem Tiefpassfilter erster Ordnung bzw. dem gleitenden Mittelwert zweiter Ordnung. Tabelle 3.1 zeigt die Filterkoeffizienten von Butterworth-Tiefpassen zweiter Ordnung für die Grenzfrequenzen  $\omega_g = 0.01, 0.003$  und  $0.001$ . Offensichtlich handelt es sich hier jeweils um



**Abb. 3.9** Originaldaten, mit FIR-Tiefpass der Ordnung 21 gefilterte Daten, sowie Koeffizienten dieses FIR-Filters

**Tab. 3.1** Filterkoeffizienten für Butterworth-Tiefpässe zweiter Ordnung mit verschiedenen Grenzfrequenzen

$\omega_g$	$a_0$	$a_1$	$a_2$	$b_0$	$b_1$	$b_2$
0.01	1	-1.96	0.957	$2.41 \cdot 10^{-4}$	$4.83 \cdot 10^{-4}$	$2.41 \cdot 10^{-4}$
0.003	1	-1.99	0.987	$2.21 \cdot 10^{-5}$	$4.41 \cdot 10^{-5}$	$2.21 \cdot 10^{-5}$
0.001	1	-2	0.996	$2.46 \cdot 10^{-6}$	$4.92 \cdot 10^{-6}$	$2.46 \cdot 10^{-6}$

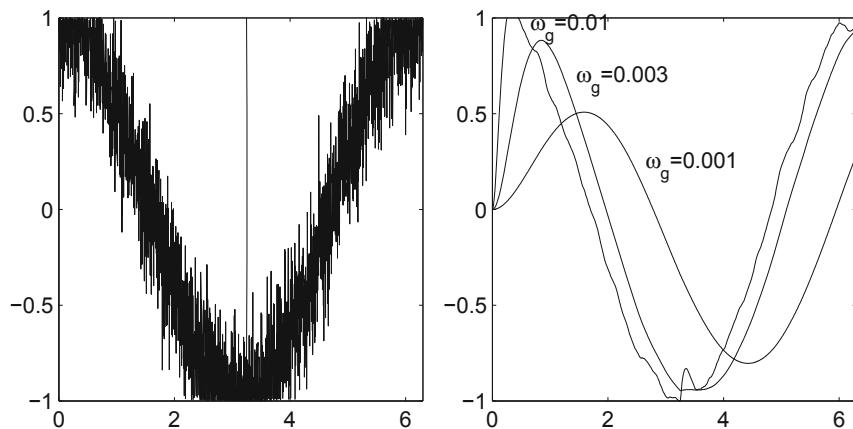
IIR-Filter. Alle Filterkoeffizienten sind positiv, und es gilt  $2 \cdot b_0 = b_1 = 2 \cdot b_2$ . Für  $\omega_g \rightarrow 0$  erhalten wir  $a = (1, -2, 1)$  und  $b = (0, 0, 0)$ . Dies entspricht einem autoregressiven System (siehe Abschn. 7.3) mit

$$y_k = -2y_{k-1} + y_{k-2} \quad (3.16)$$

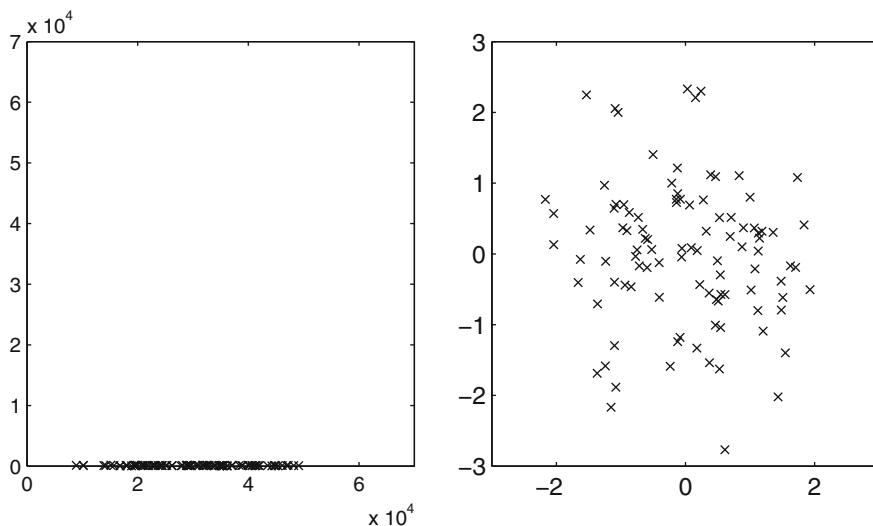
das für die Initialisierung  $y_0 = y_1 = 0$  konstant null liefert und für alle anderen Initialisierungen instabil wird. Abbildung 3.10 zeigt die Ausgabe der Butterworth-Tiefpässe zweiter Ordnung mit den Parametern aus Tab. 3.1. Für  $\omega_g = 0.01$  werden Rauschen und Ausreißer stark reduziert. Kleinere Grenzfrequenzen verursachen einen ähnlichen Effekt, den wir bereits beim exponentiellen Filter beobachtet haben (Abb. 3.6): Der Filterausgang wird mit kleinerem  $\omega_g$  immer stärker verzögert und wird für  $\omega_g \rightarrow 0$  zu null. Diese IIR-Tiefpässe benötigen nur 6 Parameter im Gegensatz zu den 21 Parametern der FIR-Tiefpässe aus Abb. 3.9. Im Vergleich zu FIR-Filters brauchen gleichwertige IIR-Filter in der Regel weniger Parameter und einen geringeren Rechenaufwand, sind aber weniger robust bezüglich Änderungen der Filterparameter und können instabil werden.

### 3.4 Datentransformationen

Die Wertebereiche von Merkmalen können sehr unterschiedlich sein. Werden beispielsweise Autos betrachtet, so unterscheiden sich die Preise in Euro von der Motorleistung in PS um mehrere Größenordnungen. Solche stark unterschiedlichen Wertebereiche können



**Abb. 3.10** Originaldaten und mit Butterworth-Tiefpass zweiter Ordnung gefilterte Daten ( $\omega_g = 0.01, 0.003$  und  $0.001$ )



**Abb. 3.11** Zwei Merkmale mit unterschiedlichen Wertebereichen: Originaldaten und standardisierte Daten

die Analyseresultate verfälschen. Beispielsweise wird der Euklidische Abstand zwischen zwei Merkmalsvektoren dann fast ausschließlich durch das größere Merkmal bestimmt. Abbildung 3.11 (links) zeigt einen Datensatz, der aus einer zweidimensionalen Gauß-Verteilung mit Mittelwert  $\bar{x} = (30000, 100)$  und Standardabweichung  $s = (9000, 30)$ , erzeugt wurde. Die beiden Merkmale sind unkorreliert und unterscheiden sich um den Faktor 300. Werden beide Koordinatenachsen gleich skaliert, so erscheint der Datensatz als horizontale Linie an der horizontalen Koordinatenachse, es ist also nur das erste (größte)

Merkmal sichtbar. Eine solche Visualisierung kann sinnvoll sein, wenn es sich um vergleichbare Merkmale handelt, beispielsweise um Längen und Breiten von Stahlröhren in Millimetern. Eine solche Visualisierung ist jedoch ungeeignet für den Vergleich von Merkmalen mit ähnlicher Wichtigkeit, wie beispielsweise in obigen Beispiel der Merkmale von Autos (Preise in Euro und Motorleistung in PS). Die Analyseergebnisse sollten auch nicht von der Wahl der Einheit der Merkmale abhängen. In unserem Beispiel sollte es keine Rolle spielen, ob die Motorleistung in PS, Kilowatt oder Watt angegeben wird. Es ist daher oft sinnvoll, die Merkmale in ähnliche Wertebereiche zu transformieren. Eine solche Transformation heißt *Standardisierung*. Der *minimale Hyperwürfel* eines  $p$ -dimensionalen Datensatzes  $X$  ist definiert als

$$H(X) = [\min\{X^{(1)}\}, \max\{X^{(1)}\}] \times \dots \times [\min\{X^{(p)}\}, \max\{X^{(p)}\}] \quad (3.17)$$

$H(X)$  enthält also alle Punkte in  $X$ , also  $X \subseteq H$ . Falls  $X$  Ausreißer enthält, oder falls der eigentlich relevante Merkmalsraum nur einen kleinen Teil von  $H(X)$  einnimmt, ist es sinnvoller statt des *beobachteten minimalen Hyperwürfels*  $H(X)$  den *relevanten Hyperwürfel*

$$H^*(X) = [x_{\min}^{(1)}, x_{\max}^{(1)}] \times \dots \times [x_{\min}^{(p)}, x_{\max}^{(p)}] \quad (3.18)$$

zu betrachten, bei dem  $x_{\min}$  und  $x_{\max}$  beliebig festgelegt werden können. Als Spezialfall des relevanten Hyperwürfels  $H^*(X)$  erhalten wir den beobachteten minimalen Hyperwürfels  $H(X)$  für  $x_{\min}^{(i)} = \min\{X^{(i)}\}$  und  $x_{\max}^{(i)} = \max\{X^{(i)}\}$ . Die *Standardisierung mit Hyperwürfel*

$$y_k^{(i)} = \frac{x_k^{(i)} - x_{\min}^{(i)}}{x_{\max}^{(i)} - x_{\min}^{(i)}} \quad (3.19)$$

bildet  $X$  auf  $Y$  ab, so dass alle Kanten des Hyperwürfels  $H^*(Y)$  die Kantenlänge eins erhalten.

Eine Standardisierung kann auch mit Hilfe statistischer Verteilungsmodelle durchgeführt werden. Die Annahme, dass die Merkmale näherungsweise Gauß-verteilt sind, führt zur sogenannten  $\mu$ - $\sigma$ -Standardisierung

$$y_k^{(i)} = \frac{x_k^{(i)} - \bar{x}^{(i)}}{s^{(i)}} \quad (3.20)$$

Abbildung 3.11 (rechts) zeigt das Ergebnis der  $\mu$ - $\sigma$ -Standardisierung des Datensatzes links. Die standardisierten Daten erscheinen als eine Punktwolke, die die Struktur der Daten möglicherweise besser widerspiegelt als die horizontale Linie links. Dies hängt jedoch von der Semantik der Daten ab, wie oben bereits dargelegt.

Die Standardisierung mit Hyperwürfel eignet sich für näherungsweise gleichförmig verteilte Merkmale. Die  $\mu$ - $\sigma$ -Standardisierung eignet sich für näherungsweise Gauß-verteilte Merkmale. In vielen Fällen können die Merkmale näherungsweise als gleichförmig oder Gauß-verteilt angenommen werden. Manche Merkmale lassen sich aber besser

durch asymmetrische Verteilungen mit oberer oder unterer Grenze modellieren. Zeitdifferenzen zwischen Zufallsereignissen wie z. B. Lieferzeitpunkte folgen beispielsweise in der Regel einer Poisson-Verteilung, die stets positive Werte liefert (untere Grenze null), bei der aber beliebig hohe Werte mit niedriger Wahrscheinlichkeit auftreten können (obere Grenze unendlich). Solche Merkmale würden mit den oben beschriebenen Standardisierungsverfahren nicht geeignet abgebildet. Sie werden daher oft zunächst in eine Verteilung transformiert, die einer Gleich- oder Gauß-Verteilung besser entspricht. Für die Auswahl einer geeigneten Transformation sollte also der beobachtete und der gewünschte Wertebereich betrachtet werden. Im Folgenden wird eine Auswahl häufig verwendeter Transformationen und deren Wertebereiche dargestellt.

- inverse Transformation  $f : \mathbb{R} \setminus \{0\} \rightarrow \mathbb{R} \setminus \{0\}$

$$f(x) = f^{-1}(x) = \frac{1}{x} \quad (3.21)$$

- Wurzeltransformation  $f : (c, \infty) \rightarrow \mathbb{R}^+$

$$f(x) = \sqrt[b]{x - c} \quad (3.22)$$

$$f^{-1}(x) = x^b + c, \quad c \in \mathbb{R}, b > 0 \quad (3.23)$$

- logarithmische Transformation  $f : (c, \infty) \rightarrow \mathbb{R}$

$$f(x) = \log_b(x - c) \quad (3.24)$$

$$f^{-1}(x) = b^x + c, \quad c \in \mathbb{R}, b > 0 \quad (3.25)$$

- Fisher-Z-Transformation  $f : (-1, 1) \rightarrow \mathbb{R}$

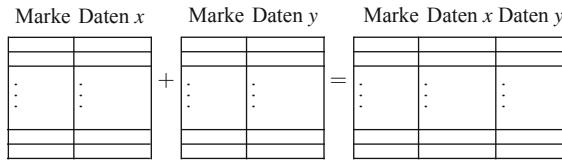
$$f(x) = \operatorname{artanh} x = \frac{1}{2} \cdot \ln \frac{1+x}{1-x} \quad (3.26)$$

$$f^{-1}(x) = \tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (3.27)$$

---

## 3.5 Datenintegration

Häufig sind die für die Analyse relevanten Daten nicht in einem einzelnen Datensatz aus einer einzelnen Datenquelle in einem einzelnen IT-System enthalten, sondern sind über verschiedene Datensätze, Datenquellen und IT-Systeme verteilt. So müssen beispielsweise Preise, Umsatzzahlen, Logistik-Daten und Fertigungsparameter oft aus unterschiedlichen Systemen extrahiert und kombiniert werden. Für die Kombination von Daten aus unterschiedlichen Systemen müssen Merkmalsvektoren aus unterschiedlichen Datensätzen einander zugeordnet werden. Eine solche Zuordnung erfolgt auf Basis von Marken wie beispielsweise Codes, die Personen oder Objekte bezeichnen, (relative) Zeitstempel oder (relative) Ortsangaben. Abbildung 3.12 stellt die markenbasierte Integration von Daten schematisch dar. Bei dieser Integration werden Merkmalsvektoren der gleichen Marke



**Abb. 3.12** Integration von Datensätzen

gesucht und zusammengehängt. Dabei muss berücksichtigt werden, dass die Marken möglicherweise nur näherungsweise übereinstimmen. So werden zwei Zeitstempel 10:59 und 11:00 möglicherweise als gleichwertig betrachtet. Werden zu einer Marke nicht in allen Datensätzen Merkmalsvektoren gefunden, so führt dies zu fehlenden Einträgen im integrierten Datensatz. Oft werden solche unvollständigen Merkmalsvektoren entfernt. Werden zu einer Marke in einem Datensatz mehrere Merkmalsvektoren gefunden, so können im integrierten Datensatz für eine Marke mehrere Einträge erzeugt werden. Oft werden solche mehrfachen Merkmalsvektoren zu einem einzigen Merkmalsvektor verknüpft, z. B. als Mittelwert der mehrdeutigen Merkmalswerte.

Die hier beschriebenen Verfahren zur Datentransformation und Datenintegration bilden die Basis für den sogenannten *ETL-Prozess* der die Schritte Extraktion, Transformation und Laden der Daten umfasst und durch zahlreiche spezielle Software-Werkzeuge unterstützt wird.

### Übungsaufgaben

**3.1.** Sind im Datensatz {0, 0, 0, 1, 0, 0, 0} Rauschen, globale oder lokale Ausreißer enthalten?

**3.2.** Bestimmen Sie die globalen und lokalen Ausreißer in folgenden Datensätzen:

- (a) {1, 1, 1, 1, 4, 2, 2, 2}, (b) {1, 2, 3, 4, 5, 1, 3, 2, 1}, (c) {(2, 9), (1, 9), (2, 1), (2, 8), (1, 7), (1, 8), (2, 7)}.

**3.3.** Berechnen Sie die Ausgabe eines (a) asymmetrischen gleitenden Mittelwert-filters mit  $q = 3$ , (b) asymmetrischen gleitenden Medianfilters mit  $q = 3$ , (c) exponentiellen Filters mit  $y_0 = 0$  und  $\eta = 0.5$  für die Zeitreihe (0, 0, 0, 1, 0, 0, 0). Welches Filterergebnis finden Sie am besten?

**3.4.** Standardisieren Sie den Datensatz  $\{(-1, -10), (1, 10)\}$ .

---

### Literatur

1. B. A. Barsky and D. P. Greenberg. Determining a set of B-spline control vertices to generate an interpolating surface. *Computer Graphics and Image Processing*, 14(3):203–226, November 1980.
2. S. Butterworth. On the theory of filter amplifiers. *Wireless Engineer*, 7:536–541, 1930.
3. A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall, 2009.

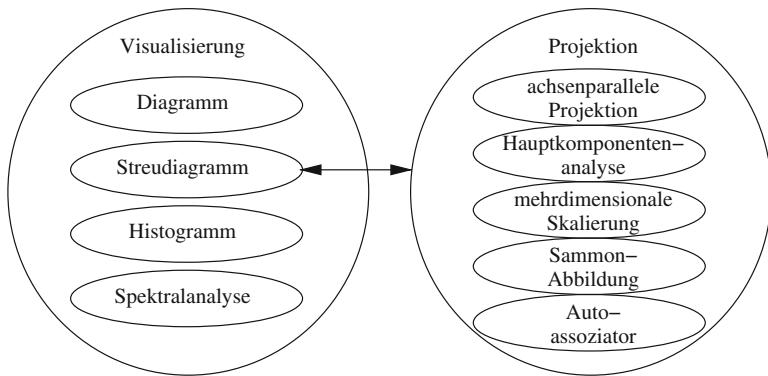
## Zusammenfassung

Visualisierungstechniken lassen sich sehr effektiv zur Datenanalyse einsetzen. Standardmethoden sind Diagramme und Streudiagramme. Zur Visualisierung hochdimensionaler Daten müssen Projektionen durchgeführt werden. Es werden lineare (Hauptkomponentenanalyse, Karhunen-Loëve-Transformation, Singulärwertzerlegung, Eigenvektorprojektion, Hotelling-Transformation, mehrdimensionale Skalierung) und nichtlineare Projektionsmethoden (Sammon-Abbildung, Auto-Assoziator) vorgestellt. Histogrammverfahren erlauben die Schätzung und Visualisierung von Datenverteilungen. Die Spektralanalyse (Kosinus- und Sinustransformation, Amplituden- und Phasenspektren) ermöglicht die Analyse und Visualisierung von periodischen Daten (z. B. Zeitreihen).

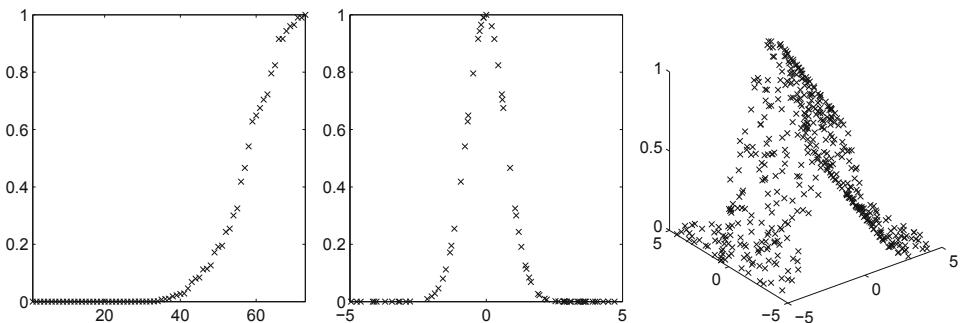
## 4.1 Diagramme

Das menschliche Auge (und das Gehirn) eignen sich hervorragend zur Analyse von Daten. Datenvisualisierung spielt daher in der Datenanalyse eine sehr große Rolle [9, 10]. Visualisierungen sind auch ein wichtiges Hilfsmittel zur Dokumentation und zur Kommunikation von Analyseergebnissen, z. B. für Diskussionen mit Domänenexperten. Auf Papier und Bildschirm lassen sich Daten zweidimensional visualisieren. Höherdimensionale Daten lassen sich mit Projektionsmethoden auf zwei Dimensionen abbilden. Abbildung 4.1 zeigt die in diesem Kapitel vorgestellten Visualisierungs- und Projektionsmethoden.

Bei der zweidimensionalen Visualisierung mit zwei orthogonalen Koordinatenachsen wird jeder Merkmalsvektor als Punkt im Koordinatensystem dargestellt. Die Visualisierung nur eines Merkmals heißt (einfaches) *Diagramm*. Zur Visualisierung von mehreren Merkmalen können *Streudiagramme* verwendet werden. Ein zweidimensionales Streudiagramm bildet jedes der beiden Merkmale auf eine der Koordinatenachsen ab, so dass die

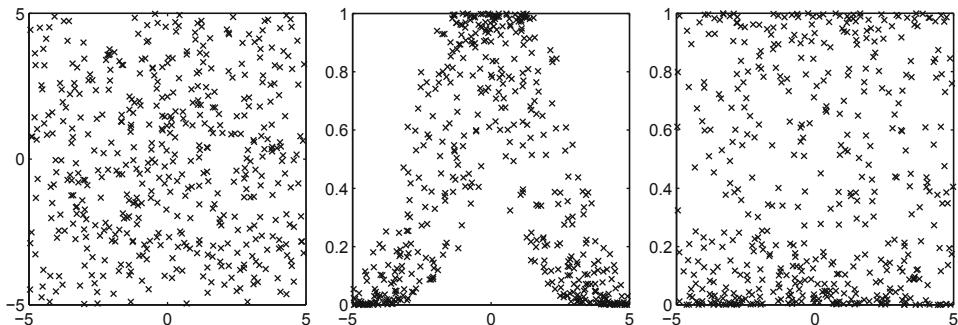


**Abb. 4.1** Visualisierungs- und Projektionsmethoden



**Abb. 4.2** Einfaches Diagramm, zwei- und dreidimensionales Streudiagramm

Merkmalsebene auf die Visualisierungsebene abgebildet wird. Höherdimensionale Streudiagramme projizieren den höherdimensionalen Merkmalsraum auf die zweidimensionale Visualisierungsebene oder verwenden bestimmte visuelle Kennzeichen (geometrische Symbole, Zahlen, Farben oder Grauwerte) zur Darstellung einzelner Merkmale. Dreidimensionale Streudiagramme bilden den dreidimensionalen Merkmalsraum meist mit linearen Projektionsverfahren auf die zweidimensionale Visualisierungsebene ab. Abbildung 4.2 zeigt einen dreidimensionalen Datensatz  $X = \{(x_1, y_1, z_1), \dots, (x_n, y_n, z_n)\}$  als einfaches Diagramm  $X^{(1)} = \{x_1, \dots, x_n\}$  (links), als zweidimensionales Streudiagramm  $X^{(1,2)} = \{(x_1, y_1), \dots, (x_n, y_n)\}$  (Mitte) und als dreidimensionales Streudiagramm  $X^{(1,2,3)} = \{(x_1, y_1, z_1), \dots, (x_n, y_n, z_n)\}$  (rechts). Das einfache Diagramm stellt nur das erste Merkmal dar, das zweidimensionale Diagramm die ersten beiden Merkmale und das dreidimensionale Diagramm alle drei Merkmale. In einem dreidimensionalen Diagramm muss die Projektionsrichtung geeignet gewählt werden. Zu den einfachsten Projektionen gehören achsenparallele Projektionen. In Kap. 6 werden Verfahren zur Merkmalselektion behandelt, die achsenparallele Projektionen darstellen. Abbildung 4.3 zeigt die drei



**Abb. 4.3** Drei zweidimensionale achsenparallele Projektionen

zweidimensionale achsenparallele Projektionen ( $x, y$ ), ( $x, z$ ) und ( $y, z$ ) des obigen dreidimensionalen Datensatzes. Solche achsenparallele Projektionen können einfach durch Weglassen der Merkmale  $x$ ,  $y$  oder  $z$  erreicht werden. Die zweidimensionalen Streudiagramme zeigen dann jeweils die beiden übrigen Merkmale. In diesem Beispiel ist die geometrische Anordnung der drei Merkmale aus den drei achsenparallelen Projektionen nicht gut erkennbar. In den nächsten Abschnitten werden besser geeignete lineare und nichtlineare Projektionsmethoden für solche Daten vorgestellt.

## 4.2 Hauptkomponentenanalyse

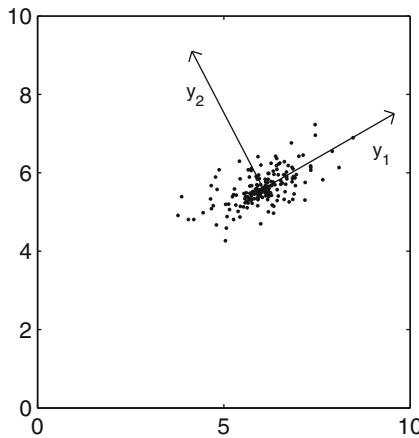
Die *Hauptkomponentenanalyse* (engl. *Principal Component Analysis, PCA*) [4] wird auch *Hauptkomponententransformation*, *Karhunen-Loëve Transformation*, *Singulärwertzerlegung* (engl. *Singular Value Decomposition, SVD*), *Eigenvektorprojektion*, oder *Hotelling-Transformation* genannt. In der Hauptkomponentenanalyse wird eine lineare Projektion der Daten gefunden, die die Datenstruktur so abbildet, dass die Varianz in der niedrigdimensionalen Projektion maximiert wird. Abbildung 4.4 zeigt einen zweidimensionalen Datensatz und zwei Projektionsachsen  $y_1$  und  $y_2$  (bzw. deren Inversen). Von allen linearen Abbildungen dieser Daten führt die Abbildung auf die Achsen  $y_1$  und  $y_2$  zu maximaler Varianz. Die Vektoren  $y_1$  und  $y_2$  heißen *Hauptachsen* oder *Hauptkomponenten* des Datensatzes, daher der Name Hauptkomponentenanalyse.

Die Hauptkomponentenanalyse eines Datensatzes  $X$  führt zu einer lineare Transformation, die sich als Verkettung einer Translation und Rotation darstellen lässt. Diese Transformation nennen wir *Hauptkomponententransformation*.

$$y_k = (x_k - \bar{x}) \cdot E \quad (4.1)$$

Dabei ist  $E$  eine Rotationsmatrix, die von  $X$  abhängt. Die entsprechende inverse Transformation lautet

$$x_k = y_k \cdot E^T + \bar{x} \quad (4.2)$$



**Abb. 4.4** Hauptkomponentenanalyse

Zur Bestimmung der Rotationsmatrix  $E$  wird die Varianz in  $Y$  maximiert. Die Varianz in  $Y$  lässt sich schreiben als

$$v_y = \frac{1}{n-1} \sum_{k=1}^n y_k^T y_k \quad (4.3)$$

$$= \frac{1}{n-1} \sum_{k=1}^n ((x_k - \bar{x}) \cdot E)^T \cdot ((x_k - \bar{x}) \cdot E) \quad (4.4)$$

$$= \frac{1}{n-1} \sum_{k=1}^n E^T \cdot (x_k - \bar{x})^T \cdot (x_k - \bar{x}) \cdot E \quad (4.5)$$

$$= E^T \left( \frac{1}{n-1} \sum_{k=1}^n (x_k - \bar{x})^T \cdot (x_k - \bar{x}) \right) \cdot E \quad (4.6)$$

$$= E^T \cdot C \cdot E \quad (4.7)$$

Dabei ist  $C$  die Kovarianzmatrix von  $X$  mit den Elementen

$$c_{ij} = \frac{1}{n-1} \sum_{k=1}^n (x_k^{(i)} - \bar{x}^{(i)})(x_k^{(j)} - \bar{x}^{(j)}), \quad i, j = 1, \dots, p \quad (4.8)$$

Die Transformationsmatrix  $E$  soll lediglich eine Rotation durchführen, aber keine Streckung. Wir fordern daher

$$E^T \cdot E = 1 \quad (4.9)$$

Diese Forderung führt auf ein Optimierungsproblem mit Nebenbedingung, das sich durch Lagrange-Optimierung lösen lässt. Das Verfahren der Lagrange-Optimierung wird im

Anhang ausführlicher erläutert. Zur Maximierung der Varianz (4.7) unter der Nebenbedingung (4.9) verwenden wir die Lagrange-Funktion

$$L = E^T C E - \lambda(E^T E - 1) \quad (4.10)$$

Die notwendige Bedingung für Optima von  $L$  ist

$$\frac{\partial L}{\partial E} = 0 \quad (4.11)$$

$$\Leftrightarrow CE + E^T C - 2\lambda E = 0 \quad (4.12)$$

$$\Leftrightarrow CE = \lambda E \quad (4.13)$$

Gleichung (4.13) beschreibt ein Eigenwertproblem, das zum Beispiel durch Umwandlung in ein homogenes Gleichungssystem gelöst werden kann.

$$(C - \lambda I) \cdot E = 0 \quad (4.14)$$

Die Spalten der Rotationsmatrix  $E$  ergeben sich aus den Eigenvektoren von  $C$ .

$$E = (v_1, \dots, v_p), \quad (v_1, \dots, v_p, \lambda_1, \dots, \lambda_p) = \text{eig } C \quad (4.15)$$

Die Varianzen in  $Y$  entsprechen den Eigenwerten  $\lambda_1, \dots, \lambda_p$  von  $C$ , denn

$$CE = \lambda E \quad \Leftrightarrow \quad \lambda = E^T CE = v_y \quad (4.16)$$

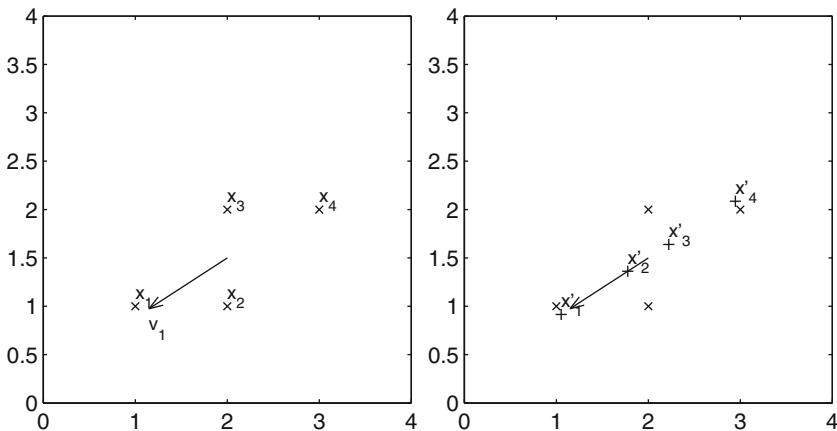
Die Hauptkomponentenanalyse liefert also nicht nur die Koordinatenachsen in  $Y$  und die Transformationsmatrix, die die Varianzen in  $Y$  maximiert, sondern auch die *Werte* dieser Varianzen. Diese können für eine Dimensionsreduktion mit maximaler Varianz genutzt werden. Für eine Abbildung eines Datensatzes  $X \subset \mathbb{R}^p$  auf einen Datensatz  $Y \subset \mathbb{R}^q$  mit  $1 \leq q < p$  betrachten wir die Eigenvektoren mit den  $q$  größten Eigenwerten und fassen diese zu einer Rotations- und Projektionsmatrix  $E$  zusammen.

$$E = (v_1, \dots, v_q) \quad (4.17)$$

Für  $q = 2$  erhalten wir eine Projektion eines hochdimensionalen Datensatzes, die sich in einem zweidimensionalen Streudiagramm darstellen lässt. Häufig wird die Dimensionsreduktion nicht zur Visualisierung verwendet, sondern um die Effizienz nachfolgender Verarbeitungsschritte zu steigern. Um einen hochdimensionalen Datensatz auf eine geeignete Zahl von Merkmalen zu reduzieren, wird oft gefordert, dass die Projektion einen gewissen Anteil der Varianzen abdeckt, beispielsweise 95 %. Um dies zu erreichen, kann  $q$  so gewählt werden, dass gilt

$$\sum_{i=1}^q \lambda_i \left/ \sum_{i=1}^p \lambda_i \right. \geq 95\% \quad (4.18)$$

Für  $p = q$  führt die Hauptkomponententransformation eine Translation und Rotation, jedoch keine Projektion der Daten durch. Für  $q < p$  wird zusätzlich eine Projektion



**Abb. 4.5** Hauptkomponententransformation des 4-Punkte-Datensatzes

durchgefrt, die einen Informationsverlust verursacht. Die Rcktransformation liefert  $x'_k$ ,  $k = 1, \dots, n$ , wobei in der Regel gilt  $x'_k \neq x_k$ . Es kann gezeigt werden, dass der durchschnittliche quadratische Fehler dieser Transformation und Rcktransformation proportional zur Summe der Eigenwerte der weggelassenen Eigenvektoren ist.

$$e = \frac{1}{n} \sum_{k=1}^n \|x_k - x'_k\|^2 = \left(1 - \frac{1}{n}\right) \sum_{i=q+1}^p \lambda_i \quad (4.19)$$

Die Hauptkomponententransformation verwendet die Eigenvektoren mit den hochsten Eigenwerten. Die Hauptkomponententransformation liefert daher nicht nur die lineare Abbildung mit der maximalen Varianz, sondern gleichzeitig auch die lineare Abbildung mit dem kleinsten quadratischen Transformationsfehler. In unserer Herleitung der Hauptkomponententransformation haben wir die Varianz maximiert und eine Transformation erhalten, die den quadratischen Transformationsfehler minimiert. Die Hauptkomponententransformation kann jedoch auch in umgekehrter Reihenfolge hergeleitet werden: Durch Minimierung des quadratischen Transformationsfehlers erhalten wir die Transformation, die die Varianz maximiert.

Wir veranschaulichen die Hauptkomponententransformation anhand eines einfachen Beispiele. Abbildung 4.5 zeigt den Datensatz

$$X = \{(1, 1), (2, 1), (2, 2), (3, 2)\} \quad (4.20)$$

mit Mittelwert (2.1) und Kovarianz (4.8)

$$\bar{x} = \frac{1}{2} \cdot (4, 3) \quad (4.21)$$

$$C = \frac{1}{3} \cdot \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} \quad (4.22)$$

und den Eigenwerten und Eigenvektoren (4.15)

$$\lambda_1 = 0.8727 \quad (4.23)$$

$$\lambda_2 = 0.1273 \quad (4.24)$$

$$v_1 = \begin{pmatrix} -0.85065 \\ -0.52573 \end{pmatrix} \quad (4.25)$$

$$v_2 = \begin{pmatrix} 0.52573 \\ -0.85065 \end{pmatrix} \quad (4.26)$$

Zur Abbildung dieses Datensatzes der Dimension  $p = 2$  auf einen Datensatz der Dimension  $q = 1$  benutzen wir den Eigenvektor  $v_1$  mit dem höchsten Eigenwert  $\lambda_1$  als Rotationsmatrix  $E$ . Diese Projektion erhält 87 % (4.23) der Varianzen und hat einen mittleren quadratischen Projektionsfehler von  $(1 - 1/4) \cdot 12.73 \% \approx 9.5 \%$  (4.19), (4.24). Das neue Koordinatensystem besteht aus nur einer Achse, die in Abb. 4.5 mit dem Vektor  $v_1$  mit Ursprung  $\bar{x}$  dargestellt ist. Mit der Rotations- und Projektionsmatrix

$$E = v_1 = \begin{pmatrix} -0.85065 \\ -0.52573 \end{pmatrix} \quad (4.27)$$

erhalten wir die projizierten Daten (4.1)

$$Y = \{1.1135, 0.2629, -0.2629, -1.1135\} \quad (4.28)$$

Die Rücktransformation (4.2) liefert

$$X' = \{(1.0528, 0.91459), (1.7764, 1.3618), \\ (2.2236, 1.6382), (2.9472, 2.0854)\} \neq X \quad (4.29)$$

Die Projektionen  $X'$  liegen auf der Hauptachse von  $X$ , dem Koordinatenvektor  $v_1$  (Abb. 4.5 rechts). In unserem einfachen Beispiel haben wir lediglich zweidimensionale auf eindimensionale Daten abgebildet. Meist wird die Hauptkomponententransformation verwendet, um höherdimensionale Daten abzubilden, insbesondere zur Projektion auf zwei Dimensionen.

### 4.3 Mehrdimensionale Skalierung

Die *mehrdimensionale Skalierung* (MDS) [11] ist eine lineare Abbildung auf Basis einer Matrixdekomposition. Für eine Datenmatrix  $X \in \mathbb{R}^{n \times p}$  liefert die Eigendekomposition der Produktmatrix  $XX^T$

$$XX^T = Q\Lambda Q^T = (Q\sqrt{\Lambda}^T) \cdot (\sqrt{\Lambda}Q^T) = (Q\sqrt{\Lambda}^T) \cdot (Q\sqrt{\Lambda}^T)^T \quad (4.30)$$

wobei  $Q = (v_1, \dots, v_n)$  die Matrix der Eigenvektoren und  $\Lambda$  die Diagonalmatrix der Eigenwerte von  $XX^T$  ist,  $\Lambda_{ii} = \lambda_i$ ,  $i = 1, \dots, n$ . Auf Basis dieser Eigendekomposition lässt sich  $X$  approximieren durch

$$Y = Q\sqrt{\Lambda}^T \quad (4.31)$$

Für eine Projektion  $Y \subset \mathbb{R}^q$ ,  $q < p$  werden nur die ersten  $q$  Dimensionen verwendet und so skaliert, dass ihre quadratische Norm den zugehörigen Eigenwerten entspricht.

Die mehrdimensionale Skalierung eines Merkmalsdatensatzes  $X$  liefert die gleichen Ergebnisse  $Y$  wie die Hauptachsentransformation. Jedoch lässt sich die mehrdimensionale Skalierung nicht nur zur Abbildung von Merkmalsdatensätzen  $X$  auf  $Y$  verwenden, sondern auch, um eine Merkmalsraumrepräsentation  $Y$  von Objekten zu bestimmen, die durch eine Euklidische Abstandsmatrix  $D$ , also durch relationale Daten beschrieben sind. Hierzu nehmen wir an, dass die Matrix  $D$  aus den paarweisen Euklidischen Abständen eines (unbekannten) Datensatzes  $\tilde{X} \in \mathbb{R}^{n \times p}$  berechnet werden kann. Wir wählen einen beliebigen Punkt  $\tilde{x}_a$ ,  $a \in \{1, \dots, n\}$  als *Ankerpunkt* und transformieren  $\tilde{X}$  nach  $X$  in ein Koordinatensystem mit Ursprung  $\tilde{x}_a$ .

$$x_k = \tilde{x}_k - \tilde{x}_a \quad (4.32)$$

$k = 1, \dots, n$ , und

$$\tilde{x}_i - \tilde{x}_j = x_i - x_j \quad (4.33)$$

$i, j = 1, \dots, n$ . Durch Bildung des Skalarprodukts jeder Seite mit sich selbst erhalten wir

$$(\tilde{x}_i - \tilde{x}_j)(\tilde{x}_i - \tilde{x}_j)^T = (x_i - x_j)(x_i - x_j)^T \quad (4.34)$$

$$\Rightarrow d_{ij}^2 = x_i x_i^T - 2x_i x_j^T + x_j x_j^T = d_{ia}^2 - 2x_i x_j^T + d_{ja}^2 \quad (4.35)$$

$$\Rightarrow x_i x_j^T = (d_{ia}^2 + d_{ja}^2 - d_{ij}^2)/2 \quad (4.36)$$

Damit kann die Produktmatrix  $XX^T$  für die mehrdimensionale Skalierung direkt aus der Euklidischen Abstandsmatrix  $D$  berechnet werden.

Wir veranschaulichen die mehrdimensionale Skalierung anhand des Beispiels (4.20) aus dem vorherigen Abschnitt. Durch Subtraktion des Mittelwerts erhalten wir

$$X = \left\{ (-1, -\frac{1}{2}), (0, -\frac{1}{2}), (0, \frac{1}{2}), (1, \frac{1}{2}) \right\} \quad (4.37)$$

und die Produktmatrix

$$XX^T = \frac{1}{4} \begin{pmatrix} 5 & 1 & -1 & -5 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ -5 & -1 & 1 & 5 \end{pmatrix} \quad (4.38)$$

Der größte Eigenwert und der zugehörige Eigenvektor von  $XX^T$  lauten

$$\lambda_1 \approx 2.618 \quad (4.39)$$

$$v_1 \approx \begin{pmatrix} -0.6882 \\ -0.1625 \\ 0.1625 \\ 0.6882 \end{pmatrix} \quad (4.40)$$

und mit (4.31) erhalten wir schließlich die Projektion

$$Y \approx \begin{pmatrix} -0.6882 \\ -0.1625 \\ 0.1625 \\ 0.6882 \end{pmatrix} \sqrt{2.618} \approx \begin{pmatrix} -1.1135 \\ -0.2629 \\ 0.2629 \\ 1.1135 \end{pmatrix} \quad (4.41)$$

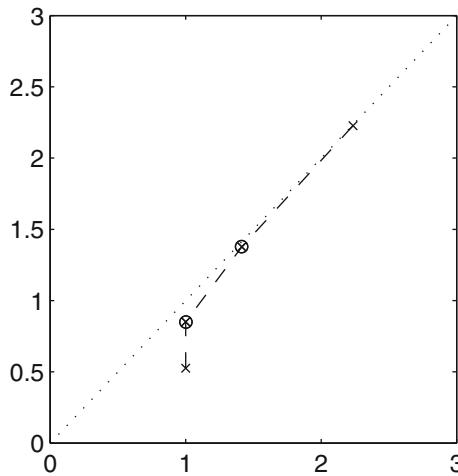
in Übereinstimmung mit den Ergebnissen der Hauptkomponententransformation (4.28).

Die Güte einer Merkmalstransformation lässt sich mit einem sogenannten *Shepard-Diagramm* visualisieren. Ein Shepard-Diagramm ist ein Streudiagramm der paarweisen Abstände  $d_{ij}^x$  und  $d_{ij}^y$  von Originaldaten und projizierten Daten,  $i, j = 1, \dots, n$ . Für unser Beispiel gilt

$$D^x = \begin{pmatrix} 0 & 1 & \sqrt{2} & \sqrt{5} \\ 1 & 0 & 1 & \sqrt{2} \\ \sqrt{2} & 1 & 0 & 1 \\ \sqrt{5} & \sqrt{2} & 1 & 0 \end{pmatrix} \approx \begin{pmatrix} 0 & 1 & 1.4142 & 2.2361 \\ 1 & 0 & 1 & 1.4142 \\ 1.4142 & 1 & 0 & 1 \\ 2.2361 & 1.4142 & 1 & 0 \end{pmatrix} \quad (4.42)$$

$$D^y \approx \begin{pmatrix} 0 & 0.8507 & 1.3764 & 2.2270 \\ 0.8507 & 0 & 0.5257 & 1.3764 \\ 1.3764 & 0.5257 & 0 & 0.8507 \\ 2.2270 & 1.3764 & 0.8507 & 0 \end{pmatrix} \quad (4.43)$$

Abbildung 4.6 zeigt das entsprechende Shepard-Diagramm, wobei mehrfache übereinanderliegende Punkte durch eingekreiste Kreuze dargestellt sind. Bei einer fehlerfreien Abbildung würden alle Punkte auf der Hauptdiagonalen liegen,  $d_{ij}^x = d_{ij}^y$ , aber dies kann in der Regel nicht erreicht werden. In unserem Beispiel sind die meisten Punkte auf der Hauptdiagonalen oder sehr nahe, bis auf den Punkt  $d_{23}^x = d_{32}^x = 1$ ,  $d_{23}^y = d_{32}^y \approx 0.5257$ . Die Hauptachsentransformation und die mehrdimensionale Skalierung führen also für dieses Beispiel zu einer Abbildung, die fast alle paarweisen Abstände nahezu beibehält, bis auf den Abstand zwischen dem zweiten und dritten Punkt. Die Punkte in Abb. 4.6 sind



**Abb. 4.6** Shepard-Diagramm für die lineare Abbildung des 4-Punkte-Datensatzes

von links nach rechts durch gestrichelte Linien verbunden. Der gestrichelte Verlauf ist in diesem Beispiel monoton, d. h. für höhere Abstände  $d^x$  erhalten wir höhere oder gleiche Abstände  $d^y$ . Torgerson [9] hat eine Projektionsmethode vorgeschlagen, die solche monotonen Abbildungen bestimmt.

Wir veranschaulichen die Hauptachsentransformation und die mehrdimensionale Skalierung anhand zweier weiterer Datensätze. Als ersten Datensatz betrachten wir die dreidimensionale Helixkurve

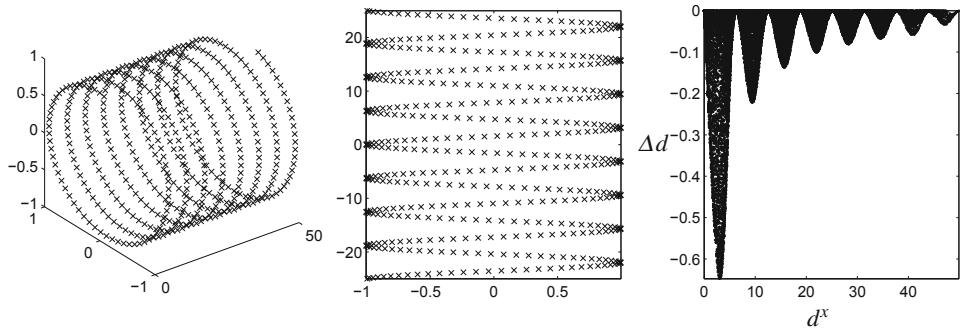
$$X = \{(t, \sin t, \cos t)^T \mid t \in \{0, 0.1, 0.2, \dots, 50\}\} \quad (4.44)$$

Abbildung 4.7 zeigt diesen Datensatz (links), seine Projektion (Mitte) und die Projektionsfehler  $\Delta d = d_{ij}^y - d_{ij}^x$  über  $d_{ij}^x$  (rechts). Wir betrachten hier die Projektionsfehler  $\Delta d$  statt der Projektionsabstände  $d^y$ , da die absoluten Projektionsfehler  $\|\Delta d\|$  hier viel kleiner sind als die Projektionsabstände  $d^y$  und ein gewöhnliches Shepard-Diagramm daher nur Punkte nahe der Hauptdiagonalen zeigen würde. Die Hauptachsenprojektion bzw. mehrdimensionale Skalierung in Abb. 4.7 (Mitte) gibt die Struktur der Daten (links) sehr gut wieder, und die Projektionsfehler (rechts) sind niedrig. Allerdings sind die Projektionsfehler null oder negativ, nie positiv, so dass die Projektion zu niedrigeren Abständen  $d^y$  tendiert und der Mittelwert des Projektionsfehlers nicht null, sondern negativ ist.

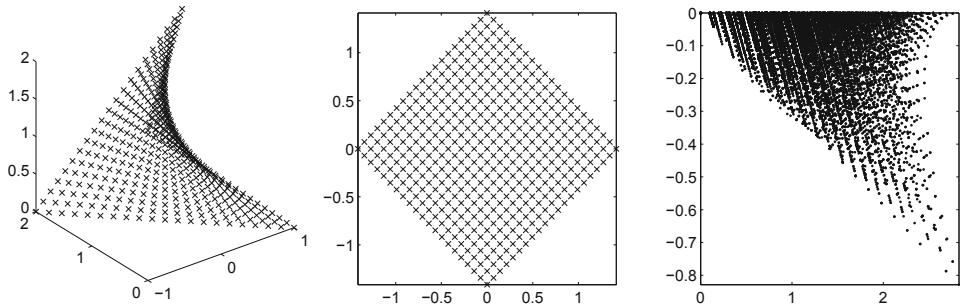
Als zweiten Datensatz betrachten wir das dreidimensionale gebogene Quadrat

$$X = \{((t_1 - 1) \cdot (t_2 - 1), t_1, t_2)^T \mid t_1, t_2 \in \{0, 0.1, 0.2, \dots, 2\}\} \quad (4.45)$$

Abbildung 4.8 zeigt diesen Datensatz, seine Projektion und die Projektionsfehler. Die Projektion gibt die Datenstruktur gut wieder, aber die Projektionsfehler zeigen erneut eine Tendenz zu niedrigeren Werten von  $d^y$ .



**Abb. 4.7** Helixdatensatz, lineare Projektion und Projektionsfehler



**Abb. 4.8** Gebogenes Quadrat, lineare Projektion und Projektionsfehler

Die Hauptachsentransformation bzw. die mehrdimensionale Skalierung ist eine lineare Abbildung mit relativ geringem Rechenaufwand und einer hohen Robustheit. Komplizierte geometrische Strukturen mit Hinterschneidungen können mit linearen Abbildungen nur unzureichend wiedergegeben werden. Hierfür sind nichtlineare Abbildungen besser geeignet. In den folgenden beiden Abschnitten werden zwei wichtige Beispiele für nichtlineare Projektionsverfahren dargestellt: die Sammon-Abbildung und der Auto-Assoziator.

## 4.4 Sammon-Abbildung

Die Idee der Sammon-Abbildung [6] ist, einen Datensatz  $X \subset \mathbb{R}^p$  so auf einen Datensatz  $Y \subset \mathbb{R}^q$  abzubilden, dass die paarweisen Abstände in  $X$  möglichst gut mit den entsprechenden paarweisen Abständen in  $Y$  übereinstimmen.

$$d_{ij}^x \approx d_{ij}^y \quad (4.46)$$

$i, j = 1, \dots, n$ . Ebenso wie die mehrdimensionale Skalierung kann die Sammon-Abbildung auch verwendet werden, um eine Merkmalsraumrepräsentation  $Y$  zu einer

Abstandsmatrix  $D^x$  zu finden. Zur Bestimmung von  $Y$  wird der Fehler zwischen  $D^x$  und  $D^y$  minimiert. Für diese Minimierung wurden folgende Fehlerfunktionen vorgeschlagen:

$$E_1 = \frac{1}{\sum_{i=1}^n \sum_{j=i+1}^n (d_{ij}^x)^2} \sum_{i=1}^n \sum_{j=i+1}^n (d_{ij}^y - d_{ij}^x)^2 \quad (4.47)$$

$$E_2 = \sum_{i=1}^n \sum_{j=i+1}^n \left( \frac{d_{ij}^y - d_{ij}^x}{d_{ij}^x} \right)^2 \quad (4.48)$$

$$E_3 = \frac{1}{\sum_{i=1}^n \sum_{j=i+1}^n d_{ij}^x} \sum_{i=1}^n \sum_{j=i+1}^n \frac{(d_{ij}^y - d_{ij}^x)^2}{d_{ij}^x} \quad (4.49)$$

Die Vorfaktoren in  $E_1$  und  $E_3$  hängen nur von  $X$  ab und können bei der Optimierung ignoriert werden.  $E_1$  betrachtet die absoluten quadratischen Fehler,  $E_2$  betrachtet die relativen quadratischen Fehler, und  $E_3$  ist ein Kompromiss zwischen absoluten und relativen quadratischen Fehlern, der in der Regel die besten Ergebnisse liefert und daher im Folgenden betrachtet wird. Für Minima der sogenannten *Sammon-Fehlerfunktion*  $E_3$  ist keine geschlossene Lösung bekannt. Zur Bestimmung von  $Y$  verwenden wir das Gradientenverfahren oder die Newton-Optimierung (siehe Anhang). Zur Berechnung der ersten und zweiten Ableitungen von  $E_3$  beachten wir, dass

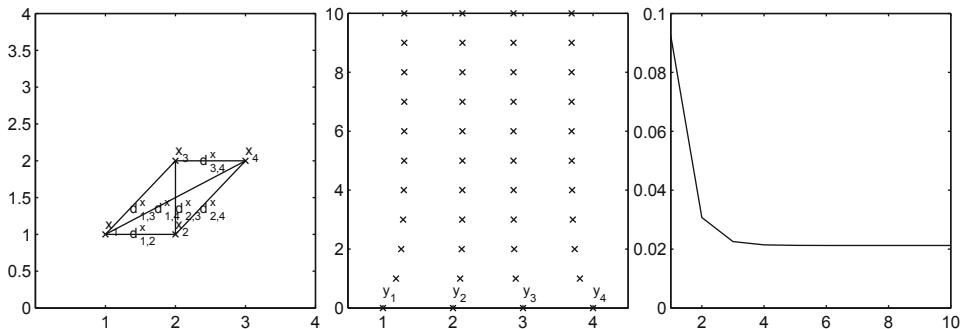
$$\frac{\partial d_{ij}^y}{\partial y_k} = \frac{\partial}{\partial y_k} \|y_i - y_j\| = \begin{cases} \frac{y_k - y_j}{d_{kj}^y} & \text{falls } i = k \\ 0 & \text{sonst} \end{cases} \quad (4.50)$$

und erhalten somit

$$\frac{\partial E_3}{\partial y_k} = \frac{2}{\sum_{i=1}^n \sum_{j=i+1}^n d_{ij}^x} \sum_{j=1}^n \left( \frac{1}{d_{kj}^x} - \frac{1}{d_{kj}^y} \right) (y_k - y_j) \quad (4.51)$$

$$\frac{\partial^2 E_3}{\partial y_k^2} = \frac{2}{\sum_{i=1}^n \sum_{j=i+1}^n d_{ij}^x} \sum_{j=1}^n \left( \frac{1}{d_{kj}^x} - \frac{1}{d_{kj}^y} - \frac{(y_k - y_j)^2}{(d_{kj}^y)^3} \right) \quad (4.52)$$

Wir betrachten erneut den Datensatz aus Abb. 4.5. Abbildung 4.9 (links) zeigt die paarweisen Abstände  $d^x$  gemäß (4.42). Wir initialisieren  $Y$  als  $Y = \{1, 2, 3, 4\}$ , was den vier Punkten in der untersten Reihe in Abb. 4.9 (Mitte) entspricht und zu folgender



**Abb. 4.9** 4-Punkte-Datensatz: paarweise Abstände, Sammon-Abbildung nach 0, …, 10 Iterationen, Werte der Sammon-Fehlerfunktion

Abstandsmatrix führt:

$$D^y = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 3 & 2 & 1 & 0 \end{pmatrix} \quad (4.53)$$

Der initiale Wert der Fehlerfunktion ergibt sich damit zu

$$E_3 = \frac{1}{3 \cdot 1 + 2 \cdot \sqrt{2} + \sqrt{5}} \cdot \left( 2 \cdot \frac{(2 - \sqrt{2})^2}{\sqrt{2}} + \frac{(3 - \sqrt{5})^2}{\sqrt{5}} \right) \approx 0.0925 \quad (4.54)$$

Dieser Wert entspricht dem linken Ende der Fehlerfunktion in Abb. 4.9 (rechts). Für diese Werte ergeben sich die Fehlergradienten zu

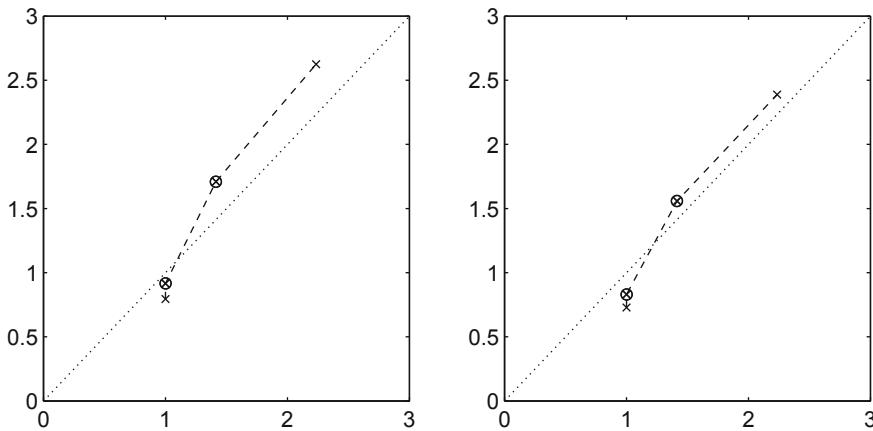
$$\frac{\partial E_3}{\partial y_1} = \frac{2}{3 \cdot 1 + 2 \cdot \sqrt{2} + \sqrt{5}} \cdot \left( -\frac{2 - \sqrt{2}}{\sqrt{2}} - \frac{3 - \sqrt{5}}{\sqrt{5}} \right) \approx -0.1875 \quad (4.55)$$

$$\frac{\partial E_3}{\partial y_2} = \frac{2}{3 \cdot 1 + 2 \cdot \sqrt{2} + \sqrt{5}} \cdot \left( -\frac{2 - \sqrt{2}}{\sqrt{2}} \right) \approx -0.1027 \quad (4.56)$$

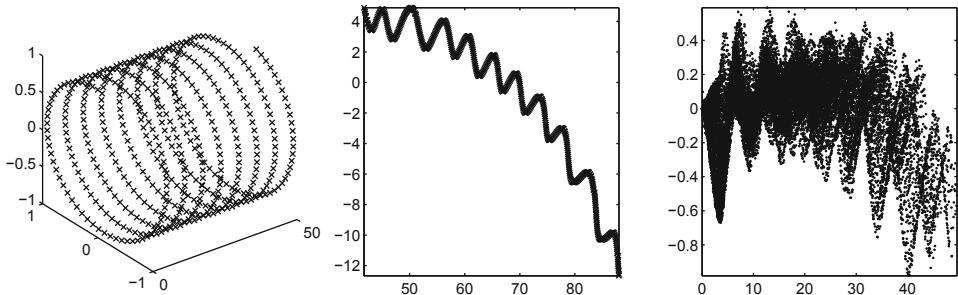
$$\frac{\partial E_3}{\partial y_3} = \frac{2}{3 \cdot 1 + 2 \cdot \sqrt{2} + \sqrt{5}} \cdot \left( \frac{2 - \sqrt{2}}{\sqrt{2}} \right) \approx 0.1027 \quad (4.57)$$

$$\frac{\partial E_3}{\partial y_4} = \frac{2}{3 \cdot 1 + 2 \cdot \sqrt{2} + \sqrt{5}} \cdot \left( \frac{3 - \sqrt{5}}{\sqrt{5}} + \frac{2 - \sqrt{2}}{\sqrt{2}} \right) \approx 0.1875 \quad (4.58)$$

Für Schrittänge  $\alpha = 1$  liefert das Gradientenverfahren den nächsten Schätzwert  $Y \approx (1.1875, 2.1027, 2.8973, 3.8125)$ , der den vier Punkten in der zweiten Reihe von unten in Abb. 4.9 (Mitte) entspricht. Das mittlere und rechte Diagramm in Abb. 4.9 zeigen die Werte

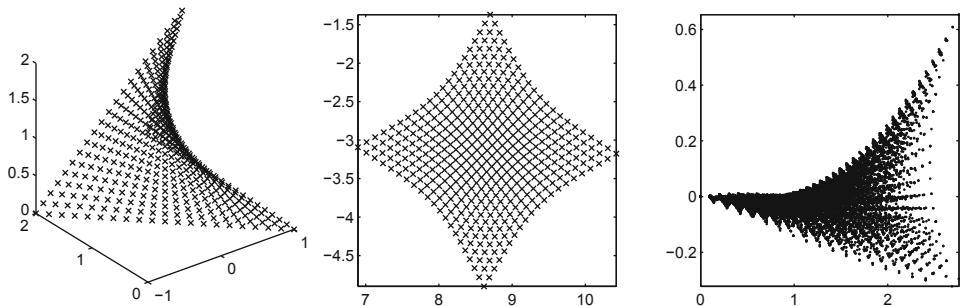


**Abb. 4.10** Shepard-Diagramme für die Sammon-Abbildung des 4-Punkte-Datensatzes



**Abb. 4.11** Helixdatensatz, Sammon-Projektion und Projektionsfehler

von  $Y$  und  $E_3$  für die ersten zehn Schritte des Gradientenverfahrens. Nach zehn Schritten erhalten wir  $Y \approx (1.3058, 2.1359, 2.8641, 3.6942)$  und  $E_3 \approx 0.0212$ . Abbildung 4.10 zeigt die Shepard-Diagramme der Sammon-Abbildungen nach einem und nach zehn Schritten des Gradientenverfahrens. Die Punkte liegen näher an der Hauptdiagonalen als bei der linearen Projektion in Abb. 4.6. Abbildung 4.11 und 4.12 zeigen die Sammon-Abbildungen für die Helix (4.44) und das gebogene Quadrat (4.45) nach 100 Schritten mit dem Newton-Verfahren. Die Sammon-Abbildung führt zu einem niedrigeren Projektionsfehler als die linearen Projektionen in Abb. 4.7 und 4.8). Zudem ist der Projektionsfehler der Sammon-Abbildung mittelwertfrei. Der Rechenaufwand der Sammon-Abbildung ist jedoch deutlich höher als bei der linearen Abbildung.



**Abb. 4.12** Gebogenes Quadrat, Sammon-Projektion und Projektionsfehler

## 4.5 Auto-Assoziator

Ein Auto-Assoziator bestimmt zwei Funktionen  $f: \mathbb{R}^p \rightarrow \mathbb{R}^q$  und  $g: \mathbb{R}^q \rightarrow \mathbb{R}^p$  für die Abbildung von  $X$  auf  $Y$  und zurück.

$$y_k = f(x_k) \quad (4.59)$$

$$x_k \approx g(y_k) \quad (4.60)$$

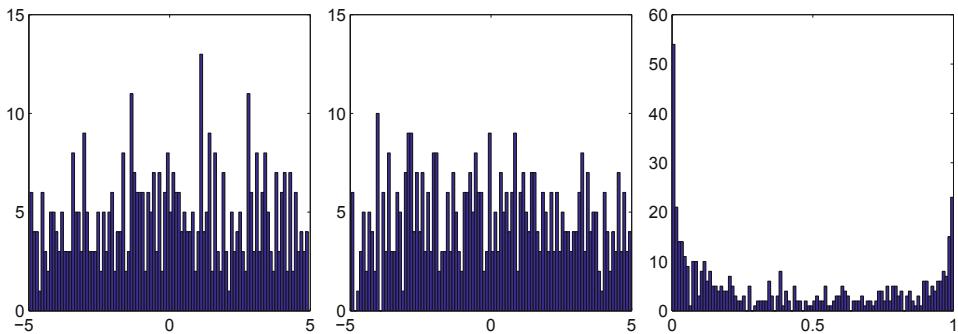
$k = 1, \dots, n$ . Für einen gegebenen Datensatz  $X$  kann die verkettete Funktion  $g \circ f$  durch Regressionsverfahren bestimmt werden (siehe Kap. 6). Dabei dient jeder Vektor  $x_k$  beim Training der Regressionsfunktion als Eingabewert und zugleich auch als Ausgabewert, daher der Name *Auto-Assoziator*.

$$x_k \approx g \circ f(x_k) = g(f(x_k)) \quad (4.61)$$

Wenn die Regressionsfunktion  $g \circ f$  gefunden ist, wird die Vorwärtstransformation  $f$  benutzt, um die projizierten Daten  $Y$  aus  $X$  mit (4.59) zu berechnen. Die Rückwärtstransformation  $g$  wird dabei ignoriert. Als Regressionsmethode für Auto-Assoziatoren werden häufig neuronale Netze, beispielsweise das *mehrschichtige Perzeptron* (MLP, siehe Kap. 6) verwendet. Sowohl die Güte der Projektion als auch der Berechnungsaufwand eines Auto-Assoziators hängt von der verwendeten Regressionsfunktion ab.

## 4.6 Histogramme

In den vorangegangenen Abschnitten wurden Methoden zur Visualisierung von Merkmalsvektoren vorgestellt. In diesem Abschnitt werden Methoden zur Visualisierung der statistischen Verteilung von Merkmalswerten betrachtet. Statistische Maße wie Modalwert, Median oder Mittelwert (siehe Kap. 2) können zur groben Beschreibung der



**Abb. 4.13** Histogramme der Merkmale  $x, y, z$

Verteilung von Merkmalswerten verwendet werden. Die tatsächliche Verteilung der Merkmalswerte kann durch *Histogramme* visualisiert werden. Wir betrachten Histogramme einzelner Merkmale, die wir der Einfachheit halber als eindimensionale Datensätze  $X$  schreiben. Für die Merkmalswerte definieren wir  $m$  Histogramm-Intervalle

$$[\xi_1, \xi_2), [\xi_2, \xi_3), \dots, [\xi_{m-1}, \xi_m), [\xi_m, \xi_{m+1}] \quad (4.62)$$

$$\xi_1 = \min X, \quad \xi_{m+1} = \max X \quad (4.63)$$

und zählen, wie viele Werte in  $X$  in das jeweilige Intervall fallen. Dies liefert die Häufigkeiten

$$h_k(X) = |\{\xi \in X \mid \xi_k \leq \xi < \xi_{k+1}\}|, \quad k = 1, \dots, m-1 \quad (4.64)$$

$$h_m(X) = |\{\xi \in X \mid \xi_m \leq \xi \leq \xi_{m+1}\}| \quad (4.65)$$

wobei die Summe der Häufigkeitswerte der Anzahl der Datenpunkte entspricht.

$$\sum_{k=1}^m h_k(X) = |X| = n \quad (4.66)$$

Häufig werden Intervalle mit der gleichen Breite  $\Delta x = (\max X - \min X)/m$  benutzt, so dass die Intervallgrenzen bei  $\xi_k = \min X + (k-1) \cdot \Delta x$  liegen,  $k = 1, \dots, m+1$ . Jeder Häufigkeitswert  $h_k$ ,  $k = 1, \dots, m$  kann durch einen Balken dargestellt werden, dessen linke und rechte Seite der unteren und oberen Grenze des  $k$ -ten Intervalls und dessen Höhe der Häufigkeit  $h_k$  entspricht. Ein Diagramm mit solchen Balken wird *Histogramm* genannt. Abbildung 4.13 zeigt die Histogramme ( $m = 100$ ) der drei Merkmale des Datensatzes aus Abb. 4.3. Die ersten beiden Merkmale sind offenbar näherungsweise gleichverteilt. Das dritte Merkmal enthält viele Werte nahe bei null und nahe bei eins, die den unteren und oberen Plateaus in Abb. 4.2 (rechts) entsprechen. Solche Häufungen können in Histogrammen leicht erkannt werden. Histogramme werden häufig auch verwendet, um abzuschätzen, welcher Typ der statistischen Verteilung vorliegt, beispielsweise Gleichverteilung, Gauß-Verteilung, Poisson-Verteilung oder skalenfreie Verteilung. Der Parameter

$m$  (die Anzahl der Intervalle) sollte sorgfältig gewählt werden. Für zu kleine Werte von  $m$  reicht die Auflösung nicht aus, um die Form der Verteilung im Histogramm zu erkennen. Für zu große Werte von  $m$  sind die meisten Intervalle leer oder enthalten nur einzelne Punkte, so dass die Form der Verteilung ebenfalls nicht erkennbar ist. Die optimale Wahl von  $m$  hängt von den Daten ab, ihrer Anzahl und ihrer Verteilung. Folgende Regeln werden häufig für die Wahl von  $m$  benutzt:

- Sturgess [8] betrachtet nur die Anzahl der Daten.

$$m = 1 + \log_2 n \quad (4.67)$$

- Scott [7] geht von einer Gauß-Verteilung mit Standard-Abweichung  $s$  aus.

$$m = \frac{3.49 \cdot s}{\sqrt[3]{n}} \quad (4.68)$$

- Freedman und Diaconis [2] betrachten das mittlere 50 %-Quantil (und sind daher auch für sogenannte Long-Tail-Verteilungen, also solche mit „langem Schwanz“, geeignet).

$$m = \frac{2 \cdot (Q_{75\%} - Q_{25\%})}{\sqrt[3]{n}} \quad (4.69)$$

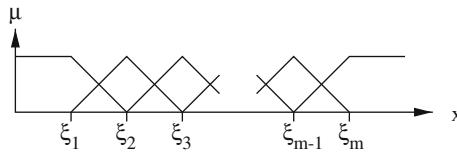
mit den Quantilen

$$|\{x \in X \mid x \leq Q_\varphi\}| = \varphi \cdot n \quad (4.70)$$

Statt gleicher Intervallbreiten werden auch häufig Intervalle unterschiedlicher Breite verwendet, z. B. um eine höhere Auflösung in Bereichen größerer Datendichte zu erreichen [3]. Histogrammintervalle können auch als  $1/m$ -Quantile definiert werden, so dass jedes Intervall die gleiche Anzahl der Daten enthält. Zur Visualisierung von Histogrammen mit unterschiedlichen Intervallbreiten ergibt sich die Balkenhöhe nicht durch die Anzahl der Daten im jeweiligen Intervall, sondern durch die Anzahl der Daten geteilt durch die Intervallbreite, so dass die Fläche jedes Balkens der Anzahl der jeweiligen Datenpunkte entspricht.

Jeder Datenpunkt zwischen der unteren und oberen Grenze eines Histogramms wird genau einem Intervall zugeordnet, ob der Punkt sich in der Mitte des Intervalls befindet oder eher an der unteren oder oberen Intervallgrenze. In einem *unscharfen Histogramm* [3] wird die Zuordnung der Daten über benachbarte Intervalle verschmiert. Ein Punkt an der Intervallgrenze würde beispielsweise beiden benachbarten Intervallen jeweils zur Hälfte zugeordnet. Unscharfe Histogramme berechnen also die Häufigkeiten in sogenannten *unscharfen Intervallen*, die durch Zugehörigkeitsfunktionen  $\mu:X \rightarrow [0, 1]$  definiert werden. Hierfür werden häufig dreiecksförmige Funktionen benutzt (Abb. 4.14):

$$\mu_1(x) = \begin{cases} 1 & \text{falls } x < \xi_1 \\ \frac{\xi_2 - x}{\xi_2 - \xi_1} & \text{falls } \xi_1 \leq x < \xi_2 \\ 0 & \text{falls } x \geq \xi_2 \end{cases} \quad (4.71)$$



**Abb. 4.14** Dreiecksförmige Zugehörigkeitsfunktionen

$$\mu_k(x) = \begin{cases} 0 & \text{falls } x < \xi_{k-1} \\ \frac{x - \xi_{k-1}}{\xi_k - \xi_{k-1}} & \text{falls } \xi_{k-1} \leq x < \xi_k \\ \frac{\xi_{k+1} - x}{\xi_{k+1} - \xi_k} & \text{falls } \xi_k \leq x < \xi_{k+1} \\ 0 & \text{falls } x \geq \xi_{k+1} \end{cases} \quad (4.72)$$

$$k = 2, \dots, m-1$$

$$\mu_m(x) = \begin{cases} 0 & \text{falls } x < \xi_{m-1} \\ \frac{x - \xi_{m-1}}{\xi_m - \xi_{m-1}} & \text{falls } \xi_{m-1} \leq x < \xi_m \\ 1 & \text{falls } x \geq \xi_m \end{cases} \quad (4.73)$$

Durch solche Zugehörigkeitsfunktionen wird jeder Wert  $x$  einem Intervall oder zu einem gewissen Grad mehreren Intervallen zugeordnet, wobei die Summe der Zugehörigkeitsgrade gleich eins ist. Die Häufigkeiten eines unscharfen Histogramms berechnen sich als

$$\tilde{h}_k(X) = \sum_{x \in X} \mu_k(x) \quad (4.74)$$

Gewöhnliche Histogramme können als Sonderfälle von unscharfen Histogrammen für rechteckige Zugehörigkeitsfunktionen aufgefasst werden.

## 4.7 Spektralanalyse

Die Datenvisualisierung soll wichtige Charakteristiken der Daten aufzeigen. Wichtige Charakteristiken von *Zeitreihen* sind die periodischen Anteile, die durch spektrale Merkmale wie Amplituden- und Phasenspektrum beschrieben werden können. Diese Spektren lassen sich aus dem Satz von Fourier herleiten: Jede kontinuierlich differenzierbare Funktion lässt sich in ihre Kosinus- und Sinusanteile zerlegen, so dass

$$f(x) = \int_0^\infty (a(y) \cos xy + b(y) \sin xy) dy \quad \text{mit} \quad (4.75)$$

$$a(y) = \frac{1}{\pi} \int_{-\infty}^{\infty} f(u) \cos yu du \quad (4.76)$$

$$b(y) = \frac{1}{\pi} \int_{-\infty}^{\infty} f(u) \sin yu \, du \quad (4.77)$$

Entsprechend wird die *Fourierkosinustransformierte*  $F_c(y)$  und die *Fouriersinustransformierte*  $F_s(y)$  definiert als

$$F_c(y) = \sqrt{\frac{2}{\pi}} \int_0^{\infty} f(x) \cos xy \, dx \quad (4.78)$$

$$f(x) = \sqrt{\frac{2}{\pi}} \int_0^{\infty} F_c(y) \cos xy \, dy \quad (4.79)$$

$$F_s(y) = \sqrt{\frac{2}{\pi}} \int_0^{\infty} f(x) \sin xy \, dx \quad (4.80)$$

$$f(x) = \sqrt{\frac{2}{\pi}} \int_0^{\infty} F_s(y) \sin xy \, dy \quad (4.81)$$

Zur Anwendung der Fourierkosinus- und Fouriersinustransformation auf diskrete Funktionen verwenden wir in (4.78–4.81) die Substitutionen  $x = k \cdot T$  und  $y = l \cdot \omega$  mit der Zeitkonstanten  $T$  und der Frequenzkonstanten  $\omega$ . Wir interpretieren die Zeitreihendaten  $x_k$ ,  $k = 1, \dots, n$ , als äquidistante Abtastwerte der kontinuierlichen Funktion  $f$  und schreiben daher  $f(k \cdot T) = x_k$ ,  $k = 1, \dots, n$ . Die diskrete Fouriertransformation liefert die Fourierkosinustransformierte  $F_c(l \cdot \omega) = y_l^c$  und die Fouriersinustransformierte  $F_s(l \cdot \omega) = y_l^s$ ,  $l = 1, \dots, m$ , gemäß

$$y_l^c = \frac{2}{n} \sum_{k=1}^n x_k \cos kl\omega T \quad (4.82)$$

$$x'_k = \frac{n\omega T}{\pi} \sum_{l=1}^m y_l^c \cos kl\omega T \quad (4.83)$$

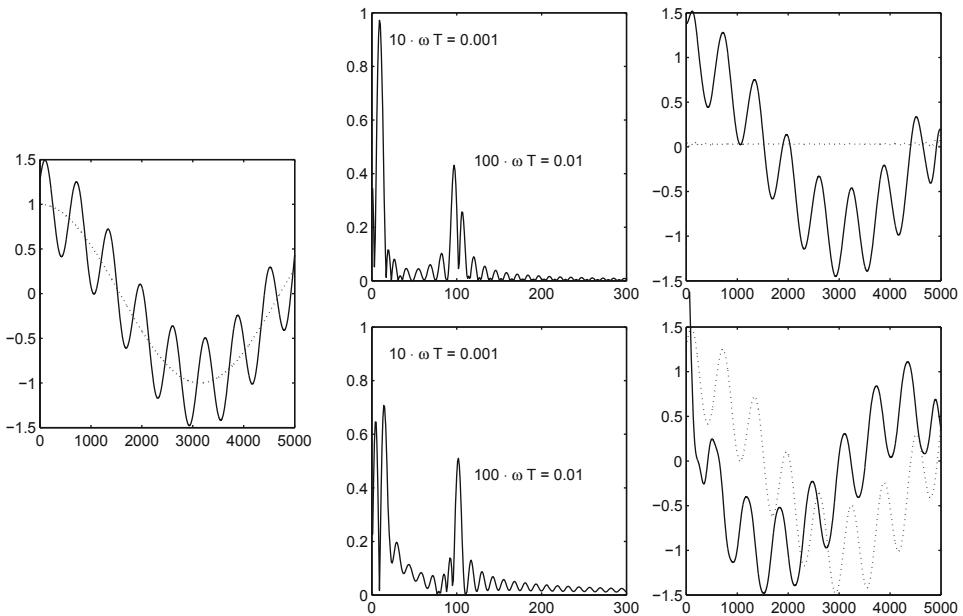
$$y_l^s = \frac{2}{n} \sum_{k=1}^n x_k \sin kl\omega T \quad (4.84)$$

$$x'_k = \frac{n\omega T}{\pi} \sum_{l=1}^m y_l^s \sin kl\omega T \quad (4.85)$$

Abbildung 4.15 (links) zeigt die  $n = 5000$  Punkte des Datensatzes

$$X = \{(i, \cos(0.001 \cdot i) + 0.5 \cdot \cos(0.01 \cdot i - 1)) \mid i \in \{1, \dots, 5000\}\} \quad (4.86)$$

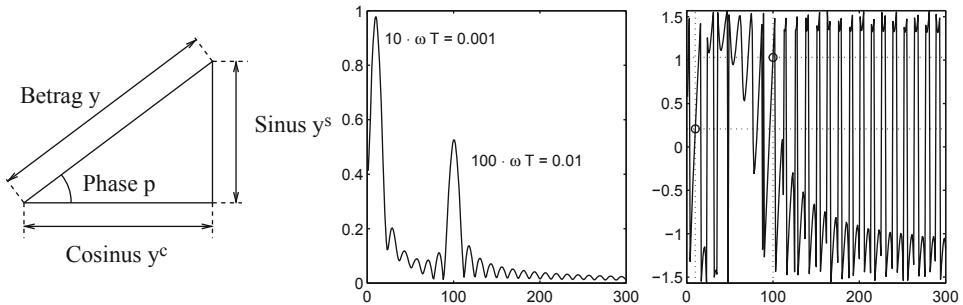
der zwei periodische Anteile enthält, eine mit niedrigerer Frequenz und höherer Amplitude und eine mit höherer Frequenz und niedrigerer Amplitude. Ein solcher Datensatz



**Abb. 4.15** Zeitreihe, Absolutwerte der Fourierkosinus- und Fouriersinustransformierten, sowie Rücktransformierte (oben Kosinus, unten Sinus)

könnte aus Absatzzahlen mit wöchentlichen und jahreszeitlichen Fluktuationen entstanden sein. Die mittlere Spalte in Abb. 4.15 zeigt die Absolutwerte der Fourierkosinustransformierten (4.82) (oben) und der Fouriersinustransformierten (4.84) (unten) für  $m = 100$  und  $\omega T = 10^{-4}$ . Beide Transformierte besitzen lokale Hauptmaxima bei etwa  $y_{10}$  und  $y_{100}$ , entsprechend den Frequenzen  $10 \cdot \omega T = 0.001$  und  $100 \cdot \omega T = 0.01$ , die den Frequenzen der Kosinusterme in (4.86) entsprechen. Die entsprechenden Maxima  $y_9^c \approx 0.9722$ ,  $y_{98}^c \approx 0.4313$ ,  $y_{14}^s \approx 0.7075$  und  $y_{103}^s \approx 0.5104$  entsprechen in etwa den Amplituden der Kosinusterme in (4.86), aber  $y_{98}^c$  und  $y_{14}^s$  stimmen nicht gut mit den Koeffizienten 0.5 und 1 überein. Die rechte Spalte in Abb. 4.15 zeigt die Fourierkosinusrücktransformierte (4.83) (oben) und die Fouriersinusrücktransformierte (4.85) (unten). Die Fourierkosinusrücktransformierte stimmt gut mit der ursprünglichen Zeitreihe überein. Der Transformationsfehler (gestrichelte Kurve) ist sehr klein. Die Fouriersinusrücktransformierte weicht dagegen stark von der ursprünglichen Zeitreihe (gestrichelte Kurve) ab. Die Fourierkosinustransformierte  $y^c$  und die Fouriersinustransformierte  $y^s$  können als Längen der Katheten eines rechtwinkligen Dreiecks interpretiert werden (Abb. 4.16 links). Das *Amplitudenspektrum*  $y$  ist definiert durch die Hypotenuse und das *Phasenspektrum* durch den Ankathetenwinkel dieses Dreiecks.

$$y_l = \sqrt{(y_l^c)^2 + (y_l^s)^2} \quad (4.87)$$



**Abb. 4.16** Dreiecksbeziehungen, Amplituden- und Phasenspektrum

$$p_l = \arctan \frac{y_l^s}{y_l^c} \quad (4.88)$$

Abbildung 4.16 (Mitte) zeigt das Amplitudenspektrum (4.87) unserer Zeitreihe. Die beiden lokalen Hauptmaxima bei  $y_{10} \approx 0.9784$  und  $y_{101} \approx 0.5261$  stimmen sehr gut mit den Frequenzen und Amplituden der Kosinusterme in (4.86) überein. Abbildung 4.16 (rechts) zeigt das Phasenspektrum (4.88). Die Winkel an den lokalen Hauptmaxima  $p_{10} \approx 0.2073$  und  $p_{100} \approx 1.0320$  stimmen näherungsweise mit den Phasenverschiebungen 0 und 1 der Kosinusterme in (4.86) überein. Das Amplitudenspektrum  $Y = \{y_1, \dots, y_m\} \subset \mathbb{R}$  und das Phasenspektrum  $P = \{p_1, \dots, p_m\} \subset \mathbb{R}$  eignen sich also zur Visualisierung der Frequenzen, Amplituden und Phasenwinkel der periodischen Anteile einer Zeitreihe.

### Übungsaufgaben

- 4.1.** Bestimmen Sie die Hauptachsen des Datensatzes  $X = \{(-3, -1, -1), (0, -1, 0), (-2, -1, 2), (1, -1, 3)\}$ .
- 4.2.** Bestimmen Sie die zweidimensionale Hauptachsentransformation, die Rücktransformation, sowie den Projektionsfehler und zeichnen Sie das Shepard-Diagramm dieser Projektion.
- 4.3.** Bestimmen Sie die eindimensionale Hauptachsentransformation, die Rücktransformation, sowie den Projektionsfehler und zeichnen Sie das Shepard-Diagramm dieser Projektion.
- 4.4.** Was schließen Sie aus den Ergebnissen dieser beiden Projektionen?

## Literatur

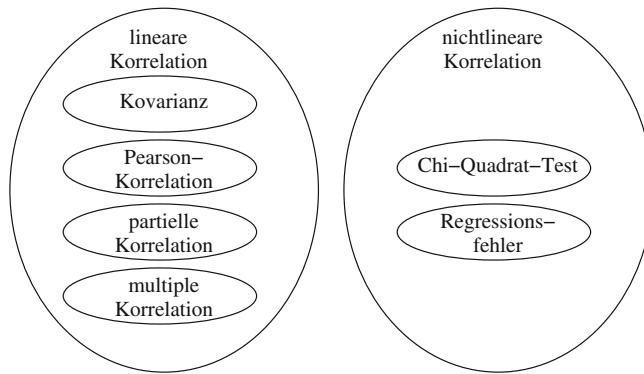
1. R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1974.
2. D. Freedman and P. Diaconis. On the histogram as a density estimator: L2 theory. *Probability Theory and Related Fields*, 57(4):453–476, December 1981.
3. A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, Englewood Cliffs, 1986.
4. K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901.
5. T. A. Runkler. Fuzzy histograms and fuzzy chi-squared tests for independence. In *IEEE International Conference on Fuzzy Systems*, volume 3, pages 1361–1366, Budapest, July 2004.
6. J. W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, C-18(5):401–409, 1969.
7. D. W. Scott. On optimal and data-based histograms. *Biometrika*, 66(3):605–610, 1979.
8. H. A. Sturges. The choice of a class interval. *Journal of the American Statistical Association*, pages 65–66, 1926.
9. W. S. Torgerson. *Theory and Methods of Scaling*. Wiley, New York, 1958.
10. J. W. Tukey. *Exploratory Data Analysis*. Addison Wesley, Reading, 1987.
11. G. Young and A. S. Householder. Discussion of a set of points in terms of their mutual distances. *Psychometrika*, 3:19–22, 1938.

## Zusammenfassung

Die Korrelationsanalyse quantifiziert den Zusammenhang zwischen Merkmalen. Die lineare Korrelationsanalyse ist robust und effizient, erfasst aber nur lineare Zusammenhänge. Die nichtlineare Korrelationsanalyse erfasst auch nichtlineare Zusammenhänge, muss aber sorgfältig parametrisiert werden. Als Beispiel für nichtlineare Korrelationsverfahren stellen wir den Chi-Quadrat-Test auf stochastische Unabhängigkeit vor, der im kontinuierlichen Fall mit Histogramm-Methoden kombiniert werden kann. Die nichtlineare Korrelation kann auch durch den Validierungsfehler von Regressionsmodellen quantifiziert werden. Stark korrelierte Merkmale stehen nicht unbedingt in kausalem Zusammenhang, sondern können auch durch Scheinkorrelationen bedingt sein. Die partielle Korrelationsanalyse erlaubt es, Effekte von Scheinkorrelationen herauszurechnen.

## 5.1 Lineare Korrelation

Die Korrelation quantifiziert den Grad des Zusammenhangs zwischen Merkmalen. Ziel der Korrelationsanalyse ist es, zusammenhängende Merkmale zu identifizieren, um Ursachen für beobachtete Effekte zu erklären oder gezielt bestimmte Effekte herbeizuführen zu können. In einer Produktionsanlage würde die Korrelationsanalyse beispielsweise diejenigen Merkmale identifizieren, die mit der Produktqualität zusammenhängen, so dass eine angestrebte Produktqualität durch gezielte Variation dieser Merkmale erzielt werden kann. Abbildung 5.1 zeigt einen Überblick über die in diesem Kapitel beschriebenen Korrelationsverfahren. Wir betrachten zunächst die lineare Korrelation von Merkmalspaaren. In (2.20) und (4.8) haben wir die Kovarianzmatrix  $C$  eines Datensatzes  $X \subset \mathbb{R}^p$  definiert, wobei jedes Element  $c_{ij}$  die Kovarianz zwischen den Merkmalen  $x^{(i)}$  und  $x^{(j)}$  beschreibt,



**Abb. 5.1** Korrelationsverfahren

$i, j = 1, \dots, p$ .

$$c_{ij} = \frac{1}{n-1} \sum_{k=1}^n (x_k^{(i)} - \bar{x}^{(i)})(x_k^{(j)} - \bar{x}^{(j)}) \quad (5.1)$$

Große positive Werte von  $c_{ij}$  deuten auf eine starke positive Abhängigkeit zwischen  $x^{(i)}$  und  $x^{(j)}$  hin, d. h. wir beobachten größere Werte von  $x^{(i)}$  zusammen mit größeren Werten von  $x^{(j)}$  und kleinere Werte von  $x^{(i)}$  zusammen mit kleineren Werten von  $x^{(j)}$ . Große negative Werte von  $c_{ij}$  deuten auf eine starke negative Abhängigkeit zwischen  $x^{(i)}$  und  $x^{(j)}$  hin, d. h. wir beobachten größere Werte von  $x^{(i)}$  zusammen mit kleineren Werten von  $x^{(j)}$  und kleinere Werte von  $x^{(i)}$  zusammen mit größeren Werten von  $x^{(j)}$ . Kleine (positive oder negative) Werte von  $c_{ij}$  deuten auf eine schwache Abhängigkeit zwischen  $x^{(i)}$  und  $x^{(j)}$  hin. Wird ein Merkmal mit einem konstanten Faktor  $\alpha$  multipliziert, beispielsweise weil das Merkmal in einer anderen Einheit gemessen wird (z. B. Meter statt Kilometer), so steigen die Kovarianzwerte zwischen diesem Merkmal und jedem anderen Merkmal ebenfalls um den Faktor  $\alpha$ , obwohl der Zusammenhang zwischen den Merkmalen eigentlich der gleiche bleibt. Die Pearson-Korrelation kompensiert diesen Skalierungseffekt, indem die Kovarianz durch das Produkt der Standardabweichungen der beiden Merkmale geteilt wird.

$$s_{ij} = \frac{c_{ij}}{s^{(i)} s^{(j)}} \quad (5.2)$$

$$= \frac{\sum_{k=1}^n (x_k^{(i)} - \bar{x}^{(i)})(x_k^{(j)} - \bar{x}^{(j)})}{\sqrt{\left(\sum_{k=1}^n (x_k^{(i)} - \bar{x}^{(i)})^2\right) \left(\sum_{k=1}^n (x_k^{(j)} - \bar{x}^{(j)})^2\right)}} \quad (5.3)$$

$$= \frac{\sum_{k=1}^n x_k^{(i)} x_k^{(j)} - n \bar{x}^{(i)} \bar{x}^{(j)}}{\sqrt{\left(\sum_{k=1}^n (x_k^{(i)})^2 - n (\bar{x}^{(i)})^2\right) \left(\sum_{k=1}^n (x_k^{(j)})^2 - n (\bar{x}^{(j)})^2\right)}} \quad (5.4)$$

Die Standardabweichungen sind die Quadratwurzeln der Varianzen und somit der Diagonalelemente der Kovarianzmatrix  $s^{(i)} = \sqrt{c_{ii}}$ . Die Korrelationsmatrix kann daher unmittelbar aus der Kovarianzmatrix berechnet werden.

$$s_{ij} = \frac{c_{ij}}{\sqrt{c_{ii} c_{jj}}} \quad (5.5)$$

Für die Korrelationen gilt  $s_{ij} \in [-1, 1]$ . Für  $s_{ij} \approx 1$  ist die Korrelation zwischen  $x^{(i)}$  und  $x^{(j)}$  stark positiv. Für  $s_{ij} \approx -1$  ist die Korrelation zwischen  $x^{(i)}$  und  $x^{(j)}$  stark negativ. Für  $s_{ij} \approx 0$  ist die Korrelation zwischen  $x^{(i)}$  und  $x^{(j)}$  gering, d. h. die Merkmale  $x^{(i)}$  und  $x^{(j)}$  sind voneinander (fast) unabhängig. Korrelation kann also als das Gegenteil von Unabhängigkeit aufgefasst werden. Für  $\mu$ - $\sigma$ -standardisierte Daten sind Kovarianz und Korrelation gleich, denn

$$c_{ii} = c_{jj} = 1 \Rightarrow s_{ij} = \frac{c_{ij}}{\sqrt{c_{ii} c_{jj}}} = c_{ij} \quad (5.6)$$

## 5.2 Korrelation und Kausalität

Wir unterscheiden zwischen einer Korrelation und einer kausalen Beziehung zwischen zwei Merkmalen. Eine Korrelation zwischen  $x$  und  $y$  kann auf folgende kausale Beziehungen hindeuten:

1. Zufall
2.  $x$  wirkt auf  $y$
3.  $y$  wirkt auf  $x$
4.  $z$  wirkt auf  $x$  und  $y$

Selbst wenn die Daten auf eine Korrelation hindeuten, kann dies ein Zufall sein und es besteht kein kausaler Zusammenhang zwischen  $x$  und  $y$  (Szenario 1). Liegt tatsächlich ein kausaler Zusammenhang zwischen  $x$  und  $y$  vor, so lässt sich mit Hilfe der Korrelationsanalyse nicht unterscheiden, ob  $x$  auf  $y$  wirkt oder  $y$  auf  $x$  (Szenarien 2 und 3). Wird beispielsweise eine Korrelation zwischen Übergewicht und dem Konsum zuckerfreier Getränke festgestellt, so erklären die Daten nicht, ob Übergewicht zum Konsum zuckerfreier Getränke oder der Konsum zuckerfreier Getränke zu Übergewicht führt. Es ist auch möglich, dass  $x$  und  $y$  zwar in keiner kausalen Beziehung zueinander stehen, aber beide durch eine dritte Variable  $z$  beeinflusst werden (Szenario 4). Eine Korrelation zwischen Waldbränden und Korntrag deutet beispielsweise nicht darauf hin, dass Waldbrände den Korntrag erhöhen oder der Korntrag die Wahrscheinlichkeit für Waldbrände erhöht, sondern dass sowohl Waldbrände als auch hoher Korntrag durch sonniges Wetter

gefördert werden. Dieses Szenario heißt *Scheinkorrelation*, *Drittvariablen-Effekt* oder *Störfaktor-Effekt* [4]. Die Korrelationsanalyse liefert keine Antwort, welches der vier betrachteten Szenarien vorliegt. Hierzu wird zusätzliche Information, z. B. Expertenwissen, benötigt.

Sind die Merkmale  $x^{(i)}$  und  $x^{(j)}$  miteinander korreliert und zusätzlich mit  $x^{(k)}$  korreliert (wie beim obigen Szenario der Scheinkorrelation), dann interessiert oft die Korrelation zwischen  $x^{(i)}$  und  $x^{(j)}$  ohne den Einfluss von  $x^{(k)}$ . Diese sogenannte *partielle* oder *bedingte Korrelation* ist definiert als

$$s_{ij|k} = \frac{s_{ij} - s_{ik}s_{jk}}{\sqrt{(1 - s_{ik}^2)(1 - s_{jk}^2)}} \quad (5.7)$$

Die Korrelation zwischen  $x^{(i)}$  und  $x^{(j)}$  ohne den Einfluss zweier Merkmale  $x^{(k)}$  und  $x^{(l)}$  heißt *bipartiale Korrelation* und ist definiert als

$$s_{i|k, j|l} = \frac{s_{ij} - s_{ik}s_{jk} - s_{il}s_{jl} + s_{ik}s_{kl}s_{jl}}{\sqrt{(1 - s_{ik}^2)(1 - s_{jl}^2)}} \quad (5.8)$$

Die Korrelation von  $x^{(i)}$  mit einer Menge von Merkmalen  $x^{(j_1)}, \dots, x^{(j_q)}$  heißt *multiple Korrelation* und ist definiert als

$$s_{i,(j_1, \dots, j_q)} = \sqrt{(s_{ij_1} \dots s_{ij_q}) \cdot \begin{pmatrix} 1 & s_{j_2 j_1} & \dots & s_{j_1 j_q} \\ s_{j_1 j_2} & 1 & \dots & s_{j_2 j_q} \\ \vdots & \vdots & \ddots & \vdots \\ s_{j_1 j_q} & s_{j_2 j_q} & \dots & 1 \end{pmatrix}^{-1} \cdot \begin{pmatrix} s_{ij_1} \\ s_{ij_2} \\ \vdots \\ s_{ij_q} \end{pmatrix}} \quad (5.9)$$

Für  $q = 1$  liefert die multiple Korrelation den Betrag der gewöhnlichen Korrelation.

$$s_{i,(j_1)} = |s_{ij_1}| \quad (5.10)$$

Für  $q = 2$  erhalten wir

$$s_{i,(j_1, j_2)} = \sqrt{\frac{s_{ij_1}^2 + s_{ij_2}^2 - 2s_{ij_1}s_{ij_2}s_{j_1 j_2}}{1 - s_{j_1 j_2}^2}} \quad (5.11)$$

Die Normalisierungsbedingung  $s_{ij} = [-1, 1]$  gilt für die gewöhnliche Korrelation, aber nicht allgemein für die partielle, bipartiale oder multiple Korrelation. Für weitere Informationen zur linearen Korrelationsanalyse verweisen wir auf die statistische Literatur, z. B. [1].

### 5.3 Chi-Quadrat-Test auf Unabhängigkeit

Die bisher vorgestellten Methoden der Korrelationsanalyse betrachten lineare Zusammenhänge zwischen Merkmalen. Starke nichtlineare Zusammenhänge werden mit der linearen Korrelationsanalyse möglicherweise als nur schwache Korrelation identifiziert. Eine Methode zur Quantifizierung der nichtlinearen Korrelation zwischen Merkmalen ist der *Chi-Quadrat-Test auf stochastische Unabhängigkeit* [2]. Zur Bestimmung der nichtlinearen Korrelation zwischen zwei (kontinuierlichen) Merkmalen  $x^{(1)}$  und  $x^{(2)}$  berechnen wir zunächst die Histogramme von  $x^{(1)}$  mit  $r$  Intervallen und von  $x^{(2)}$  mit  $s$  Intervallen.

$$h^{(1)} = (h_1^{(1)}, \dots, h_r^{(1)}), \quad h^{(2)} = (h_1^{(2)}, \dots, h_s^{(2)}) \quad (5.12)$$

Dann bestimmen wir  $h_{ij}$ ,  $i = 1, \dots, r$ ,  $j = 1, \dots, s$ , die Anzahl der Daten, die in das  $i$ -te Intervall von  $x^{(1)}$  und das  $j$ -te Intervall von  $x^{(2)}$  fallen, und schreiben diese Häufigkeiten als Matrix

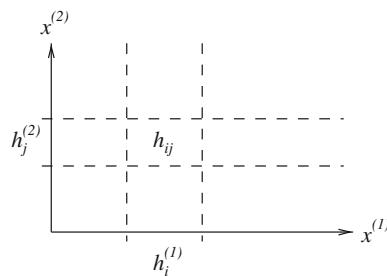
$$H = \begin{pmatrix} h_{11} & h_{12} & \cdots & h_{1s} \\ h_{21} & h_{22} & \cdots & h_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ h_{r1} & h_{r2} & \cdots & h_{rs} \end{pmatrix} \quad (5.13)$$

Abbildung 5.2 zeigt die Intervalle  $h_i^{(1)}$ ,  $h_j^{(2)}$  und das daraus resultierende Rechteck  $h_{ij}$ ,  $i = 1, \dots, r$ ,  $j = 1, \dots, s$ . Die Histogramme von  $x^{(1)}$  und  $x^{(2)}$  entsprechen den Zeilen- und Spaltensummen von  $H$ .

$$\sum_{j=1}^s h_{ij} = h_i^{(1)}, \quad i = 1, \dots, r \quad (5.14)$$

$$\sum_{i=1}^r h_{ij} = h_j^{(2)}, \quad j = 1, \dots, s \quad (5.15)$$

Falls die Merkmale  $x^{(1)}$  und  $x^{(2)}$  stochastisch unabhängig sind, dann entspricht die Wahrscheinlichkeit, dass ein Datenpunkt in das Rechteck  $h_{ij}$  fällt, dem Produkt aus der



**Abb. 5.2** Häufigkeiten für den Chi-Quadrat-Test auf stochastische Unabhängigkeit

Wahrscheinlichkeit, dass er in Intervall  $h_i^{(1)}$  fällt, und der Wahrscheinlichkeit, dass er in Intervall  $h_j^{(2)}$  fällt, also

$$\frac{h_{ij}}{n} = \frac{h_i^{(1)}}{n} \cdot \frac{h_j^{(2)}}{n} \Rightarrow h_{ij} = \frac{h_i^{(1)} \cdot h_j^{(2)}}{n} \quad (5.16)$$

wobei

$$n = \sum_{i=1}^r \sum_{j=1}^s h_{ij} = \sum_{i=1}^r h_i^{(1)} = \sum_{j=1}^s h_j^{(2)} \quad (5.17)$$

Ähnlich wie bei der Sammon-Abbildung können wir die Abweichung von  $h_{ij}$  von der stochastischen Unabhängigkeit (5.16) als absoluten quadratischen Fehler definieren,

$$E_1 = \left( h_{ij} - \frac{h_i^{(1)} \cdot h_j^{(2)}}{n} \right)^2 \quad (5.18)$$

als relativen quadratischen Fehler

$$E_2 = \left( h_{ij} - \frac{h_i^{(1)} \cdot h_j^{(2)}}{n} \right)^2 / \left( \frac{h_i^{(1)} \cdot h_j^{(2)}}{n} \right)^2 \quad (5.19)$$

oder als Kompromiss zwischen absolutem und relativem quadratischen Fehler

$$E_3 = \left( h_{ij} - \frac{h_i^{(1)} \cdot h_j^{(2)}}{n} \right)^2 / \left( \frac{h_i^{(1)} \cdot h_j^{(2)}}{n} \right) \quad (5.20)$$

Ebenso wie bei der Sammon-Abbildung wählen wir hier den Kompromiss  $E_3$  und erhalten die Statistik des Chi-Quadrat-Tests als

$$\chi^2 = \frac{1}{n} \sum_{i=1}^r \sum_{j=1}^s \frac{(n \cdot h_{ij} - h_i^{(1)} \cdot h_j^{(2)})^2}{h_i^{(1)} \cdot h_j^{(2)}} \quad (5.21)$$

Die Hypothese, dass die beiden Merkmale stochastisch unabhängig sind, wird mit Signifikanz  $\alpha$  abgelehnt, falls

$$\chi^2 > \chi^2(1 - \alpha, r - 1, s - 1) \quad (5.22)$$

Wertetabellen dieser Verteilungsfunktion lassen sich in der Literatur finden, z.B. in [1]. Kleine Werte von  $\chi^2$  unterstützen die Hypothese, dass die beiden Merkmale stochastisch unabhängig sind. Die Chi-Quadrat-Verteilung ist streng monoton, daher gilt: Je kleiner der Wert von  $\chi^2$ , desto größer ist die stochastische Unabhängigkeit zwischen den beiden Merkmalen. Um also für  $p$  Merkmale eine sortierte Liste der Merkmalspaare mit den höchsten nichtlinearen Korrelationen zu finden, genügt es, die entsprechenden Werte von

$\chi^2$  zu berechnen und zu sortieren, ohne explizit die Werte der Chi-Quadrat-Verteilung  $\chi^2(1 - \alpha, r - 1, s - 1)$  zu berechnen und ohne eine Signifikanz  $\alpha$  vorzugeben.

Die Anzahl der Intervalle sollte genauso wie bei den Histogrammen sorgfältig gewählt werden. Für eine zu geringe Anzahl von Intervallen ist das resultierende Gitter zu grob, um den nichtlinearen Zusammenhang erkennen zu können. Für eine zu hohe Anzahl von Intervallen sind viele Intervalle leer oder enthalten zu wenige Punkte, um den Zusammenhang statistisch zuverlässig erkennen zu können. Für die Wahl einer geeigneten Anzahl von Intervallen können die Regeln aus dem Abschnitt über Histogramme benutzt werden. Auch unscharfe Intervalle können im Chi-Quadrat-Test auf stochastische Unabhängigkeit benutzt werden [3].

Als Alternative zum Chi-Quadrat-Test auf stochastische Unabhängigkeit kann die nichtlineare Korrelation zwischen Merkmalen auch durch den Fehler von Regressionsmodellen quantifiziert werden. Dieser Ansatz wird im nächsten Kapitel näher erläutert.

---

### Übungsaufgaben

**5.1.** Bestimmen Sie die Kovarianz- und Korrelationsmatrizen für den Datensatz  $X = \{(1, 0), (2, 0), (3, 1), (4, 1), (5, 1), (6, 1), (7, 0), (8, 0)\}$ .

**5.2.** Berechnen Sie für den Datensatz  $X$  den Wert  $\chi^2$  des Chi-Quadrat-Tests auf stochastische Unabhängigkeit, wenn für jedes Merkmal zwei Intervalle verwendet werden.

**5.3.** Berechnen Sie für den Datensatz  $X$  den Wert  $\chi^2$  des Chi-Quadrat-Tests auf stochastische Unabhängigkeit, wenn für das erste Merkmal vier und für das zweite Merkmal zwei Intervalle verwendet werden.

**5.4.** Wie interpretieren Sie die drei verschiedenen Ergebnisse?

**5.5.** Erklären Sie den Unterschied zwischen Korrelation und Kausalität!

---

## Literatur

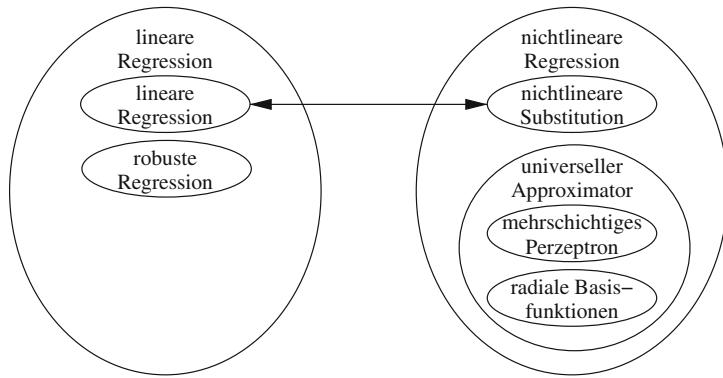
1. D. Freedman, R. Pisani, and R. Purves. *Statistics*. W. W. Norton & Company, New York, 2007.
2. K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine*, 50(302):157–175, 1900.
3. T. A. Runkler. Fuzzy histograms and fuzzy chi-squared tests for independence. In *IEEE International Conference on Fuzzy Systems*, volume 3, pages 1361–1366, Budapest, July 2004.
4. H. A. Simon. Spurious correlation: A causal interpretation. *Journal of the American Statistical Association*, 49:467–479, 1954.

## Zusammenfassung

Die Regressionsanalyse schätzt die funktionalen Abhängigkeiten zwischen Merkmalen, um Zusammenhänge zu verstehen und gezielt zu steuern. Lineare Regressionsmodelle können effizient aus den Kovarianzen berechnet werden, sind aber auf lineare Zusammenhänge beschränkt. Durch Substitution lassen sich auch bestimmte nichtlineare Regressionsmodelle durch lineare Regression finden. Robuste Regression ist weniger empfindlich gegenüber Ausreißern. Eine wichtige Familie nichtlinearer Regressionsmodelle sind universelle Approximatoren. Wir stellen zwei bekannte Beispiele für universelle Approximatoren mit neuronalen Netzen vor: mehrschichtiges Perzeptron und Netze mit radialen Basisfunktionen. Mit universellen Approximatoren lassen sich beliebig kleine Trainingsfehler erreichen, aber für Modelle mit guter Generalisierungsfähigkeit werden geringe Validierungsfehler benötigt, die sich mit Kreuzvalidierungsverfahren bestimmen lassen. Durch Merkmalselektion werden nur die relevanten Merkmale berücksichtigt, was zu einfacheren und oft genaueren Modellen führt.

## 6.1 Lineare Regression

Die im vorigen Kapitel vorgestellten Korrelationsmethoden quantifizieren den Grad des Zusammenhangs zwischen Merkmalen. Im Gegensatz dazu schätzt die Regressionsanalyse die Art des funktionalen Zusammenhangs zwischen Merkmalen. Wurden in der Korrelationsanalyse beispielsweise diejenigen Merkmale gefunden, die am stärksten mit der Produktqualität korrelieren, so besagen die Regressionsmodelle, auf welche Werte die betreffenden Merkmale gesetzt werden müssen, um eine bestimmte Zielqualität zu erreichen. Abbildung 6.1 zeigt eine Übersicht der in diesem Kapitel vorgestellten Regressionsverfahren. Wie im vorangegangenen Kapitel betrachten wir zunächst *lineare* Methoden. Die



**Abb. 6.1** Regressionsverfahren

*lineare Regression* liefert lineare funktionale Zusammenhänge zwischen Merkmalen. Die Approximation eines Merkmals  $x^{(i)}$  durch eine lineare Funktion  $f$  eines anderen Merkmals  $x^{(j)}$ , also  $x^{(i)} \approx f(x^{(j)})$ ,  $i, j \in \{1, \dots, p\}$ ,  $i, j \in \{1, \dots, p\}$ , kann geschrieben werden als

$$x_k^{(i)} \approx a \cdot x_k^{(j)} + b \quad (6.1)$$

Die lineare Regression schätzt die Parameter  $a, b \in \mathbb{R}$  dieser linearen Funktion aus  $X$  durch Minimierung einer geeigneten Fehlerfunktion. Für die lineare Regression wird gewöhnlich der quadratische Regressionsfehler minimiert:

$$E = \frac{1}{n} \sum_{k=1}^n e_k^2 = \frac{1}{n} \sum_{k=1}^n (x_k^{(i)} - a \cdot x_k^{(j)} - b)^2 \quad (6.2)$$

Eine notwendige Bedingung für (lokale) Extrema von  $E$  lautet

$$\frac{\partial E}{\partial b} = -\frac{2}{n} \sum_{k=1}^n (x_k^{(i)} - a \cdot x_k^{(j)} - b) = 0 \quad \Rightarrow \quad b = \bar{x}^{(i)} - a \cdot \bar{x}^{(j)} \quad (6.3)$$

Damit können wir den Regressionsfehler schreiben als

$$E = \frac{1}{n} \sum_{k=1}^n (x_k^{(i)} - \bar{x}^{(i)} - a(x_k^{(j)} - \bar{x}^{(j)}))^2 \quad (6.4)$$

Die andere notwendige Bedingung für (lokale) Extrema von  $E$  lautet

$$\frac{\partial E}{\partial a} = -\frac{2}{n} \sum_{k=1}^n (x_k^{(j)} - \bar{x}^{(j)}) (x_k^{(i)} - \bar{x}^{(i)} - a(x_k^{(j)} - \bar{x}^{(j)})) = 0 \quad (6.5)$$

Diese Gleichung liefert

$$a = \frac{\sum_{k=1}^n (x_k^{(i)} - \bar{x}^{(i)})(x_k^{(j)} - \bar{x}^{(j)})}{\sum_{k=1}^n (x_k^{(j)} - \bar{x}^{(j)})^2} = \frac{c_{ij}}{c_{jj}} \quad (6.6)$$

Die Parameter aller paarweisen linearen Regressionsmodelle für einen Datensatz  $X$  können also unmittelbar aus den Mittelwerten und der Kovarianzmatrix von  $X$  bestimmt werden.

Die Approximation eines Merkmals  $x^{(i)}$  durch eine lineare Funktion  $f$  von  $m \in \{2, 3, \dots\}$  Merkmalen, also  $x^{(i)} \approx f(x^{(j_1)}, \dots, x^{(j_m)})$ ,  $i, j_1, \dots, j_m \in \{1, \dots, p\}$ , kann geschrieben werden als

$$x_k^{(i)} \approx \sum_{l=1}^m a_l \cdot x_k^{(j_l)} + b \quad (6.7)$$

Die Parameter  $a_1, \dots, a_m, b \in \mathbb{R}$  werden bestimmt durch Minimierung von

$$E = \frac{1}{n} \sum_{k=1}^n e_k^2 = \frac{1}{n} \sum_{k=1}^n \left( x_k^{(i)} - \sum_{l=1}^m a_l \cdot x_k^{(j_l)} - b \right)^2 \quad (6.8)$$

Eine notwendige Bedingung für (lokale) Extrema von  $E$  liefert

$$\frac{\partial E}{\partial b} = -\frac{2}{n} \sum_{k=1}^n \left( x_k^{(i)} - \sum_{l=1}^m a_l \cdot x_k^{(j_l)} - b \right) = 0 \quad \Rightarrow \quad b = \bar{x}^{(i)} - \sum_{l=1}^m a_l \cdot \bar{x}^{(j_l)} \quad (6.9)$$

so dass

$$E = \frac{1}{n} \sum_{k=1}^n \left( x_k^{(i)} - \bar{x}^{(i)} - \sum_{l=1}^m a_l \cdot (x_k^{(j_l)} - \bar{x}^{(j_l)}) \right)^2 \quad (6.10)$$

Die andere notwendige Bedingung für (lokale) Extrema von  $E$  lautet

$$\frac{\partial E}{\partial a_r} = -\frac{2}{n} \sum_{k=1}^n (x_k^{(j_r)} - \bar{x}^{(j_r)}) \left( x_k^{(i)} - \bar{x}^{(i)} - \sum_{l=1}^m a_l \cdot (x_k^{(j_l)} - \bar{x}^{(j_l)}) \right) = 0 \quad (6.11)$$

$r = 1, \dots, m$ , was als lineares Gleichungssystem geschrieben werden kann:

$$\sum_{l=1}^m a_l \sum_{k=1}^n (x_k^{(j_l)} - \bar{x}^{(j_l)}) (x_k^{(j_r)} - \bar{x}^{(j_r)}) = \sum_{k=1}^n (x_k^{(i)} - \bar{x}^{(i)}) (x_k^{(j_r)} - \bar{x}^{(j_r)}) \quad (6.12)$$

$$\Leftrightarrow \sum_{l=1}^m a_l c_{j_l j_r} = c_{i j_r}, \quad r = 1, \dots, m \quad (6.13)$$

Dieses lineare Gleichungssystem kann z. B. mit dem Gaußschen Eliminationsverfahren oder der Cramerschen Regel gelöst werden. Für die multiple lineare Regression können also alle Parameter der Regressionsmodelle mit Hilfe der Mittelwerte und der Kovarianzmatrix von  $X$  berechnet werden.

Ein äquivalentes Ergebnis erhalten wir, wenn wir die lineare Regression (6.7) in Matrixform schreiben. Mit (6.9) erhalten wir

$$Y = \begin{pmatrix} x_1^{(i)} - \bar{x}^{(i)} \\ \vdots \\ x_n^{(i)} - \bar{x}^{(i)} \end{pmatrix} \quad X = \begin{pmatrix} x_1^{(j_1)} - \bar{x}^{(j_1)} & \dots & x_1^{(j_m)} - \bar{x}^{(j_m)} \\ \vdots & \ddots & \vdots \\ x_n^{(j_1)} - \bar{x}^{(j_1)} & \dots & x_n^{(j_m)} - \bar{x}^{(j_m)} \end{pmatrix} \quad A = \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \quad (6.14)$$

und schreiben (6.7) als

$$Y = X \cdot A \quad (6.15)$$

$$X^T \cdot Y = X^T \cdot X \cdot A \quad (6.16)$$

$$(X^T \cdot X)^{-1} \cdot X^T \cdot Y = A \quad (6.17)$$

Der Ausdruck  $(X^T \cdot X)^{-1} \cdot X^T$  heißt *Pseudoinverse* von  $X$ . Die Matrix der Regressionsparameter  $A$  kann also als Produkt der Pseudoinversen von  $X$  und der Matrix  $Y$  berechnet werden.

Als Beispiel für die multiple lineare Regression betrachten wir den Datensatz

$$X = \begin{pmatrix} 6 & 4 & -2 \\ 2 & 1 & -1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 2 & 4 & 2 \end{pmatrix} \quad (6.18)$$

Gesucht sei eine lineare Funktion  $x^{(1)} = f(x^{(2)}, x^{(3)})$ , also

$$x_k^{(1)} \approx \bar{x}^{(1)} + a_1(x_k^{(2)} - \bar{x}^{(2)}) + a_2(x_k^{(3)} - \bar{x}^{(3)}) \quad (6.19)$$

Die Mittelwerte der Merkmale sind

$$\bar{x}^{(1)} = \frac{6+2+2}{5} = 2 \quad (6.20)$$

$$\bar{x}^{(2)} = \frac{4+1+1+4}{5} = 2 \quad (6.21)$$

$$\bar{x}^{(3)} = \frac{-2-1+1+2}{5} = 0 \quad (6.22)$$

Die Kovarianzmatrix von  $X$  lautet

$$C = \begin{pmatrix} 6 & 3.5 & -2.5 \\ 3.5 & 3.5 & 0 \\ -2.5 & 0 & 2.5 \end{pmatrix} \quad (6.23)$$

Das lineare Gleichungssystem zur Bestimmung von  $a_1$  und  $a_2$  ist somit

$$c_{22}a_1 + c_{32}a_2 = c_{12} \quad (6.24)$$

$$c_{23}a_1 + c_{33}a_2 = c_{13} \quad (6.25)$$

$$(6.24) \Leftrightarrow 3.5a_1 = 3.5 \Leftrightarrow a_1 = 1 \quad (6.26)$$

$$(6.25) \Leftrightarrow 2.5a_2 = -2.5 \Leftrightarrow a_2 = -1 \quad (6.27)$$

Wir erhalten also die multiple lineare Regressionsfunktion

$$x_k^{(1)} \approx 2 + (x_k^{(2)} - 2) - (x_k^{(3)} - 0) = x_k^{(2)} - x_k^{(3)} \quad (6.28)$$

Durch Einsetzen von  $X$  (6.18) in (6.28) sehen wir, dass der Approximationfehler in diesem Fall gleich null ist. In der Regel erhalten wir jedoch Regressionsmodelle mit Approximationfehlern ungleich null.

Mit dem Ansatz der Pseudoinverse erhalten wir alternativ

$$Y = \begin{pmatrix} 6 - 2 \\ 2 - 2 \\ 0 - 2 \\ 0 - 2 \\ 2 - 2 \end{pmatrix} = \begin{pmatrix} 4 \\ 0 \\ -2 \\ -2 \\ 0 \end{pmatrix} \quad X = \begin{pmatrix} 4 - 2 - 2 - 0 \\ 1 - 2 - 1 - 0 \\ 0 - 2 \quad 0 - 0 \\ 1 - 2 \quad 1 - 0 \\ 4 - 2 \quad 2 - 0 \end{pmatrix} = \begin{pmatrix} 2 - 2 \\ -1 - 1 \\ -2 \quad 0 \\ -1 \quad 1 \\ 2 \quad 2 \end{pmatrix} \quad (6.29)$$

und damit

$$\begin{aligned} A &= (X^T \cdot X)^{-1} \cdot X^T \cdot Y \\ &= \left( \begin{pmatrix} 2 - 2 \\ -1 - 1 \\ -2 \quad 0 \\ -1 \quad 1 \\ 2 \quad 2 \end{pmatrix} \right)^{-1} \left( \begin{pmatrix} 4 \\ 0 \\ -2 \\ -2 \\ 0 \end{pmatrix} \right) \\ &= \left( \begin{pmatrix} 14 & 0 \\ 0 & 10 \end{pmatrix} \right)^{-1} \left( \begin{pmatrix} 14 \\ 10 \end{pmatrix} \right) = \left( \begin{pmatrix} \frac{1}{14} & 0 \\ 0 & \frac{1}{10} \end{pmatrix} \right) \left( \begin{pmatrix} 14 \\ 10 \end{pmatrix} \right) = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \end{aligned} \quad (6.30)$$

was der oben gefundenen Lösung  $a_1 = 1, a_2 = -1$  entspricht.

## 6.2 Lineare Regression mit nichtlinearer Substitution

Nichtlineare Regressionsmodelle mit *vorgegebenen* nichtlinearen Funktionen lassen sich effizient mit Methoden der linearen Regression bestimmen. Ist beispielsweise bekannt, dass der Luftwiderstand quadratisch mit der Geschwindigkeit zunimmt, so kann statt der Geschwindigkeit das Quadrat der Geschwindigkeit als Merkmal betrachtet und der zugehörige Koeffizient dann mit linearer Regression bestimmt werden. Ein anderes Beispiel für die lineare Regression mit nichtlinearer Substitution ist die *polynomielle Regression*. Zur Schätzung eines Merkmals  $x$  aus einem Merkmal  $y$  durch polynomielle Regression der Ordnung  $q \in \{2, 3, \dots\}$  berechnen wir zusätzlich die Merkmale

$$x^2, \dots, x^q \quad (6.31)$$

Die lineare Regression mit den  $q$  Merkmalen  $x, x^2, \dots, x^q$  liefert dann die Koeffizienten  $a_0, a_1, \dots, a_p$  der Polynomfunktion

$$y \approx f(x) = \sum_{i=0}^p a_i x^i \quad (6.32)$$

### 6.3 Robuste Regression

Die quadratische Fehlerfunktion der gewöhnlichen linearen Regression ist sehr empfindlich gegenüber Ausreißern, da diese den Fehler besonders stark beeinflussen. *Robuste* Fehlerfunktionen versuchen, den Einfluss von Ausreißern zu verringern. Die lineare Regression mit robusten Fehlerfunktionen wird auch *robuste lineare Regression* genannt. Ein Beispiel für eine robuste Fehlerfunktion ist die Funktion von Huber, bei der die Fehler nur bis zu einer Grenze  $\varepsilon > 0$  quadriert und danach linear extrapoliert werden [5].

$$E_H = \sum_{k=1}^n \begin{cases} e_k^2 & \text{falls } e_k < \varepsilon \\ 2\varepsilon \cdot e_k - \varepsilon^2 & \text{sonst} \end{cases} \quad (6.33)$$

Ein anderes Beispiel für eine robuste Fehlerfunktion ist die *Summe der kleinsten quadratischen Fehler* (engl. *Least Trimmed Squares, LTS*) [10] bei der die quadratischen Fehler sortiert

$$e'_1 \leq e'_2 \leq \dots \leq e'_n \quad (6.34)$$

und nur die  $m$  kleinsten Fehler berücksichtigt werden,  $1 \leq m \leq n$ :

$$E_{LTS} = \sum_{k=1}^m e_k'^2 \quad (6.35)$$

### 6.4 Neuronale Netze

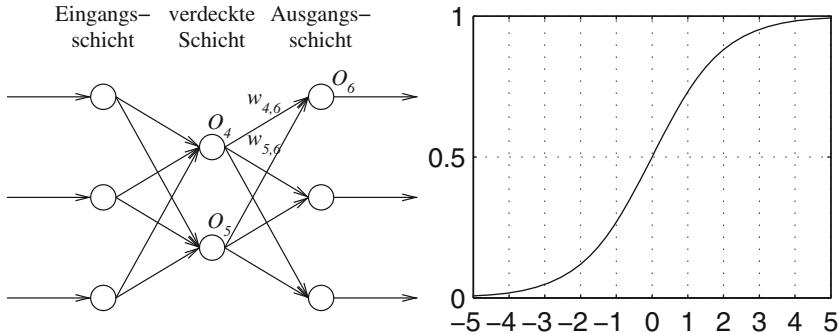
Eine wichtige Klasse nichtlinearer Regressionsmodelle sind die sogenannten *universellen Approximatoren* [4, 6]. Sei  $f$  eine stetige reellwertige Funktion  $f$  über einer kompakten Menge  $U \subset \mathbb{R}^n$

$$f: U \rightarrow \mathbb{R} \quad (6.36)$$

Eine Klasse  $F$  solcher Funktionen  $f$  heißt universeller Approximator genau dann, wenn es für jedes  $\varepsilon > 0$  eine Funktion  $f^* \in F$  gibt, so dass

$$|f(x) - f^*(x)| < \varepsilon \quad (6.37)$$

für alle  $x \in U$ . In diesem und dem folgenden Abschnitt werden zwei Beispiele für universelle Approximatoren vorgestellt: das *mehrschichtige Perzeptron* und *Netze mit radialen Basisfunktionen*.



**Abb. 6.2** Mehrschichtiges Perzeptron und Sigmoid-Funktion

Ein *mehrschichtiges Perzeptron* (engl. *Multi Layer Perceptron, MLP*) [9] ist ein (künstliches) *neuronales Netz* [3], das wie in Abb. 6.2 (links) als gerichteter Graph dargestellt werden kann. Aufgrund der strukturellen Ähnlichkeit der künstlichen neuronalen Netze mit biologischen neuronalen Netzen werden die Knoten *Neuronen* genannt. Die Neuronen werden oft nicht vollständig vernetzt, sondern in mehreren Schichten angeordnet und von jeder Schicht zur Folgeschicht verbunden. Die Eingänge des neuronalen Netzes sind die Eingänge der Neuronen der ersten Schicht (Eingangsschicht), und die Ausgänge des neuronalen Netzes sind die Ausgänge der Neuronen der letzten Schicht (Ausgangsschicht). Die Schichten zwischen der Eingangsschicht und der Ausgangsschicht heißen *verdeckte Schichten*. Das neuronale Netz in Abb. 6.2 (links) besitzt drei Schichten: eine Eingangsschicht, eine Ausgangsschicht und eine verdeckte Schicht. Über jede Kante eines solchen neuronalen Netzes fließt ein skalarer Wert. Ein neuronales Netz mit  $p$  Neuronen in der Eingangsschicht und  $q$  Neuronen in der Ausgangsschicht realisiert also eine Funktion  $f: \mathbb{R}^p \rightarrow \mathbb{R}^q$ . Das Beispiel in Abb. 6.2 (links) realisiert eine Funktion  $f: \mathbb{R}^3 \rightarrow \mathbb{R}^3$ . Jede gerichtete Kante von Neuron  $i$  zu Neuron  $j$  besitzt ein Kantengewicht  $w_{ij} \in \mathbb{R}$ . Zur Vereinfachung stellen wir fehlende Kanten durch Kanten mit Gewicht null dar. Jedes Neuron  $i$  liefert einen skalaren Ausgangswert  $O_i \in \mathbb{R}$ . Der „effektive“ Eingangswert jedes Neurons  $i$  ist die gewichtete Summe

$$I_i = \sum_j w_{ji} O_j \quad (6.38)$$

Der Ausgangswert jedes Neurons ergibt sich aus dem Eingangswert durch eine nichtlineare Aktivierungsfunktion  $s: \mathbb{R} \rightarrow \mathbb{R}$ .

$$O_i = s(I_i) \quad (6.39)$$

Im Einklang mit dem experimentell gemessenen Verhalten von biologischen Neuronen werden als Aktivierungsfunktion häufig *sigmoide* (s-förmige) Funktionen verwendet, z. B. die *logistische* Funktion

$$s(x) = \frac{1}{1 + e^{-x}} \in (0, 1) \quad (6.40)$$

oder die hyperbolische Tangensfunktion

$$s(x) = \tanh x \in (-1, 1) \quad (6.41)$$

Sigmoidfunktionen bilden den unbeschränkten Wertebereich der möglichen Neuroneneingänge auf einen beschränkten Wertebereich der Neuronenausgänge ab (siehe die Datentransformationen in Kap. 3). Für einen gegebenen Eingangsvektor kann der Ausgangsvektor des mehrschichtigen Perzeptrons mit den Formeln (6.38), (6.39) und (6.40) oder (6.41) berechnet werden. Die durch das mehrschichtige Perzeptron realisierte Funktion  $f: \mathbb{R}^p \rightarrow \mathbb{R}^q$  wird durch die Gewichte  $w_{ij}$  parametriert. In der Regression werden die Gewichte  $w_{ij}$  aus dem Ein-/Ausgabedatensatz  $Z = \{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathbb{R}^{p+q}$  bestimmt, so dass sich für jeden Eingangsvektor  $x_k$ ,  $k = 1, \dots, n$ , näherungsweise der gewünschte Ausgangsvektor  $y_k$  ergibt,  $y_k \approx f(x_k)$ ,  $(x_k, y_k) \in Z$ . Die Schätzung der Parameter  $w_{ij}$  aus  $Z$  wird auch *Training*, und  $Z$  wird auch *Trainings-Datensatz* genannt. Zur Beschreibung des Trainingsprozesses betrachten wir zunächst ein dreischichtiges Perzeptron mit  $p$  Neuronen in der Eingangsschicht,  $h$  Neuronen in der verdeckten Schicht und  $q$  Neuronen in der Ausgangsschicht. Das Netzwerk hat also  $p + h + q$  Neuronen und  $p \cdot h + h \cdot q$  Kanten. Wir nummerieren die Neuronen so, dass die Neuronen  $1, \dots, p$  in der Eingangsschicht, die Neuronen  $p + 1, \dots, p + h$  in der verdeckten Schicht und die Neuronen  $p + h + 1, \dots, p + h + q$  in der Ausgangsschicht liegen. Der Netzwerkeingang ist also  $x = (I_1, \dots, I_p) \in \mathbb{R}^p$ , und der Netzwerkausgang ist  $f(x) = (O_{p+h+1}, \dots, O_{p+h+q}) \in \mathbb{R}^q$ . Für einen gegebenen Eingangsvektor  $x_k$  liefert das Netz den Ausgangsvektor  $f(x_k)$ , der näherungsweise gleich dem gewünschten Ausgangsvektor  $y_k$  sein sollte. Zum Training des Netzes minimieren wir also den durchschnittlichen quadratischen Fehler zwischen  $f(x_k)$  und  $y_k$ . Mit  $y_k = (O'_{p+h+1}, \dots, O'_{p+h+q})$  schreiben wir diesen Fehler als

$$E = \frac{1}{q} \cdot \sum_{i=p+h+1}^{p+h+q} (O_i - O'_i)^2 \quad (6.42)$$

Die Gewichte  $w_{ij}$  werden zufällig initialisiert und dann iterativ mit dem Gradientenverfahren aktualisiert:

$$w_{ij} - \alpha(t) \cdot \frac{\partial E}{\partial w_{ij}} \quad (6.43)$$

Die Schrittweite  $\alpha$  wird hier auch *Lernrate* genannt. Mit Hilfe der Kettenregel können die Fehlergradienten aus (6.38), (6.39) und (6.42) berechnet werden.

$$\begin{aligned} \frac{\partial E}{\partial w_{ij}} &= \frac{\partial E}{\partial O_j} \cdot \frac{\partial O_j}{\partial I_j} \cdot \frac{\partial I_j}{\partial w_{ij}} \\ &\sim \underbrace{(O_j - O'_j) \cdot s'(I_j)}_{= \delta_j^{(O)}} \cdot O_i \end{aligned} \quad (6.44)$$

Der Ausdruck  $\delta_j^{(O)}$ ,  $j = p+q+1, \dots, p+q+r$ , wird auch *Deltawert* der Ausgangsschicht genannt. Mit diesem Deltawert kann die Aktualisierungsvorschrift (6.43) als sogenannte *Deltaregel* [11] geschrieben werden:

$$w_{ij} - \alpha(t) \cdot \delta_j^{(O)} \cdot O_i \quad (6.45)$$

Für die logistische Funktion (6.40) erhalten wir für die Ableitung  $s'(I_j)$  den Ausdruck

$$\begin{aligned} s'(I_j) &= \frac{\partial}{\partial I_j} \frac{1}{1 + e^{-I_j}} = -\frac{1}{(1 + e^{-I_j})^2} \cdot (-e^{-I_j}) \\ &= \frac{1}{1 + e^{-I_j}} \cdot \frac{e^{-I_j}}{1 + e^{-I_j}} = O_j \cdot (1 - O_j) \end{aligned} \quad (6.46)$$

Durch Einsetzen in (6.44) erhalten wir die Deltawerte für die logistische Funktion, der sich bequem aus den Ist- und Soll-Ausgangswerten der einzelnen Neuronen der Ausgangsschicht berechnen lässt:

$$\delta_j^{(O)} = (O_j - O'_j) \cdot O_j \cdot (1 - O_j) \quad (6.47)$$

Zur Aktualisierung der Gewichte zwischen Eingangsschicht und verdeckter Schicht muss die Summe der Ableitungen aller Ausgangswerte berücksichtigt werden.

$$\begin{aligned} \frac{\partial E}{\partial w_{ij}} &= \sum_{l=p+h+1}^{p+h+q} \frac{\partial E}{\partial O_l} \cdot \frac{\partial O_l}{\partial I_l} \cdot \frac{\partial I_l}{\partial O_j} \cdot \frac{\partial O_j}{\partial I_j} \cdot \frac{\partial I_j}{\partial w_{ij}} \\ &\sim \sum_{l=p+h+1}^{p+h+q} (O_l - O'_l) \cdot s'(I_l) \cdot w_{jl} \cdot s'(I_j) \cdot O_i \\ &= \sum_{l=p+h+1}^{p+h+q} \delta_l^{(O)} \cdot w_{jl} \cdot s'(I_j) \cdot O_i \end{aligned} \quad (6.48)$$

In Anlehnung an (6.44) schreiben wir diesen Gradienten als

$$\begin{aligned} \frac{\partial E}{\partial w_{ij}} &= s'(I_j) \cdot \underbrace{\sum_{l=p+h+1}^{p+h+q} \delta_l^{(O)} \cdot w_{jl} \cdot O_i}_{= \delta_j^{(H)}} \\ &= \delta_j^{(H)} \end{aligned} \quad (6.49)$$

und erhalten entsprechend zu (6.45) die Deltaregel für die Gewichte zwischen Eingangsschicht und verdeckter Schicht:

$$w_{ij} - \alpha(t) \cdot \delta_j^{(H)} \cdot O_i \quad (6.50)$$

Für die logistische Funktion erhalten wir mit (6.46) und (6.49)

$$\delta_j^{(H)} = (O_j) \cdot (1 - O_j) \cdot \sum_{l=p+q+1}^{p+q+r} \delta_l^{(O)} \cdot w_{jl} \quad (6.51)$$

1. Eingabe: Anzahl Neuronen  $p, h, q \in \{1, 2, \dots\}$ ,  
Eingangs-Trainingsdaten  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$ ,  
Ausgangs-Trainingsdaten  $Y = \{y_1, \dots, y_n\} \subset \mathbb{R}^q$ ,  
Lernrate  $\alpha(t)$
2. Initialisiere  $w_{ij}$  für  $i = 1, \dots, p$ ,  $j = p+1, \dots, p+h$   
und für  $i = p+1, \dots, p+h$ ,  $j = p+h+1, \dots, p+h+q$
3. Für jedes Ein-/Ausgangsdatenpaar  $(x_k, y_k)$ ,  $k = 1, \dots, n$ ,
  - a. Aktualisiere die Gewichte der Ausgangsschicht
$$w_{ij} = w_{ij} - \alpha(t) \cdot \delta_j^{(O)} \cdot O_i, \quad \begin{matrix} i = p+1, \dots, p+h \\ j = p+h+1, \dots, p+h+q \end{matrix}$$
  - b. Aktualisiere die Gewichte der verdeckten Schicht
$$w_{ij} = w_{ij} - \alpha(t) \cdot \delta_j^{(H)} \cdot O_i, \quad \begin{matrix} i = 1, \dots, p \\ j = p+1, \dots, p+h \end{matrix}$$
4. Wiederhole ab (3.) bis Terminierungskriterium erfüllt
5. Ausgabe: Gewichte  $w_{ij}$ ,  $i = 1, \dots, p$ ,  $j = p+1, \dots, p+h$   
und  $i = p+1, \dots, p+h$ ,  $j = p+h+1, \dots, p+h+q$

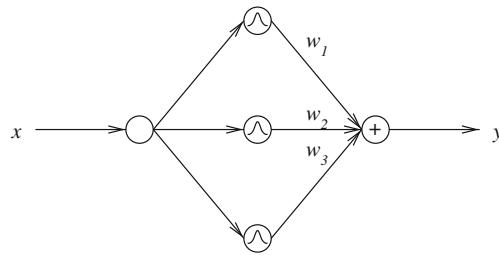
**Abb. 6.3** Backpropagation-Algorithmus für dreischichtiges Perzeptron

Für ein Perzeptron mit mehr als drei Schichten erhalten wir entsprechende Deltaregeln: Die Deltawerte werden aus den Neuronenausgängen der betrachteten Schicht und den Deltawerten der folgenden Schicht berechnet. Dieses Prinzip wird auch *generalisierte Deltaregel* genannt. Die Gewichte des neuronalen Netzes werden schichtweise von der Ausgangsschicht bis zur Eingangsschicht aktualisiert. Dieses Schema wird auch *Zurückpropagieren* (engl. *Backpropagation*) [14] genannt. Die Gewichtsaktualisierung durch Zurückpropagieren wird für jedes Paar von Ein- und Ausgangsvektoren durchgeführt. Ein Durchlauf durch alle Daten wird *Lernepochen* genannt. Der Algorithmus terminiert nach einer vorgegebenen Anzahl von Lernepochen. Abbildung 6.3 fasst den Backpropagation-Algorithmus für ein dreischichtiges Perzeptron zusammen. Nach dem Training kann das Perzeptron als nichtlineare Regressionsfunktion eingesetzt werden, um Ausgangswerte für bisher unbekannte Eingangswerte zu schätzen. Dies wird auch *Generalisierung* oder engl. *Recall* genannt.

## 6.5 Radiale Basisfunktionen

Als ein zweites Beispiel für universelle Approximatoren betrachten wir *Netzwerke mit radialen Basisfunktionen (RBF)* [8]. Jedes Neuron eines RBF-Netzes realisiert eine einzelne radiale Basisfunktion mit dem Argument  $\|x - \mu_i\|$ . Hierzu werden häufig Gauß-Funktionen verwendet:

$$u_i(x) = e^{-\left(\frac{\|x - \mu_i\|}{\sigma_i}\right)^2} \quad (6.52)$$



**Abb. 6.4** RBF-Netz mit drei radialen Basisfunktionen

$\mu_i \in \mathbb{R}^p$ ,  $\sigma_i > 0$ ,  $i = 1, \dots, c$ . Abbildung 6.4 zeigt ein RBF-Netz mit drei Neuronen und entsprechend drei Basisfunktionen. Jedes Neuron ist mit dem Eingang  $x \in \mathbb{R}^p$  und über das Gewicht  $w_i \in \mathbb{R}$  mit dem additiven Ausgangsneuron verbunden. Im Unterschied zur skalaren Struktur des mehrschichtigen Perzeptrons erlauben wir bei RBF-Netzen vektorförmige Daten entlang der Kanten. Der Ausgang eines Gaußschen RBF-Netzes ergibt sich als

$$y = \sum_{i=1}^c w_i \cdot e^{-\left(\frac{\|x-\mu_i\|}{\sigma_i}\right)^2} \quad (6.53)$$

Das RBF-Netz realisiert also eine nichtlineare Funktion, die aus  $c$  gewichteten lokalen linearen Funktionen zusammengesetzt ist. Der Gültigkeitsbereich jeder lokalen Funktion wird durch die entsprechende Basisfunktion festgelegt. Die Funktion des RBF-Netzes wird durch zwei Sätze von Parametern bestimmt: die Parameter der Basisfunktionen (Mittelwerte  $\mu_1, \dots, \mu_c$  und Standardabweichungen  $\sigma_1, \dots, \sigma_c$  für Gauß-Funktionen) und die Gewichte  $w_1, \dots, w_c$ . Die RBF-Parameter werden oft durch Clusteranalyse bestimmt (siehe Kap. 9). Wir stellen hier stattdessen eine *alternierende Optimierung* vor, bei der abwechselnd ein gradientenbasierter Optimierungsschritt der RBF-Parameter und eine Optimierung der Gewichte mit Hilfe der Pseudoinversen durchgeführt wird. Der durchschnittliche quadratische Fehler des RBF-Netzes ist

$$E = \frac{1}{n} \sum_{k=1}^n \left( \sum_{i=1}^c w_i e^{-\left(\frac{\|x_k-\mu_i\|}{\sigma_i}\right)^2} - y_k \right)^2 \quad (6.54)$$

Dies liefert die Fehlergradienten für den Gradientenabstieg der RBF-Parameter:

$$\frac{\partial E}{\partial \mu_i} = \frac{4w_i}{n\sigma_i^2} \sum_{k=1}^n \left( \sum_{j=1}^m w_j e^{-\left(\frac{\|x_k-\mu_j\|}{\sigma_j}\right)^2} - y_k \right) \|x_k - \mu_i\| e^{-\left(\frac{\|x_k-\mu_i\|}{\sigma_i}\right)^2} \quad (6.55)$$

$$\frac{\partial E}{\partial \sigma_i} = \frac{4w_i}{n\sigma_i^3} \sum_{k=1}^n \left( \sum_{j=1}^m w_j e^{-\left(\frac{\|x_k-\mu_j\|}{\sigma_j}\right)^2} - y_k \right) \|x_k - \mu_i\|^2 e^{-\left(\frac{\|x_k-\mu_i\|}{\sigma_i}\right)^2} \quad (6.56)$$

Zur Bestimmung des Vektors  $W = (w_1 \dots w_c)^T$  der optimalen Gewichtswerte zu den Soll-Ausgangswerten  $Y = (y_1 \dots y_n)^T$  schreiben wir die Ausgangswerte der verdeckten

1. Eingabe: Anzahl der Basisfunktionen  $c \in \{1, 2, \dots\}$ ,  
Eingangs-Trainingsdaten  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$ ,  
Ausgangs-Trainingsdaten  $Y = \{y_1, \dots, y_n\} \subset \mathbb{R}^r$ ,  
Lernrate  $\alpha(t)$
2. Initialisiere die Gewichte  $w_1, \dots, w_c \in \mathbb{R}$
3. Für jedes Ein-/Ausgangsdatenpaar  $(x_k, y_k)$ ,  $k = 1, \dots, n$ ,  
Aktualisieren die RBF-Parameter

$$\mu_i = \mu_i - \alpha(t) \cdot \frac{\partial E}{\partial \mu_i}, \quad i = 1, \dots, c \quad (6.55)$$

$$\sigma_i = \sigma_i - \alpha(t) \cdot \frac{\partial E}{\partial \sigma_i}, \quad i = 1, \dots, c \quad (6.56)$$

- bis Terminierungsbedingung erfüllt
4. Berechne die Matrix der Ausgangswerte der verdeckten Schicht

$$U = \begin{pmatrix} e^{-\left(\frac{x_1-\mu_1}{\sigma_1}\right)^2} & \dots & e^{-\left(\frac{x_1-\mu_c}{\sigma_c}\right)^2} \\ \vdots & & \vdots \\ e^{-\left(\frac{x_n-\mu_1}{\sigma_1}\right)^2} & \dots & e^{-\left(\frac{x_n-\mu_c}{\sigma_c}\right)^2} \end{pmatrix}$$

5. Berechne die optimalen Gewichte  $w_1, \dots, w_c \in \mathbb{R}$

$$W = (U^T \cdot U)^{-1} \cdot U^T \cdot Y$$

6. Wiederhole ab (3.), bis Terminierungsbedingung erfüllt
7. Ausgabe: RBF-Parameter  $\mu_i$ ,  $\sigma_i$  und Gewichte  $w_i$ ,  $i = 1, \dots, c$

**Abb. 6.5** RBF-Trainingsalgorithmus

Schicht als Matrix

$$U = \begin{pmatrix} e^{-\left(\frac{x_1-\mu_1}{\sigma_1}\right)^2} & \dots & e^{-\left(\frac{x_1-\mu_c}{\sigma_c}\right)^2} \\ \vdots & & \vdots \\ e^{-\left(\frac{x_n-\mu_1}{\sigma_1}\right)^2} & \dots & e^{-\left(\frac{x_n-\mu_c}{\sigma_c}\right)^2} \end{pmatrix} \quad (6.57)$$

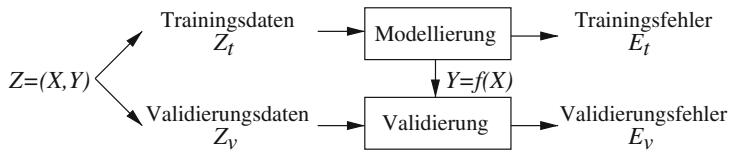
und erhalten durch lineare Regression mit dem Verfahren der Pseudoinversen

$$Y = U \cdot W \quad \Rightarrow \quad W = (U^T \cdot U)^{-1} \cdot U^T \cdot Y \quad (6.58)$$

Der entsprechende RBF-Trainingsalgorithmus ist in Abb. 6.5 dargestellt.

## 6.6 Kreuzvalidierung

Die Kreuzvalidierung ist eine Methode, um die Generalisierungsfähigkeit von datenbasier-ten Modellen (z. B. Regressionsmodellen) zu quantifizieren. Regressionsmodelle können die gegebenen Trainingsdaten oft mit sehr geringem oder sogar null Fehler abbilden, wenn

**Abb. 6.6** Kreuzvalidierung

genügend freie Modellparameter genutzt werden, z. B. wenn sehr viele Neuronen in der verdeckten Schicht oder sehr viele Basisfunktionen verwendet werden. Dies kann jedoch zu einer Überanpassung des Modells an die Daten (engl. *Overfitting*) führen, bei dem das Modell zwar die benutzten Trainingsdaten gut, aber den zugrundeliegenden Zusammenhang schlecht abbildet, und somit nicht gut genug auf andere Daten generalisieren kann. Eine solche Überanpassung kann durch Kreuzvalidierung verhindert werden. Hierzu wird der verfügbare Ein-/Ausgangsdatensatz  $Z = \{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathbb{R}^{p+q}$  in einen Trainingsdatensatz  $Z_t \subset Z$  und einen davon disjunkten Validierungsdatensatz  $Z_v \subset Z$  aufgeteilt,  $Z_t \cap Z_v = \{\}$ ,  $Z_t \cup Z_v = Z$ . Der Trainingsdatensatz wird verwendet, um das Modell  $f$  zu erstellen (trainieren), und der Validierungsdatensatz wird verwendet, um dieses Modell zu validieren (siehe Abb. 6.6). Für Regressionsprobleme lautet der durchschnittliche quadratische Trainingsfehler

$$E_t = \frac{1}{|Z_t|} \sum_{(x, y) \in Z_t} \|y - f(x)\|^2 \quad (6.59)$$

und der durchschnittliche quadratische Validierungsfehler

$$E_v = \frac{1}{|Z_v|} \sum_{(x, y) \in Z_v} \|y - f(x)\|^2 \quad (6.60)$$

Da die Validierungsdaten bei der Modellerstellung nicht berücksichtigt werden, deutet ein niedriger Validierungsfehler auf eine gute Generalisierungsfähigkeit des Modells hin.

Bei der  $k$ -fachen Kreuzvalidierung wird der Datensatz  $Z$  zufällig in  $k$  paarweise disjunkte und (näherungsweise) gleich große Teilmengen  $Z_1, \dots, Z_k$  aufgeteilt,  $Z_i \cap Z_j = \{\}$  für alle  $i \neq j$ ,  $Z_1 \cup \dots \cup Z_k = Z$  und  $|Z_i| \approx |Z_j|$  für alle  $i, j = 1, \dots, k$ . Jede der Teilmengen  $Z_i$ ,  $i = 1, \dots, k$ , wird benutzt, um das Modell  $f_i$  zu validieren, das auf Basis der übrigen  $k - 1$  Teilmengen  $Z_j$ ,  $j = 1, \dots, k$ ,  $j \neq i$ , erstellt wurde. Bei der  $k$ -fachen Kreuzvalidierung werden also  $k$  Modelle trainiert, und jeder Datenvektor wird einmal zur Validierung und  $k - 1$  mal zum Training verwendet. Dies liefert  $k$  Validierungsfehler  $E_{v1}, \dots, E_{vk}$  für die  $k$  Modelle  $f_1, \dots, f_k$

$$E_{vi} = \frac{1}{|Z_i|} \sum_{(x, y) \in Z_i} \|y - f_i(x)\|^2 \quad (6.61)$$

$i = 1, \dots, k$ , deren Mittelwert *k-facher Kreuzvalidierungsfehler* genannt wird:

$$E_v = \frac{1}{k} \sum_{i=1}^k E_{vi} \quad (6.62)$$

Dieser Fehler ist der Mittelwert der Validierungsfehler *unterschiedlicher* Modelle, die mit der gleichen Methode aus unterschiedlichen Daten gefunden werden. Streng genommen quantifiziert der  $k$ -fache Kreuzvalidierungsfehler daher nicht die Generalisierungsfähigkeit der einzelnen Modelle, sondern der Modellierungsmethode.

Die sogenannte *Leave-One-Out-Kreuzvalidierung* verwendet zur Validierung jedes einzelnen Modells nur ein einziges Ein-/Ausgangsdatenpaar und kann daher auch als  $n$ -fache Kreuzvalidierung bezeichnet werden.

In der Regel ist der Validierungsfehler größer als der Trainingsfehler,  $E_v > E_t$ , da die Modelle auf Basis der Trainingsdaten und nicht der Validierungsdaten erstellt werden. Der Zusammenhang zwischen Trainingsfehler und Validierungsfehler kann mit den folgenden Formeln näherungsweise abgeschätzt werden [1, 2]:

$$E_v \approx \frac{1 + d/n}{1 - d/n} E_t \quad (6.63)$$

$$E_v \approx (1 + 2d/n) E_t \quad (6.64)$$

$$E_v \approx \frac{1}{(1 - d/n)^2} E_t \quad (6.65)$$

wobei  $d$  die Anzahl der freien Modellparameter ist, z. B. die Anzahl der Kantengewichte eines Perzeptrons. Bei Modellen mit guter Generalisierungsfähigkeit entspricht der Validierungsfehler ungefähr dem Trainingsfehler,  $E_v \approx E_t$ . Mit den drei obigen Abschätzungen wird dies für  $d \ll n$  erreicht. Um geeignete Werte von  $d$  zu finden, kann  $d$  sukzessive erhöht werden, bis Trainings- und Validierungsfehler auseinander zu laufen beginnen, oder  $d$  kann sukzessive verringert werden, bis Trainings- und Validierungsfehler sich ausreichend angenähert haben.

Ein mit Hilfe der Kreuzvalidierung gefundenes Modell kann als bestmögliche modellbasierte Approximation der Daten interpretiert werden. Wenn die Ein- und Ausgangsdaten stark korreliert sind, dann ist ein geringer Validierungsfehler erzielbar. Wenn die Ein- und Ausgangsdaten dagegen schwach korreliert sind, dann muss mit einem hohen Validierungsfehler gerechnet werden. Der inverse Validierungsfehler kann daher auch als ein Maß für die Korrelation interpretiert werden, als Alternative zu den im vorigen Kapitel vorgestellten Verfahren der Korrelationsanalyse.

## 6.7 Merkmalsselektion

In den vorangegangenen Abschnitten haben wir angenommen, dass die Ausgangsmerkmale  $Y$  von allen Eingangsmerkmalen  $X$  abhängen, und haben daher alle Merkmale in  $X$  bei der Modellierung berücksichtigt. Oft enthält  $X$  jedoch auch Merkmale, die für  $Y$  nicht

relevant sind. Die Berücksichtigung irrelevanter Merkmale kann zu unnötig hoher Modellkomplexität führen (zu viele Modellparameter und zu hoher Rechenaufwand für das Modelltraining) oder sogar zu schlechterer Modellgüte, z. B. bei einer Überanpassung auf die irrelevanten Merkmale. Daher wird oft versucht, die relevanten Merkmale auszuwählen und nur diese zur Modellbildung zu nutzen, nicht nur bei der Regression, sondern auch bei Prognose, Klassifikation und Clusteranalyse (siehe folgende drei Kapitel). In Kap. 8 finden wir einen Entscheidungsbaum, der implizit die für eine gegebene Klassifikationsaufgabe relevanten Merkmale findet. Es kann jedoch prinzipiell jedes Regressions-, Prognose-, Klassifikations- oder Clusterverfahren mit einer Merkmalsselektion kombiniert werden. Hierzu werden die Modelle nur mit Teilmengen der verfügbaren Merkmale gebildet, und die Teilmengen werden sukzessive angepasst, bis eine zufriedenstellende Merkmalsauswahl gefunden ist. Ein einfacher Ansatz der Merkmalsselektion beginnt mit dem kompletten Satz an Merkmalen und entfernt nach und nach das jeweils am wenigsten relevante Merkmal, bis die Modellgüte einen vorgegebenen Schwellwert unterschreitet. Der umgekehrte Ansatz beginnt mit einer leeren Merkmalsmenge und fügt sukzessive die relevantesten Merkmale hinzu, bis keine signifikante Modellverbesserung mehr erzielt werden kann. Auf Grund der möglicherweise komplexen Zusammenhänge zwischen den Merkmalen können beide Ansätze zu ungünstigen Ergebnissen kommen. Wenn beispielsweise zwei Merkmale einzeln nur eine geringe, zusammen jedoch eine hohe Relevanz besitzen, so werden diese Merkmale beim zweiten Ansatz nicht berücksichtigt. Bessere Ergebnisse können oft durch Kombination beider Ansätze erzielt werden, indem die Merkmalsmengen abwechselnd erweitert und wieder reduziert werden. Um eine (bezüglich eines gegebenen Kriteriums) optimale Merkmalsmenge zu finden, müssen alle möglichen Merkmalskombinationen betrachtet werden, was auf Grund der großen Anzahl von Möglichkeiten nicht praktikabel ist. Eine Abhilfe schaffen hier stochastische Verfahren, die eine näherungsweise optimale Merkmalsauswahl finden. Hierzu gehören sogenannte Meta-Heuristiken wie *Simulated Annealing* [7], *evolutionäre Algorithmen* [12] oder Verfahren der *Schwarmintelligenz* [13]. Die Merkmalsselektion kann als achsenparallele Projektion der Daten aufgefasst werden (vgl. Kap. 4).

---

### Übungsaufgaben

**6.1.** Berechnen Sie die Kovarianzmatrix und die linearen Regressionsmodelle für den Datensatz  $X = \{(1, 1), (2, 2), (3, 3), (4, 4), (5, 5)\}$ .

**6.2.** Nun ändern Sie den letzten Punkt in  $X$  von  $(5, 5)$  auf  $(5, 0)$  und berechnen Sie erneut die Kovarianzmatrix und die linearen Regressionsmodelle.

**6.3.** Wie erklären Sie die unterschiedlichen Ergebnisse?

**6.4.** Erklären Sie den Unterschied zwischen Korrelation und Regression!

**6.5.** Wodurch kann ein kleiner Trainingsfehler und ein großer Validierungsfehler verursacht sein?

## Literatur

1. H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, AC-19:716–723, 1974.
2. P. Craven and G. Wahba. Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerical Mathematics*, 31:377–403, 1979.
3. R. Hecht-Nielsen. *Neurocomputing*. Addison-Wesley, 1990.
4. K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.
5. P. J. Huber. *Robust Statistics*. Wiley, New York, 2nd edition, 2009.
6. A. N. Kolmogorov. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. *Doklady Akademii Nauk SSSR*, 144:679–681, 1957.
7. R. Meiria and J. Zahavi. Using simulated annealing to optimize the feature selection problem in marketing applications. *European Journal of Operational Research*, 171(3):842–858, 2006.
8. J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1:281–294, 1989.
9. F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Reviews*, 65:386–408, 1958.
10. P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. Wiley, New York, 1987.
11. D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error backpropagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing. Explorations in the Microstructure of Cognition*, volume 1, pages 318–362. MIT Press, Cambridge, 1986.
12. W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10(5):335–347, 1989.
13. S. Vieira, J. M. Sousa, and T. A. Runkler. Multi-criteria ant feature selection using fuzzy classifiers. In C. A. Coello Coello, S. Deburi, and S. Ghosh, editors, *Swarm Intelligence for Multi-objective Problems in Data Mining*, pages 19–36. Springer, 2009.
14. P. J. Werbos. *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting (Adaptive and Learning Systems for Signal Processing, Communications and Control Series)*. Wiley-Interscience, 1994.

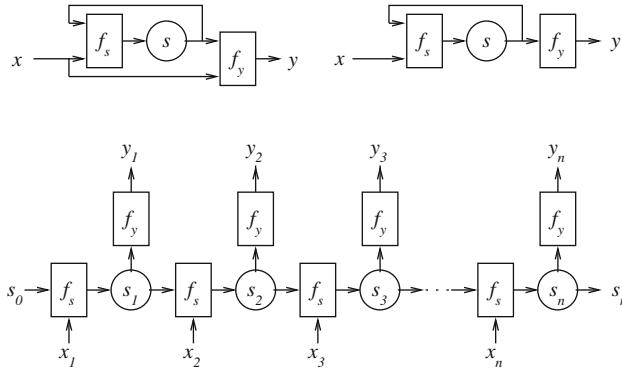
## Zusammenfassung

Zur Prognose zukünftiger Werte von Zeitreihen nehmen wir an, dass die Zeitreihe durch einen (möglicherweise verrauschten) deterministischen Prozess erzeugt wurde. Solche Prozesse können mit Mealy- oder Moore-Maschinen modelliert werden, wodurch sich rekurrente oder auto-regressive Modelle entwickeln lassen. Die Bestimmung der eigentlichen Prognosemodelle ist eine Regressionsaufgabe, bei der die Trainingsdaten durch endliche Entfaltung der Zeitreihe zusammengestellt werden. Zu den wichtigsten linearen Prognosemodellen gehören autoregressive Modelle (AR), generalisierte AR-Modelle mit gleitendem Mittelwert (ARMA) oder mit lokaler Regression (ARMAX). Zu den wichtigsten nichtlinearen Prognosemodellen gehören rekurrente neuronale Netze.

## 7.1 Endliche Zustandsautomaten

In der Datenanalyse betrachten wir häufig Zeitreihen. In Kap. 2 wurden Relationen für sequenzielle Daten und Abtastverfahren für kontinuierliche Signale dargestellt. In Kap. 3 wurde gezeigt, wie Zeitreihen durch Filterverfahren vorverarbeitet werden. In Kap. 4 wurde die Spektralanalyse zur Visualisierung von Zeitreihen behandelt. Die Zeitreihenanalyse betrachtet häufig nicht nur vergangene Daten, sondern versucht, zukünftige Daten möglichst zuverlässig vorherzusagen.

Zur Bestimmung eines geeigneten Prognosemodells nehmen wir an, dass die Zeitreihe durch ein deterministisches rückgekoppeltes dynamisches System mit Eingang  $x \in \mathbb{R}^p$ , internem Zustand  $s \in \mathbb{R}^h$  und Ausgang  $y \in \mathbb{R}^q$  erzeugt wird [2]. Diese Architektur besitzt eine gewisse Ähnlichkeit mit dem dreischichtigen Perzeptron aus dem vorigen Kapitel. Ein solches rückgekoppeltes dynamisches System kann zu einem Zeitschritt  $k$  beschrieben



**Abb. 7.1** Endliche Zustandsautomaten: Mealy-Maschine (oben links), Moore-Maschine (oben rechts), endlich entfaltete Moore-Maschine (unten)

werden durch eine Zustandsgleichung

$$s_k = f_s(s_{k-1}, x_k) \quad (7.1)$$

und eine Ausgangsgleichung

$$y_k = f_y(s_k, x_k) \quad (7.2)$$

Abbildung 7.1 (oben links) zeigt ein Blockdiagramm eines solchen Systems. Falls die Anzahl der Zustände, die ein solches System erreichen kann, endlich ist, so heißt das System *endlicher Zustandsautomat*, z. B. wenn  $s \in \{0, 1\}^r$ ,  $r \in \{1, 2, \dots\}$ . Gleichungen (7.1) und (7.2) beschreiben eine sogenannte *Mealy-Maschine* [3]. Sonderfälle von Mealy-Maschinen sind *Moore-Maschinen* [4], bei denen der Ausgang nur vom Zustand, aber nicht vom aktuellen Eingang abhängt. Eine Moore-Maschine kann also beschrieben werden durch die Zustandsgleichung

$$s_k = f_s(s_{k-1}, x_k) \quad (7.3)$$

und die Ausgangsgleichung

$$y_k = f_y(s_k) \quad (7.4)$$

Abbildung 7.1 (oben rechts) zeigt ein Blockdiagramm einer Moore-Maschine. Jede Mealy-Maschine kann in eine äquivalente Moore-Maschine übersetzt werden und umgekehrt. Mealy-Maschinen benötigen oft einen kleineren Zustandsraum als Moore-Maschinen, um ein bestimmtes Verhalten zu erzielen. Moore-Maschinen zeichnen sich gegenüber Mealy-Maschinen dadurch aus, dass ihr Entwurf oft leichter und ihr Verhalten einfacher zu analysieren ist. Wegen der funktionalen Äquivalenz zwischen Mealy-Maschinen und Moore-Maschinen beschränken wir uns hier auf Moore-Maschinen.

## 7.2 Rekurrente Modelle

Die Prognose nutzt die beobachteten Eingangs- und Ausgangssequenzen  $\{x_1, \dots, x_n\}$  und  $\{y_1, \dots, y_n\}$  zur Vorhersage der erwarteten Ausgangssequenz  $\{y_{n+1}, \dots, y_{n+q}\}$ , ohne dass die erwartete Eingangssequenz  $\{x_{n+1}, \dots, x_{n+q}\}$  bekannt ist. Für die Prognose müssen die Funktionen  $f_s$  und  $f_y$  gefunden werden, die die beobachteten Daten am besten approximieren, z. B. durch Minimierung des durchschnittlichen quadratischen Fehlers zwischen den Beobachtungen  $y_k$  und den Vorhersagen  $y'_k$ .

$$E = \frac{1}{n} \sum_{k=1}^n (y_k - y'_k)^2 \quad (7.5)$$

Dies führt zu einem Regressionsproblem (wie im letzten Kapitel beschrieben), bei dem die Funktionen  $f_s$  und  $f_y$  aus den Quadrupeln  $(x_k, s_{k-1}, s_k, y_k)$ ,  $k = 1, \dots, n$ , geschätzt werden. Die internen Zustände  $s$  sind in der Regel unbekannt und müssen anhand der beobachteten Daten geschätzt werden. Jeder Zustand  $s_k$  hängt ab von den (ebenso unbekannten) vorherigen Zuständen  $s_0, \dots, s_{k-1}$ . Die vorherigen Zustände wurden über die Funktion  $f_s$  durch die Eingangswerte  $x_1, \dots, x_{k-1}$  beeinflusst und haben ihrerseits über die Funktion  $f_y$  die Ausgangswerte  $y_1, \dots, y_{k-1}$  beeinflusst. Wir können daher annehmen, dass  $s_k$  aus  $x_1, \dots, x_{k-1}$  und  $y_1, \dots, y_{k-1}$  geschätzt werden kann. Diese Annahme führt zu einem zustandsfreien Modell, dass als Approximation einer Moore-Maschine interpretiert werden kann:

$$y_k = f_k(y_1, \dots, y_{k-1}, x_1, \dots, x_{k-1}), \quad k = 2, \dots, n \quad (7.6)$$

Die Funktionen  $f_2, \dots, f_n$  können allerdings nicht durch Regression bestimmt werden, da in jedem Zeitschritt  $k$  nur ein einziges vollständiges Datentupel vorliegt. Dieses Problem kann behoben werden, indem der Zustand nicht aus allen vorangegangenen Ein- und Ausgangswerten bestimmt wird, sondern nur aus den letzten  $m$ . Das entsprechende Modell lautet

$$y_k = f(y_{k-m}, \dots, y_{k-1}, x_{k-m}, \dots, x_{k-1}), \quad k = m + 1, \dots, n \quad (7.7)$$

Die Funktion  $f$  kann aus  $n - m$  Datentupeln bestimmt werden. Für  $m = 3$  und  $n = 8$  lauten diese  $8 - 3 = 5$  Tupel beispielsweise

$$y_4 = f(y_1, y_2, y_3, x_1, x_2, x_3) \quad (7.8)$$

$$y_5 = f(y_2, y_3, y_4, x_2, x_3, x_4) \quad (7.9)$$

$$y_6 = f(y_3, y_4, y_5, x_3, x_4, x_5) \quad (7.10)$$

$$y_7 = f(y_4, y_5, y_6, x_4, x_5, x_6) \quad (7.11)$$

$$y_8 = f(y_5, y_6, y_7, x_5, x_6, x_7) \quad (7.12)$$

Die Werte  $x_m, \dots, x_{n-m}$  und  $y_{m+1}, \dots, y_{n-m}$  ( $x_3, x_4, x_5, y_4, y_5$  in obigem Beispiel) treten am häufigsten auf und haben daher den größten Einfluss auf das Prognosemodell.

1. Eingabe: Eingangsdaten  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$ ,  
Ausgangsdaten  $Y = \{y_1, \dots, y_n\} \subset \mathbb{R}^q$ ,  
Parameter  $m \in \{1, \dots, n\}$
2. Bilde Regressionsdatensatz  $Z$  gemäß (7.7)
3. Schätze  $f$  aus  $Z$  mit einer geeigneten Regressionsmethode
4. Ausgabe:  $f$

**Abb. 7.2** Zustandsfreie Prognose

Die Anzahl der betrachteten Zeitschritte  $m$  sollte nicht zu klein sein, um die internen Zustände gut identifizieren zu können, und nicht zu groß, damit die Anzahl der Datentupel groß genug ist für ein gutes Regressionsmodell. Abbildung 7.2 zeigt den Algorithmus zur Bestimmung eines zustandsfreien Prognosemodells. Dieser Algorithmus führt die Prognose auf ein Regressionsproblem zurück. Hierfür lässt sich jedes beliebige Regressionsmodell verwenden, z. B. die in Kap. 6 beschriebenen Modelle. Geeignete Werte für  $m$  und die Parameter der gewählten Regressionsmodelle können durch Kreuzvalidierung bestimmt werden (siehe Kap. 6). Das Prognosemodell liefert die modellbasierten Schätzungen für die (bereits bekannten) Ausgangswerte  $y_1, \dots, y_n$  und kann benutzt werden, um die zukünftigen Ausgangswerte  $y_{n+1}, y_{n+2}, \dots$  vorherzusagen, falls die zukünftigen Eingangswerte  $x_{n+1}, x_{n+2}, \dots$  bekannt sind oder zumindest geschätzt werden können. Eine einfache Abschätzung liefert der Mittelwert der bisherigen Eingangswerte.

$$x_k = \frac{1}{n} \sum_{j=1}^n x_j, \quad k = n + 1, n + 2, \dots \quad (7.13)$$

### 7.3 Autoregressive Modelle

Das Problem der unbekannten zukünftigen Eingangswerte kann vermieden werden, wenn die Eingangswerte in der Prognose überhaupt nicht betrachtet werden. Dies führt zu den sogenannten *autoregressiven Modellen*

$$y_k = f(y_{k-m}, \dots, y_{k-1}), \quad k = m + 1, \dots, n \quad (7.14)$$

Für  $m = 3$  und  $n = 8$  lauten die entsprechenden Regressionstupel beispielsweise

$$y_4 = f(y_1, y_2, y_3) \quad (7.15)$$

$$y_5 = f(y_2, y_3, y_4) \quad (7.16)$$

$$y_6 = f(y_3, y_4, y_5) \quad (7.17)$$

$$y_7 = f(y_4, y_5, y_6) \quad (7.18)$$

$$y_8 = f(y_5, y_6, y_7) \quad (7.19)$$

Bei Verwendung der linearen Regression sprechen wir von *linearen autoregressiven (AR) Modellen* [1]. Solche linearen Modelle können sehr effizient geschätzt werden, sind aber weniger geeignet, wenn die zugrunde liegenden Zusammenhänge stark nichtlinear sind. In diesem Fall können generalisierte AR-Modelle mit gleitendem Mittelwert (ARMA), Integraltermen (ARIMA) oder mit lokaler Regression (ARMAX) besser geeignet sein. Für die nichtlineare Prognose sind nichtlineare Regressionsmodelle am besten geeignet, z. B. neuronale Netze. Ein erprobter Ansatz zur Prognose mit rekurrenten neuronalen Netzen basiert auf der (endlich) entfalteten Moore-Maschine in Abb. 7.1 (unten) gemäß (7.3),(7.4), trainiert  $f_s$  und  $f_y$  mit neuronalen Lernverfahren und repräsentiert die unbekannten Zustandsvektoren durch freie Variablen, die während des Trainings geschätzt werden [5].

---

### Übungsaufgaben

- 7.1. Erstellen Sie einen Datensatz für ein autoregressives Prognosemodell mit Zeithorizont  $m = 2$  für die Zeitreihe  $x = \{1, 2, 3, 5, 8\}$ .
  - 7.2. Welches Prognosemodell erhalten Sie mit linearer Regression?
  - 7.3. Welche Vorhersagen liefert dieses Prognosemodell?
- 

## Literatur

1. G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice Hall, 4th edition, 2008.
2. J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, 3rd edition, 2006.
3. G. H. Mealy. A method for synthesizing sequential circuits. *Bell System Technology Journal*, 34:1045–1079, September 1955.
4. E. F. Moore. Gedankenexperiments on sequential machines. In W. R. Ashby, C. E. Shannon, and J. McCarthy, editors, *Automata studies*, pages 129–156. Princeton University Press, 1956.
5. H. G. Zimmermann and R. Neuneier. Modeling dynamical systems by recurrent neural networks. In *International Conference on Data Mining*, pages 557–566, Cambridge, 2000.

## Zusammenfassung

Klassifikation ist ein überwachtes Lernverfahren, das markierte Daten verwendet, um Objekte zu Klassen zuzuordnen. Es werden falsch positive und falsch negative Fehler unterschieden und auf dieser Basis zahlreiche Klassifikationskriterien definiert. Oft werden Paare solcher Kriterien zur Bewertung von Klassifikatoren verwendet und z. B. in einem ROC- (engl. *Receiver Operating Curve*) oder PR-Diagramm (engl. *Precision Recall*) dargestellt. Unterschiedliche Klassifikatoren mit spezifischen Vor- und Nachteilen werden vorgestellt: der naive Bayes-Klassifikator, lineare Diskriminanzanalyse, die Supportvektormaschine auf Basis des Kernel-Tricks, nächste-Nachbarn-Klassifikatoren, lernende Vektorquantifizierung und hierarchische Klassifikation mit Regressionsbäumen.

## 8.1 Klassifikationskriterien

In den vorangegangenen Kapiteln haben wir Merkmalsdaten

$$X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p \quad (8.1)$$

und Paare von Eingangs- und Ausgangsdaten betrachtet

$$Z = \{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathbb{R}^{p+q} \quad (8.2)$$

In diesem Kapitel betrachten wir Merkmalsdaten von Objekten, die  $c$  Klassen zugeordnet werden,  $c \in \{2, 3, \dots\}$ . Beispielsweise liefert eine medizinische Untersuchung Merkmalswerte wie Körpertemperatur und Blutdruck, mit denen festgestellt werden kann, ob ein Patient krank oder gesund ist, d. h. ob er der Klasse der kranken oder gesunden Patienten

zuzuordnen ist. Die Klassenzuordnung eines Merkmalsdatensatzes kann durch einen Klassenvektor  $y \subset \{1, \dots, c\}$  spezifiziert werden. Merkmalsdaten und Klassenzuordnungen bilden einen *markierten* Merkmalsdatensatz

$$Z = \{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathbb{R}^p \times \{1, \dots, c\} \quad (8.3)$$

Bei der Klassifikation bezeichnen wir also mit  $y_k \in \{1, \dots, c\}$  die Klassenzuordnung, und bei der Regression bezeichnen wir mit  $y_k \in \mathbb{R}^q$  die Ausgangsdaten.

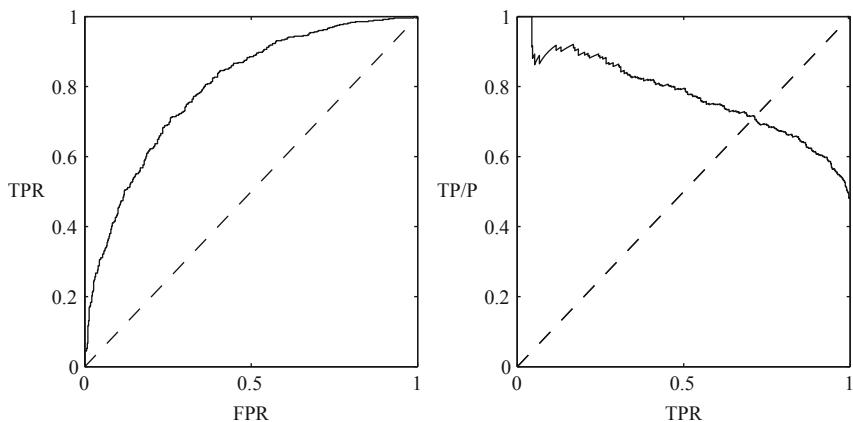
Wir nehmen an, dass die Merkmalsvektoren und die Klassen der dadurch beschriebenen Objekte in einem systematischen Zusammenhang stehen, und dass die Objekte für die jeweiligen Klassen repräsentativ sind. Unter dieser Annahme können die Merkmalsvektoren und Klassenzuordnungen zur Bestimmung von generalisierenden Klassenmodellen verwendet werden. Neue Objekte mit bisher unbekannter Klassenzuordnung können klassifiziert werden, indem deren Merkmalsvektoren mit den einzelnen Klassenmodellen verglichen werden. Wenn die Daten beispielsweise darauf hindeuten, dass Patienten mit Fieber zur Klasse „krank“ gehören, so werden alle neuen Patienten mit Fieber als krank klassifiziert. Wir nutzen also einen markierten Datensatz (d. h. einen Datensatz mit Merkmalsdaten und Klassenzuordnungen), um eine Klassifikationsfunktion  $f: \mathbb{R}^p \rightarrow \{1, \dots, c\}$  zu bestimmen, die für einen gegebenen Merkmalsvektor  $x$  die Klasse  $y = f(x)$  liefert. Die Bestimmung einer solchen Klassifikationsfunktion aus Merkmalsvektoren und Klassenzuordnungen heißt *Klassifikator-Entwurf*, und die Anwendung dieser Funktion auf Merkmalsvektoren heißt *Klassifikation* [9, 21]. Der Klassifikator-Entwurf hat große Ähnlichkeit mit der Bestimmung von Regressionsmodellen (Kap. 6). Die verfügbaren markierten Daten werden in Trainings- und Validierungsdaten aufgeteilt, der Klassifikator wird mit den Trainingsdaten bestimmt und dann mit den Validierungsdaten validiert. Ziel ist eine hohe Klassifikationsgüte auf den Validierungsdaten.

Zur Bestimmung der Klassifikationsgüte betrachten wir eine bestimmte Klasse, z. B. die Klasse der kranken Patienten. Ein idealer Klassifikator klassifiziert alle gesunden Patienten als gesund und alle kranken Patienten als krank. Eine Fehlklassifikation liegt vor, wenn ein gesunder Patient als krank klassifiziert wird oder ein kranker Patient als gesund. Diese beiden Varianten von Fehlklassifikationen haben sehr unterschiedliche Bedeutung und Auswirkungen. Sie werden daher bei der Bestimmung der Klassifikationsgüte unterschieden. Für ein Klassifikationsergebnis sind folgende vier Fälle möglich [2, 15]:

1. *richtig positiv* (engl. *true positive*, *TP*):  $y = i, f(x) = i$   
(ein kranker Patient wird als krank klassifiziert)
2. *richtig negativ* (engl. *true negative*, *TN*):  $y \neq i, f(x) \neq i$   
(ein gesunder Patient wird als gesund klassifiziert)
3. *falsch positiv* (engl. *false positive*, *FP*):  $y \neq i, f(x) = i$   
(ein gesunder Patient wird als krank klassifiziert)
4. *falsch negativ* (engl. *false negative*, *FN*):  $y = i, f(x) \neq i$   
(ein kranker Patient wird als gesund klassifiziert)

Falsch positiv wird auch Fehler erster Ordnung genannt, und falsch negativ Fehler zweiter Ordnung. Auf Basis der Häufigkeiten der vier Fehler TP, TN, FP und FN werden die folgenden häufig verwendeten Kriterien definiert:

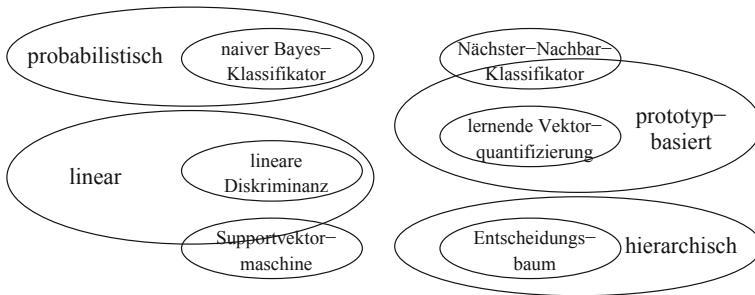
- Anzahl der Klassifikationen  $n = TP + TN + FP + FN$
- *richtige Klassifikationen* (engl. *True Classifications*)  $T = TP + TN$   
(Anzahl der richtig klassifizierten Patienten)
- *falsche Klassifikationen* (engl. *False Classifications*)  $F = FP + FN$   
(Anzahl der falsch klassifizierten Patienten)
- *relevante Klassifikationen*  $R = TP + FN$   
(Anzahl der kranken Patienten)
- *irrelevante Klassifikationen*  $I = FP + TN$   
(Anzahl der gesunden Patienten)
- *positive Klassifikationen*  $P = TP + FP$   
(Anzahl der als krank klassifizierten Patienten)
- *negative Klassifikationen*  $N = TN + FN$   
(Anzahl der als gesund klassifizierten Patienten)
- *Korrektklassifikationsrate* oder *Vertrauenswahrscheinlichkeit*  $T/n$   
(Wahrscheinlichkeit, dass ein Patient korrekt klassifiziert wird)
- *Falschklassifikationsrate*  $F/n$   
(Wahrscheinlichkeit, dass ein Patient falsch klassifiziert wird)
- *Richtig-Positiv-Rate* (engl. *True Positive Rate*) oder *Sensitivität* oder *Empfindlichkeit* oder *Trefferquote* (engl. *Recall*)  $TPR = TP/R$   
(Wahrscheinlichkeit, dass ein kranker Patient als krank klassifiziert wird)
- *Richtig-Negativ-Rate* (engl. *True Negative Rate*) oder *Spezifität*  $TNR = TN/I$   
(Wahrscheinlichkeit, dass ein gesunder Patient als gesund klassifiziert wird)
- *Falsch-Positiv-Rate* (engl. *False Positive Rate*) oder *Ausfallrate*  $FPR = FP/I$   
(Wahrscheinlichkeit, dass ein gesunder Patient als krank klassifiziert wird)
- *Falsch-Negativ-Rate* (engl. *False Negative Rate*)  $FNR = FN/R$   
(Wahrscheinlichkeit, dass ein kranker Patient als gesund klassifiziert wird)
- *positiver Vorhersagewert* oder *positiver prädiktiver Wert* oder *Relevanz* oder *Wirksamkeit* oder *Genauigkeit* (engl. *Precision*)  $TP/P$   
(Wahrscheinlichkeit, dass ein als krank klassifizierter Patient krank ist)
- *negativer Vorhersagewert* oder *negativer prädiktiver Wert* oder *Segreganz* oder *Trennfähigkeit*  $TN/N$   
(Wahrscheinlichkeit, dass ein als gesund klassifizierter Patient gesund ist)
- *negative Falschklassifikationsrate*  $FN/N$   
(Wahrscheinlichkeit, dass ein als gesund klassifizierter Patient krank ist)
- *positive Falschklassifikationsrate*  $FP/P$   
(Wahrscheinlichkeit, dass ein als krank klassifizierter Patient gesund ist)
- *F-Maß*  $F = 2 \cdot TP / (R+P)$   
(harmonisches Mittel aus positivem Vorhersagewert und Richtig-Positiv-Rate)



**Abb. 8.1** ROC- und PR-Diagramm

Um einen Klassifikator zu bewerten, reicht es nicht aus, nur eines dieser Kriterien zu betrachten. Für einzelne Kriterien lassen sich mit trivialen Klassifikatoren gute Werte erzielen. Z. B. lässt sich eine Richtig-Positiv-Rate von 100 % (sehr gut) mit einem trivialen Klassifikator erreichen, der für jeden Merkmalsvektor stets die Klassifikation positiv liefert, dieser Klassifikator führt jedoch auch zu einer Falsch-Positiv-Rate von 100 % (sehr schlecht). Bei der Bewertung von Klassifikatoren werden daher meist zwei oder mehr Klassifikationskriterien betrachtet.

Ein Beispiel für diese Herangehensweise ist die *Grenzwertoptimierungskurve* (engl. *Receiver Operating Characteristic*, kurz *ROC-Diagramm*), ein Streudiagramm von Richtig-Positiv-Rate TPR und Falsch-Positiv-Rate FPR. Abbildung 8.1 (links) zeigt ein Beispiel für ein ROC-Diagramm eines Klassifikators mit unterschiedlichen Parameterwerten. Die (Trainings- oder Validierungs-)Güte eines bestimmten Klassifikators mit bestimmten Parameterwerten auf einem bestimmten Datensatz lässt sich als Punkt im ROC-Diagramm darstellen. Durch Variation der Parameter entsteht eine ROC-Kurve. Das ROC-Diagramm lässt sich für verschiedene Analysen nutzen, z. B. um Trainings- und Validierungsgüte eines Klassifikators zu vergleichen, um verschiedene Klassifikatoren zu bewerten, um optimale Werte für Klassifikationsparameter zu finden oder um die Güte eines Klassifikators für verschiedene Datensätze zu analysieren. Ein idealer Klassifikator zeichnet sich durch 100 % TPR und 0 % FPR aus und entspricht somit der oberen linken Ecke im ROC-Diagramm. Ein guter Klassifikator sollte daher möglichst nah an der oberen linken Ecke liegen. Ein Klassifikator, der immer das Ergebnis positiv liefert (siehe oben), entspricht der oberen rechten Ecke (100 % TPR, 100 % FPR). Ein Klassifikator, der immer das Ergebnis negativ liefert, entspricht der unteren linken Ecke (0 % TPR, 0 % FPR). Ein Klassifikator, der immer das falsche Ergebnis liefert, entspricht der unteren rechten Ecke (0 % TPR, 100 % FPR). Ein solcher Klassifikator lässt sich durch Invertierung in einen idealen

**Abb. 8.2** Klassifikationsverfahren

Klassifikator überführen: Aus positiv wird negativ und umgekehrt. Die Invertierung eines beliebigen Klassifikators entspricht im ROC-Diagramm einer Spiegelung am Mittelpunkt (50 % TPR, 50 % FPR). Somit lässt sich durch Invertierung jeder Klassifikator unterhalb der gestrichelten Hauptdiagonalen in einen (besseren) Klassifikator oberhalb der Hauptdiagonalen überführen. In ROC-Diagrammen werden daher meist nur Punkte oberhalb der Hauptdiagonalen dargestellt.

Ein zweites Beispiel für die Bewertung von Klassifikatoren mit zwei Klassifikationskriterien ist das *Genauigkeit-Trefferquote-Diagramm* (engl. *Precision Recall* (PR) Diagramm), ein Streudiagramm von Genauigkeit (positivem Vorhersagewert) TP/P und Trefferquote (Richtig-Positiv-Rate) FPR. Abbildung 8.1 (rechts) zeigt ein Beispiel für ein Genauigkeit-Trefferquote-Diagramm. Für eine niedrige Trefferquote lässt sich relativ leicht eine hohe Genauigkeit erzielen, d. h. es lässt sich leicht ein Klassifikator erstellen, der wenige kranke Patienten korrekt klassifiziert. Bei einer höheren Trefferquote sinkt in der Regel die Genauigkeit. Ein guter Klassifikator behält eine hohe Genauigkeit auch für eine steigende Trefferquote. Der Schnittpunkt der PR-Kurve mit der gestrichelten Hauptdiagonalen wird *Genauigkeit-Trefferquote-Grenzwert* (engl. *Precision Recall Breakeven Point*) genannt. Bei einem guten Klassifikator sollte dieser Grenzwert möglichst hoch sein. Eine detailliertere Abhandlung und ein Vergleich von ROC- und PR-Diagrammen findet sich in [8].

In den folgenden Abschnitten werden einige wichtige Familien von Klassifikatoren mit ihren spezifischen Vor- und Nachteilen vorgestellt: probabilistische, lineare, prototypbasierte und hierarchische Klassifikatoren (Abb. 8.2).

## 8.2 Naiver Bayes Klassifikator

Ein *naiver Bayes-Klassifikator* ist ein probabilistisches Klassifikationsverfahren, das auf dem Satz von Bayes [3] basiert: Für zwei Zufallereignisse  $A$  und  $B$  gilt

$$p(A \mid B) \cdot p(B) = p(B \mid A) \cdot p(A) \quad (8.4)$$

Lässt sich das Ereignis  $A$  in die disjunkten Ereignisse  $A_1, \dots, A_c$  zerlegen,  $p(A_i) > 0$ ,  $i = 1, \dots, c$ , so gilt weiter

$$p(A_i | B) = \frac{p(A_i) \cdot p(B | A_i)}{\sum_{j=1}^c p(A_j) \cdot p(B | A_j)} \quad (8.5)$$

Die in der Klassifikation betrachteten Ereignisse lauten „Objekt gehört zur Klasse  $i$ “ (kurz  $i$ ) und „Objekt hat den Merkmalsvektor  $x$ “ (kurz  $x$ ). Durch Einsetzen dieser Ereignisse als  $A_i$ ,  $A_j$  und  $B$  in (8.5) erhalten wir

$$p(i | x) = \frac{p(i) \cdot p(x | i)}{\sum_{j=1}^c p(j) \cdot p(x | j)} \quad (8.6)$$

Falls die  $p$  Merkmale in  $x$  stochastisch unabhängig sind, so gilt

$$p(x | i) = \prod_{k=1}^p p(x^{(k)} | i) \quad (8.7)$$

Einsetzen in (8.6) liefert die Klassifikationswahrscheinlichkeiten des naiven Bayes-Klassifikators.

$$p(i | x) = \frac{p(i) \cdot \prod_{k=1}^p p(x^{(k)} | i)}{\sum_{j=1}^c p(j) \cdot \prod_{k=1}^p p(x^{(k)} | j)} \quad (8.8)$$

Für einen gegebenen klassifizierten Merkmalsdatensatz  $Z = \{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathbb{R}^p \times \{1, \dots, c\}$  können die benötigten Wahrscheinlichkeiten durch die beobachteten relativen Häufigkeiten geschätzt werden: Die Wahrscheinlichkeit  $p(i)$  ist die relative Häufigkeit von Klasse  $y = i$ ,  $i = 1, \dots, c$ , und jede Wahrscheinlichkeit  $p(x^{(k)} | i)$  ergibt sich als relative Häufigkeit von Merkmal  $x^{(k)}$  in Klasse  $y = i$ ,  $i = 1, \dots, c$ ,  $k = 1, \dots, p$ . Für einen gegebenen Merkmalsvektor  $x$  liefert Gl. (8.8) eine Wahrscheinlichkeit für jede Klasse. Für eine deterministische Klassifikation wird die Klasse mit der höchsten Wahrscheinlichkeit gewählt.

Als Beispiel für die naive Bayes-Klassifikation betrachten wir eine Gruppe von Studierenden, die an einer Prüfung teilnehmen. Einige der Teilnehmer waren regelmäßig in der Vorlesung, andere nicht. Einige haben das Skript durchgearbeitet, andere nicht. Einige haben die Prüfung bestanden, andere nicht. Tabelle 8.1 zeigt die entsprechenden Anzahlen. Gesucht ist die Wahrscheinlichkeit, dass ein Teilnehmer die Prüfung besteht, der regelmäßig in der Vorlesung war und das Skript durchgearbeitet hat. Aus den Daten in Tab. 8.1 berechnen wir die Wahrscheinlichkeiten

$$p(\text{regelmäßig in der Vorlesung} | \text{bestanden}) = \frac{21}{21+1} = \frac{21}{22} \quad (8.9)$$

**Tab. 8.1** Naiver Bayes-Klassifikator: Daten des Prüfungsbeispiels

	Bestanden	Nicht bestanden
Regelmäßig in der Vorlesung	21	4
Nicht regelmäßig in der Vorlesung	1	3
Skript durchgearbeitet	16	2
Skript nicht durchgearbeitet	6	5

$$p(\text{Skript durchgearbeitet} \mid \text{bestanden}) = \frac{16}{16+6} = \frac{16}{22} \quad (8.10)$$

$$\Rightarrow p(x \mid \text{bestanden}) = \frac{21 \cdot 16}{22 \cdot 22} = \frac{84}{121} \quad (8.11)$$

$$p(\text{regelmäßig in der Vorlesung} \mid \text{nicht bestanden}) = \frac{4}{4+3} = \frac{4}{7} \quad (8.12)$$

$$p(\text{Skript durchgearbeitet} \mid \text{nicht bestanden}) = \frac{2}{2+5} = \frac{2}{7} \quad (8.13)$$

$$\Rightarrow p(x \mid \text{nicht bestanden}) = \frac{4 \cdot 2}{7 \cdot 7} = \frac{8}{49} \quad (8.14)$$

$$p(\text{bestanden}) = \frac{22}{22+7} = \frac{22}{29} \quad (8.15)$$

$$p(\text{nicht bestanden}) = \frac{7}{22+7} = \frac{7}{29} \quad (8.16)$$

$$p(\text{bestanden}) \cdot p(x \mid \text{bestanden}) = \frac{22}{29} \cdot \frac{84}{121} = \frac{168}{319} \quad (8.17)$$

$$p(\text{nicht bestanden}) \cdot p(x \mid \text{nicht bestanden}) = \frac{7}{29} \cdot \frac{8}{49} = \frac{8}{203} \quad (8.18)$$

$$\Rightarrow p(\text{bestanden} \mid x) = \frac{\frac{168}{319}}{\frac{168}{319} + \frac{8}{203}} \quad (8.19)$$

$$= \frac{168 \cdot 203}{168 \cdot 203 + 8 \cdot 319} = \frac{147}{158} \approx 93\% \quad (8.20)$$

Ein Teilnehmer, der regelmäßig in der Vorlesung war und das Skript durchgearbeitet hat, wird die Prüfung also mit einer Wahrscheinlichkeit von 93 % bestehen, und ein deterministischer naiver Bayes-Klassifikator wird die Klassifikation „bestanden“ liefern. Tabelle 8.2 zeigt die Klassifikationswahrscheinlichkeiten für alle vier möglichen Teilnehmerprofile. Zu den Vorteilen des naiven Bayes-Klassifikators gehört seine Effizienz, da die Trainingsdaten nur ein einziges Mal durchlaufen werden müssen (zur Bestimmung der relativen Häufigkeiten), und dass fehlende Daten einfach ignoriert werden können. Zu den Nachteilen gehört, dass die Merkmale stochastisch unabhängig sein müssen, was in praktischen Anwendungen häufig nicht der Fall ist, und dass die Merkmale diskret sein müssen. Um kontinuierliche Merkmale zu verwenden, können diese diskretisiert werden, wie in den Kap. 4 (Histogramme) und 5 (Chi-Quadrat-Test) beschrieben.

**Tab. 8.2** Naiver Bayes-Klassifikator: Klassifikationsfunktion für das Prüfungsbeispiel

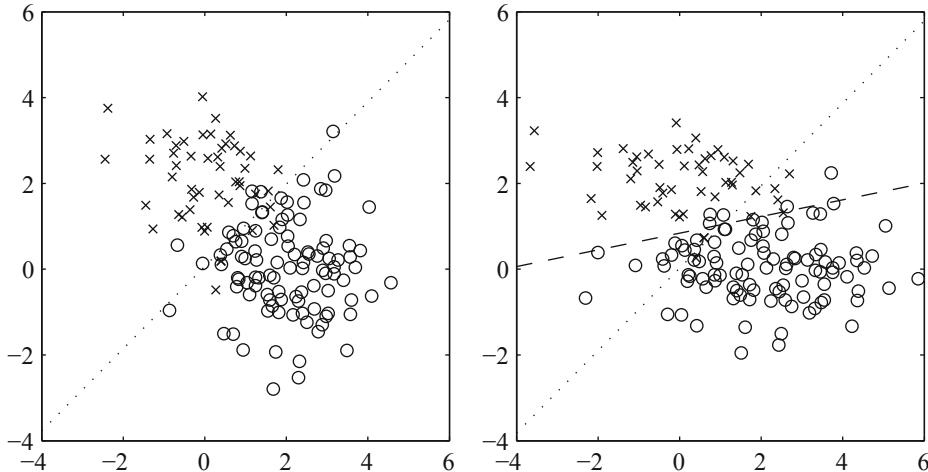
Regelmäßig in der Vorlesung	Skript durchgearbeitet	Wahrscheinlichkeit für bestanden (%)	Wahrscheinlichkeit für nicht bestanden (%)	Klassifikation bestanden
Ja	Ja	93	7	Ja
Ja	Nein	67	33	Ja
Nein	Ja	46	54	Nein
Nein	Nein	11	89	Nein

### 8.3 Lineare Diskriminanzanalyse

Abbildung 8.3 zeigt ein Streudiagramm eines zweidimensionalen reellwertigen Datensatzes mit zwei Klassen. Die Merkmalsvektoren von Klasse 1 sind als Kreuze gekennzeichnet und die Merkmalsvektoren von Klasse 2 als Kreise. Die meisten Punkte oberhalb der gepunkteten Geraden gehören zu Klasse 1 (Kreuze), und die meisten Punkte unterhalb dieser Geraden gehören zu Klasse 2 (Kreise). Die gepunktete Gerade kann also als Grenze zwischen den beiden Klassen betrachtet werden, eine sogenannte *Diskriminanzgerade*, in Normalform geschrieben als

$$w \cdot x^T + b = 0, \quad w \in \mathbb{R}^p, \quad b \in \mathbb{R} \quad (8.21)$$

Für höhere Dimensionen wird damit eine Diskriminanzebene ( $p = 3$ ) oder Diskriminanzhyperebene ( $p > 3$ ) beschrieben. Ohne Beschränkung der Allgemeinheit betrachten wir

**Abb. 8.3** Lineare Diskriminanz bezüglich der Klassenmittelwerte (links, gepunktet) und lineare Diskriminanzanalyse (rechts, gestrichelt)

hier nur den zweidimensionalen Fall ( $p = 2$ ). Für einen gegebenen Datensatz  $Z$  bestimmt die lineare Diskriminanzanalyse die Parameter  $w$  und  $b$  einer Diskriminanzgerade (bzw. -ebene oder -hyperebene), so dass ein gegebenes Kriterium optimiert wird. Der Einfachheit halber beschränken wir uns in diesem Abschnitt auf nur zwei Klassen. Eine Erweiterung auf mehrere Klassen kann durch Verwendung mehrerer Diskriminanzgeraden erfolgen oder durch Kombination mehrerer zweiklassiger Klassifikatoren.

Die Daten in Abb. 8.3 (links) sind näherungsweise Gauß-verteilt mit den Mittelwerten  $\mu_1$ ,  $\mu_2$  und näherungsweise gleichen Standardabweichungen  $\sigma_1 \approx \sigma_2$ . Aus Symmetriegründen verläuft die Diskriminanzgerade durch die Mitte der beiden Zentren, senkrecht zur Verbindungsgeraden, mit den Parametern

$$w = \mu_1 - \mu_2 \quad (8.22)$$

$$b = -w \cdot \frac{\mu_1^T + \mu_2^T}{2} \quad (8.23)$$

Die gepunktete Gerade in Abb. 8.3 wurde auf diese Weise berechnet.

Abbildung 8.3 (rechts) zeigt zwei Klassen mit den gleichen Zentren  $\mu_1$  und  $\mu_2$  wie links, jedoch sind die horizontalen Standardabweichungen größer als die vertikalen. Die gepunktete Gerade gemäß (8.22) und (8.23) führt nicht zu einer geeigneten Separierung der beiden Klassen. Eine bessere Separierung liefert die im Folgenden beschriebene *lineare Diskriminanzanalyse*.

Durch Projektion der Daten auf eine Senkrechte zur gepunkteten Geraden erhalten wir für die Daten in Abb. 8.3 (links) eine gute Separierung und für die Daten in Abb. 8.3 (rechts) eine schlechte Separierung. Gute Separierbarkeit bedeutet, dass in der Projektion die Kovarianzen innerhalb der Klassen

$$v_w = \sum_{i=1}^c \sum_{y_k=i} (x_k - \mu_i)^T (x_k - \mu_i) \quad (8.24)$$

möglichst gering sind und die Kovarianzen zwischen den Klassen

$$v_b = \sum_{i=1}^c (\mu_i - \bar{x})^T (\mu_i - \bar{x}) \quad (8.25)$$

möglichst hoch. Die lineare Diskriminanzanalyse nach Fisher [10] (die ursprünglich auch zur Analyse der Iris-Daten verwendet wurde, siehe Kap. 2) maximiert daher den Quotienten

$$J = \frac{w^T \cdot v_b \cdot w}{w^T \cdot v_w \cdot w} \quad (8.26)$$

Ähnlich wie bei der Hauptkomponentenanalyse (Kap. 4) führt die Maximierung dieses Quotienten zu dem Eigenwertproblem

$$(v_b^{-1} v_w) \cdot w = \lambda \cdot w \quad (8.27)$$

mit dem sich  $w$  und daraus  $b$  (8.23) bestimmen lässt. Für  $c = 2$  lässt sich (8.25) vereinfachen zu

$$v_b = (\mu_1 - \mu_2)^T(\mu_1 - \mu_2) \quad (8.28)$$

mit den Lösungen

$$w = v_w^{-1}(\mu_1 - \mu_2) \quad (8.29)$$

und (8.23). Die gestrichelte Gerade in Abb. 8.3 (rechts) wurde mit (8.29) und (8.23) berechnet und separiert die Klassen deutlich besser als die gepunktete Gerade.

Zu den Vorteilen der linearen Diskriminanzanalyse gehören ihre Effizienz und ihre Eignung auch für korrelierte Merkmale. Zu den Nachteilen gehört, dass sie eine näherungsweise Gauß-Verteilung der Merkmale voraussetzt und nur für lineare Klassengrenzen geeignet ist.

## 8.4 Supportvektormaschine

Die *Supportvektormaschine (SVM)* [19] verwendet ebenfalls lineare Klassengrenzen, es wird jedoch gefordert, dass die Daten einen Mindestabstand  $b > 0$  von der Klassengrenze einhalten. Für zwei Klassen wird also gefordert, dass

$$w \cdot x_k^T + b \geq +1 \quad \text{falls } y_k = 1 \quad (8.30)$$

$$w \cdot x_k^T + b \leq -1 \quad \text{falls } y_k = 2 \quad (8.31)$$

Falls für diese Randbedingungen mehrere (möglicherweise unendlich viele) Lösungen existieren, sucht die SVM die Lösung mit minimalem

$$J = \|w\|^2 \quad (8.32)$$

was einer Maximierung des Abstands  $b$  entspricht. Falls die Klassen nahe beieinander liegen oder gar überlappen, kann möglicherweise keine Klassengrenze gefunden werden, die (8.30) und (8.31) erfüllt. Die Bedingungen werden also relaxiert zu

$$w \cdot x_k^T + b \geq +1 - \xi_k \quad \text{falls } y_k = 1 \quad (8.33)$$

$$w \cdot x_k^T + b \leq -1 + \xi_k \quad \text{falls } y_k = 2 \quad (8.34)$$

mit den Schlupfvariablen  $\xi_1, \dots, \xi_n > 0$ . Um die Werte der Schlupfvariablen gering zu halten, wird ein Strafterm zur Kostenfunktion (8.32) addiert, so dass

$$J = \|w\|^2 + \gamma \cdot \sum_{k=1}^n \xi_k, \quad \gamma > 0 \quad (8.35)$$

Die SVM-Parameter  $w$  und  $b$  werden bestimmt durch Minimierung von (8.35) unter den Randbedingungen (8.33) und (8.34). Dieses Optimierungsproblem unter Randbedingungen wird oft mit quadratischer Programmierung gelöst (siehe Anhang). Der Normalvektor  $w$  lässt sich als Linearkombination der Trainingsdaten darstellen

$$w = \sum_{y_j=1} \alpha_j x_j - \sum_{y_j=2} \alpha_j x_j \quad (8.36)$$

wobei statt des Normalvektors  $w$  dann die optimalen Gewichte  $\alpha_1, \dots, \alpha_n$  gefunden werden müssen. Die entsprechende Klassifikationsregel lautet

$$\sum_{y_j=1} \alpha_j x_j x_k^T - \sum_{y_j=2} \alpha_j x_j x_k^T + b \geq +1 - \xi_k \quad \text{falls } y_k = 1 \quad (8.37)$$

$$\sum_{y_j=1} \alpha_j x_j x_k^T - \sum_{y_j=2} \alpha_j x_j x_k^T + b \leq -1 + \xi_k \quad \text{falls } y_k = 2 \quad (8.38)$$

Hierdurch wird die Anzahl der freien Parameter von  $p + 1$  auf  $n + 1$  erhöht, wobei oft  $n \gg p$ , so dass die Optimierung deutlich schwieriger wird, aber die Anwendung des sogenannten *Kernel-Tricks* ermöglicht, mit dem SVM auf nichtlineare Klassengrenzen erweitert werden kann. Die Idee des Kernel-Tricks ist es, einen Datensatz  $X = \{x_1, \dots, x_n\} \in \mathbb{R}^p$  auf einen höherdimensionalen Datensatz  $X' = \{x'_1, \dots, x'_n\} \in \mathbb{R}^q$ ,  $q > p$ , abzubilden, so dass die Struktur der Daten in  $X'$  für die gegebene Aufgabe besser geeignet ist als die Struktur in  $X$ . Bei der Supportvektormaschine nutzen wir den Kernel-Trick, um Daten mit nichtlinearen Klassengrenzen in  $X$  auf Daten mit näherungsweise linearen Klassengrenzen in  $X'$  abzubilden und dann das oben beschriebene Verfahren für lineare Klassengrenzen auf  $X'$  anzuwenden. In Kap. 9 wird gezeigt, wie sich der Kernel-Trick auch in der relationalen Clusteranalyse einsetzen lässt. Die mathematische Grundlage des Kernel-Tricks ist der *Satz von Mercer* [14], der bereits vor über hundert Jahren veröffentlicht wurde, aber erst in den letzten Jahren durch seine Anwendung als Kernel-Trick populär geworden ist. Der Satz von Mercer besagt, dass es für jeden Datensatz  $X$  und jede Kernelfunktion  $k: \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}$  eine Abbildung  $\varphi: \mathbb{R}^p \rightarrow \mathbb{R}^q$  gibt so, dass

$$k(x_j, x_k) = \varphi(x_j) \cdot \varphi(x_k)^T \quad (8.39)$$

Das bedeutet, dass eine Abbildung von  $X$  auf  $X'$  implizit dadurch erfolgen kann, dass Skalarprodukte in  $X'$  durch Kernelfunktionen in  $X$  ersetzt werden, ohne dass  $X'$  explizit berechnet werden muss. Das Ersetzen von Skalarprodukten in  $X'$  durch Kernelfunktionen in  $X$  wird als Kernel-Trick bezeichnet. Häufig verwendete Kernel-Funktionen sind

- linearer Kernel

$$k(x_j, x_k) = x_j \cdot x_k^T \quad (8.40)$$

- polynomieller Kernel

$$k(x_j, x_k) = (x_j \cdot x_k^T)^d, \quad d \in \{2, 3, \dots\} \quad (8.41)$$

- Gauß-Kernel

$$k(x_j, x_k) = e^{-\frac{\|x_j - x_k\|^2}{\sigma^2}}, \quad \sigma > 0 \quad (8.42)$$

- Tangens Hyperbolicus Kernel

$$k(x_j, x_k) = 1 - \tanh \frac{\|x_j - x_k\|^2}{\sigma^2}, \quad \sigma > 0 \quad (8.43)$$

- radialer Basisfunktionskernel [16]

$$k(x_j, x_k) = f(\|x_j - x_k\|) \quad (8.44)$$

Gauß-Kernel und Tangens Hyperbolicus Kernel sind Spezialfälle von radialen Basisfunktionskerneln. Durch Anwendung des Kernel-Tricks auf (8.37) und (8.38) erhalten wir schließlich die SVM-Klassifikationsregel

$$\sum_{y_j=1} \alpha_j k(x_j, x_k) - \sum_{y_j=2} \alpha_j k(x_j, x_k) + b \geq +1 - \xi_k \quad \text{falls } y_k = 1 \quad (8.45)$$

$$\sum_{y_j=1} \alpha_j k(x_j, x_k) - \sum_{y_j=2} \alpha_j k(x_j, x_k) + b \leq -1 + \xi_k \quad \text{falls } y_k = 2 \quad (8.46)$$

Zu den Vorteilen der SVM gehört, dass sie nichtlineare Klassengrenzen finden kann. Zu den Nachteilen gehört der relativ hohe Rechenaufwand, um ein sehr großes Optimierungsproblem unter Nebenbedingungen zu lösen.

## 8.5 Nächster-Nachbar-Klassifikator

Ein viel einfacheres Klassifikationsverfahren ist der *Nächster-Nachbar-Klassifikator* [20], der ein Objekt mit einem gegebenen Merkmalsvektor zur Klasse des Trainingsobjekts mit dem ähnlichsten Merkmalsvektor zuordnet. Für einen gegebenen Merkmalsvektor  $x$  liefert der Nächster-Nachbar-Klassifikator also Klasse  $y_k$ , falls

$$\|x - x_k\| = \min_{j=1, \dots, n} \|x - x_j\| \quad (8.47)$$

wobei  $\|\cdot\|$  ein geeignetes Unähnlichkeitsmaß ist, z. B. der Euklidische oder der Mahalanobis-Abstand. Im Falle mehrerer Minima kann aus diesen eine Zufallsauswahl getroffen werden. In der Nähe von verrauschten Daten oder an Grenzen überlappender Klassen funktioniert der Nächster-Nachbar-Klassifikator schlecht. In diesen Fällen liefert der *Nächste-k-Nachbarn-Klassifikator* oft besserer Ergebnisse. Dieser betrachtet nicht nur den nächsten Nachbarn, sondern die  $k \in \{2, \dots, n\}$  nächsten Nachbarn und liefert deren häufigste Klasse. Für zwei Klassen kann eine eindeutige Klassenzuweisung sichergestellt werden, wenn  $k$  ungerade ist. Die Anzahl  $k$  der nächsten Nachbarn sollte nicht mit dem Objektindex  $k$  verwechselt werden.

Der Nächster-Nachbar- und der Nächste- $k$ -Nachbarn-Klassifikator berechnen in der Trainingsphase keine explizite Klassifikationsvorschrift, sondern führen lediglich eine Speicherung der Daten durch. Die Auswertung der Daten erfolgt erst, wenn ein neuer Merkmalsvektor klassifiziert werden soll. Solche Verfahren werden auch *faules Lernen* (engl. *Lazy Learning*) [1] genannt.

Zu den größten Vorteilen des Nächster-Nachbar- und Nächste- $k$ -Nachbarn-Klassifikators gehört, dass in der Trainingsphase kein Lernaufwand anfällt, dass der Klassifikator leicht um zusätzliche Trainingsdaten ergänzt werden kann, und dass nichtlineare Klassengrenzen gefunden werden können. Zu den größten Nachteilen gehört der hohe Rechenaufwand bei der Klassifikation, da die Unähnlichkeiten mit allen Trainingsvektoren berechnet werden müssen.

---

## 8.6 Lernende Vektorquantisierung

Es wurden zahlreiche Ansätze entwickelt, den Rechenaufwand der Nächster-Nachbar- und Nächste- $k$ -Nachbarn-Klassifikatoren zu verringern. Ein solcher Ansatz extrahiert eine gewisse Anzahl von repräsentativen Merkmalsvektoren (sogenannte *Prototypen*) aus den Trainingsdaten und berechnet dann zur Klassifikation nur die Ähnlichkeiten mit diesen Prototypen anstatt mit allen Trainingsvektoren. Ein Beispiel für diesen Ansatz ist die *lernende Vektorquantisierung (LVQ)* [12, 13], ein heuristisches Verfahren, das genau einen Prototypen  $v_i \in \mathbb{R}^p$ ,  $i = 1, \dots, c$ , für jede Klasse bestimmt und dann einen Merkmalsvektor  $x$  zur Klasse  $i$  zuordnet, falls

$$\|x - v_i\| = \min_{j=1, \dots, c} \|x - v_j\| \quad (8.48)$$

Zum Training der LVQ werden die Prototypen zunächst zufällig initialisiert. Für jeden Trainingsvektor wird dann der ähnlichste Prototyp bewegt, und zwar in Richtung des Trainingsvektors, falls er zur selben Klasse gehört, und in Gegenrichtung, falls er zu einer anderen Klasse gehört. Diese Vorgehensweise ähnelt dem exponentiellen Filter aus Kap. 3. Um eine Konvergenz dieses Verfahrens zu erzwingen, wird die Weglänge der Verschiebungen der Prototypen während des Trainings sukzessive reduziert. Der Algorithmus durchläuft den Trainingsdatensatz mehrere Male, bis eine geeignete Terminierungsbedingung erfüllt ist. Abbildung 8.4 zeigt das LVQ-Trainingsverfahren im Detail. LVQ ordnet jedem Merkmalsvektor genau eine Klasse zu, während die *unscharfe lernende Vektorquantisierung (Fuzzy LVQ, FLVQ)* [4, 7] jeden Merkmalsvektor auch zu gewissen Graden zu mehreren Klassen zuordnet.

Zu den größten Vorteilen von LVQ im Vergleich zum Nächster-Nachbar-Klassifikator gehört, dass bei der Klassifikation nur  $c$  anstatt  $n$  Unähnlichkeitswerte bestimmt werden müssen. Für  $c \ll n$  ist die Klassifikation also deutlich effizienter. Zu den größten Nachteilen gehört, dass eine Trainingsphase benötigt wird, in der jeder Trainingsvektor mehrfach betrachtet werden muss, und dass sich mit einem Prototypen pro Klasse

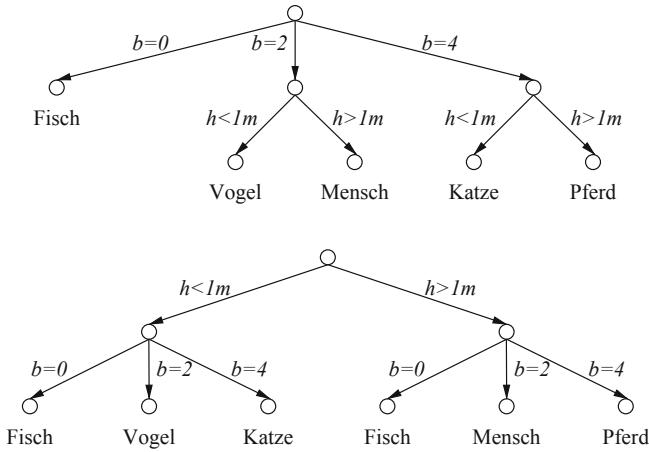
1. Eingabe: klassifizierter Datensatz  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$ ,  $y = \{y_1, \dots, y_n\} \subset \{1, \dots, c\}$ , Klassenanzahl  $c \in \{2, \dots, n-1\}$ , Schrittänge  $\alpha(t)$
2. Initialisiere  $V = \{v_1, \dots, v_c\} \subset \mathbb{R}^p$ ,  $t = 1$
3. Für  $k = 1, \dots, n$ 
  - a. Bestimme Gewinnerprototyp  $v_i \in V$  mit
$$\|x_k - v_i\| \leq \|x_k - v\| \quad \forall v \in V$$
  - b. Verschiebe Gewinnerprototyp
$$v_i = \begin{cases} v_i + \alpha(t)(x_k - v_i) & \text{falls } y_k = i \\ v_i - \alpha(t)(x_k - v_i) & \text{sonst} \end{cases}$$
4.  $t = t + 1$
5. Wiederhole ab (3.), bis Terminierungsbedingung erfüllt
6. Ausgabe: Prototypen  $V = \{v_1, \dots, v_c\} \subset \mathbb{R}^p$

**Abb. 8.4** LVQ-Trainingsalgorithmus

nur vergleichsweise einfache Klassenstrukturen in Form sogenannter *Voronoi-Diagramme* darstellen lassen. Um kompliziertere Klassenstrukturen darzustellen, können pro Klasse mehrere Prototypen verwendet werden. Eine vertiefte Darstellung dieser Ansätze findet sich in [5].

## 8.7 Entscheidungsbäume

Bei allen bisher betrachteten Klassifikatoren werden alle  $p$  Merkmale gleichzeitig betrachtet. Dies kann ungünstig sein, wenn nicht alle Merkmale für eine gute Klassifikation benötigt werden, oder wenn die Erfassung der einzelnen Merkmalsdaten mit großem Aufwand verbunden ist. In der medizinischen Diagnose werden beispielsweise häufig routinemäßig Temperatur- oder Blutdruckmessungen durchgeführt, weil sie mit wenig Aufwand wichtige Informationen über den Gesundheitszustand des Patienten liefern, während aufwändiger Untersuchungen nur bei Bedarf durchgeführt werden. Auf Basis der bisher ermittelten Merkmalsdaten lässt sich der durch zusätzliche Merkmale zu erwartende Informationsgewinn abschätzen, und so lässt sich sukzessive entscheiden, ob und welche weiteren Merkmalswerte berücksichtigt werden. Dies führt zu einer *Hierarchie* der Merkmale, im Gegensatz zu der flachen Merkmalsstruktur der bisher betrachteten Klassifikatoren. Hierarchische Klassifikatoren lassen sich mit sogenannten *Entscheidungsbäumen* darstellen [18]. Abbildung 8.5 zeigt zwei äquivalente Entscheidungsbäume für den (funktional) gleichen Klassifikator. Beide Klassifikatoren erhalten den zweidimensionalen Merkmalsvektor  $x = (b, h)$  eines Lebewesens, wobei  $b \in \{0, 2, 4\}$  die Anzahl der Beine und  $h > 0$  die Körpergröße (in Metern) ist. Auf Basis dieser beiden Merkmalswerte soll ein Lebewesen als Fisch, Vogel, Mensch, Katze oder Pferd klassifiziert werden,



**Abb. 8.5** Zwei (funktional) äquivalente Entscheidungsbäume

es werden hier also fünf Klassen betrachtet. Der erste Entscheidungsbau (Abb. 8.5 oben) betrachtet zuerst das Merkmal  $b$ . Für  $b = 0$  terminiert der Klassifikationsprozess, das Merkmal  $h$  kann ignoriert werden und das Ergebnis ist Fisch. Für  $b = 2$  sind die Klassen Vogel oder Mensch möglich, und für  $b = 4$  die Klassen Katze oder Pferd, das Merkmal  $b$  hat also einen *Informationsgewinn* geliefert, auch wenn der Klassifikationsprozess noch nicht beendet ist und noch das Merkmal  $h$  betrachtet werden muss. Der zweite Entscheidungsbau (Abb. 8.5 unten) betrachtet zuerst das Merkmal  $h$ , das einen Informationsgewinn liefert, und anschließend das Merkmal  $b$ . Beide Entscheidungsbäume realisieren den funktional gleichen Klassifikator, liefern aber bei Betrachtung der einzelnen Merkmale unterschiedliche Informationsgewinne. Für einen Trainingsdatensatz  $Z = (X, Y)$  kann die Wahrscheinlichkeit für Klasse  $k$  berechnet werden als

$$p(y = k) = \frac{|\{y \in Y \mid y = k\}|}{|Y|} \quad (8.49)$$

die *Entropie* am Wurzelknoten beträgt also

$$H(Z) = - \sum_{k=1}^c p(y = k) \log_2 p(y = k) \quad (8.50)$$

$$= - \sum_{k=1}^c \frac{|\{y \in Y \mid y = k\}|}{|Y|} \log_2 \frac{|\{y \in Y \mid y = k\}|}{|Y|} \quad (8.51)$$

Am Wurzelknoten betrachten wir Merkmal  $x^{(j)}$ ,  $j \in \{1, \dots, p\}$ , mit dem diskreten Wertebereich  $x^{(j)} \in \{1, \dots, v_j\}$ ,  $v_j \in \{1, 2, \dots\}$ . Die Wahrscheinlichkeit für  $x^{(j)} = k$ ,  $k = 1, \dots, v_j$ , beträgt

$$p(x^{(j)} = k) = \frac{|\{x \in X \mid x^{(j)} = k\}|}{|X|} \quad (8.52)$$

**Tab. 8.3** Datensatz zu Abb. 8.5

Index	1	2	3	4	5	6	7	8
Körpergröße $h$ [m]	0, 1	0, 2	1, 8	0, 2	2, 1	1, 7	0, 1	1, 6
Beine $l$	0	2	2	4	4	2	4	2
Klasse	Fisch	Vogel	Mensch	Katze	Pferd	Mensch	Katze	Mensch

und für  $x^{(j)} = k$  beträgt die Entropie  $H(Z \mid x^{(j)} = k)$ . Die Betrachtung des Merkmals  $x^{(j)}$  liefert einen Informationsgewinn von  $H(Z)$  minus dem Erwartungswert von  $H(Z \mid x^{(j)} = k)$ .

$$g_j = H(Z) - \sum_{k=1}^{v_j} p(x^{(j)} = k) \cdot H(Z \mid x^{(j)} = k) \quad (8.53)$$

$$= H(Z) - \sum_{k=1}^{v_j} \frac{|\{x \in X \mid x^{(j)} = k\}|}{|X|} H(Z \mid x^{(j)} = k) \quad (8.54)$$

An jedem Knoten des (entropieoptimalen) Entscheidungsbaumes wird das Merkmal gewählt, das den höchsten erwarteten Informationsgewinn liefert. Für das Beispiel in Abb. 8.5 betrachten wir die Datensätze  $X$  und  $Y$  gemäß Tab. 8.3. Der Datensatz enthält einen Fisch, einen Vogel, ein Pferd, drei Menschen und zwei Katzen. Die Entropie beträgt also

$$\begin{aligned} H(Z) &= -\frac{1}{8} \log_2 \frac{1}{8} - \frac{1}{8} \log_2 \frac{1}{8} - \frac{3}{8} \log_2 \frac{3}{8} - \frac{2}{8} \log_2 \frac{2}{8} - \frac{1}{8} \log_2 \frac{1}{8} \\ &\approx 2.1556 \text{ bit} \end{aligned} \quad (8.55)$$

Der Entscheidungsbaum aus Abb. 8.5 oben betrachtet zuerst das Merkmal  $b \in \{0, 2, 4\}$ . Die entsprechenden Entropiewerte betragen

$$H(Z \mid l = 0) = -1 \log_2 1 = 0 \quad (8.56)$$

$$H(Z \mid l = 2) = -\frac{1}{4} \log_2 \frac{1}{4} - \frac{3}{4} \log_2 \frac{3}{4} \approx 0.8113 \text{ bit} \quad (8.57)$$

$$H(Z \mid l = 4) = -\frac{1}{3} \log_2 \frac{1}{3} - \frac{2}{3} \log_2 \frac{2}{3} \approx 0.9183 \text{ bit} \quad (8.58)$$

Am Wurzelknoten beträgt also der Informationsgewinn für Merkmal  $b$

$$\begin{aligned} g_l &= H(Z) - \frac{1}{8} H(Z \mid l = 0) - \frac{4}{8} H(Z \mid l = 2) - \frac{3}{8} H(Z \mid l = 4) \\ &\approx 1.4056 \text{ bit} \end{aligned} \quad (8.59)$$

Der Entscheidungsbaum aus Abb. 8.5 unten betrachtet zuerst das Merkmal  $h \in \{< 1m, > 1m\}$ . Die entsprechenden Entropiewerte betragen

$$H(Z \mid h < 1m) = -\frac{1}{4} \log_2 \frac{1}{4} - \frac{1}{4} \log_2 \frac{1}{4} - \frac{2}{4} \log_2 \frac{2}{4} \approx 1.5 \text{ bit} \quad (8.60)$$

1. Eingabe:  $X = \{x_1, \dots, x_n\} \subset \{1, \dots, v_1\} \times \dots \times \{1, \dots, v_p\}$ ,  $Y = \{y_1, \dots, y_n\} \subset \{1, \dots, c\}$
2. Erzeuge Wurzelknoten  $W$
3.  $\text{ID3}(X, Y, W, \{1, \dots, p\})$
4. Ausgabe: Baum mit Wurzelknoten  $W$

Prozedur  $\text{ID3}(X, Y, N, I)$

1. Wenn  $I$  leer ist oder alle  $Y$  gleich sind, dann Ende
2. Berechne Informationsgewinn  $g_i(X, Y) \forall i \in I$
3. Bestimme Gewinnermerkmal  $j = \operatorname{argmax}\{g_i(X, Y)\}$
4. Zerlege  $X, Y$  in  $v_j$  disjunkte Teilmengen

$$X_i = \{x_k \in X \mid x_k^{(j)} = i\}, \quad Y_i = \{y_k \in Y \mid x_k^{(j)} = i\}, \quad i = 1, \dots, v_j$$

5. Für alle  $i$  mit  $X_i \neq \{\}$ ,  $Y_i \neq \{\}$ 
  - Erzeuge neuen Knoten  $N_i$  und hänge ihn an  $N$
  - $\text{ID3}(X_i, Y_i, N_i, I \setminus \{j\})$

**Abb. 8.6** ID3-Algorithmus

$$H(Z \mid h > 1m) = -\frac{1}{4} \log_2 \frac{1}{4} - \frac{3}{4} \log_2 \frac{3}{4} \approx 0.8113 \text{ bit} \quad (8.61)$$

Am Wurzelknoten beträgt also der Informationsgewinn für Merkmal  $h$

$$\begin{aligned} g_h &= H(Z) - \frac{4}{8} H(Z \mid h < 1m) - \frac{4}{8} H(Z \mid h > 1m) \\ &\approx 1 \text{ bit} \end{aligned} \quad (8.62)$$

Am Wurzelknoten ist der Informationsgewinn durch Merkmal  $b$  also höher als der Informationsgewinn durch Merkmal  $h$ ,  $g_l > g_h$ . Der optimale Entscheidungsbaum ist also der in Abb. 8.5 oben gezeigte.

Optimale Entscheidungsbäume lassen sich mit dem *ID3-Algorithmus* (engl. *Iterative Dichotomiser*) [17], bestimmen, der in Abb. 8.6 dargestellt ist. Die rekursive Prozedur ID3 wird mit den Datensätzen  $X$ ,  $Y$ , dem Wurzelknoten und der Menge aller verfügbaren Merkmalsindizes aufgerufen. Die Rekursion endet, wenn keine weiteren Merkmale zu betrachten sind oder alle Daten zur selben Klasse gehören, dann werden keine weiteren Verzweigungen mehr benötigt. Andernfalls berechnet der Algorithmus den Informationsgewinn jedes verbleibenden Merkmals und bestimmt das Gewinnermerkmal. Die Daten werden anhand der Werte des Gewinnermerkmals zerlegt, und für jede nichtleere Teilmenge wird ein neuer Knoten erzeugt und angehängt. Für jeden angehängten Knoten wird rekursiv der entsprechende Unterbaum berechnet.

ID3 betrachtet diskrete Merkmale. Eine Erweiterung von ID3 auf kontinuierliche Merkmale ist *CART* (engl. *Classification and Regression Tree*) [6], mit dem entropieoptimale Intervallgrenzen (wie  $< 1m$  und  $> 1m$  in unserem Beispiel) gefunden werden. Bei *CHAID*

(engl. *Chi-Square Automatic Interaction Detection*) [11] werden die Intervallgrenzen mit einem Chi-Quadrat-Test bestimmt. Andere Erweiterungen von ID3 wie *C4.5* und *C5.0* erlauben fehlende Einträge, entfernen überflüssige Äste der Entscheidungsbäume (sog. *Pruning*), und können Entscheidungsbäume als *Regeln* repräsentieren.

Zu den größten Vorteilen der Entscheidungsbaumklassifikatoren gehört, dass nicht notwendigerweise alle Merkmale verwendet werden müssen, was besonders bei einer großer Anzahl von Merkmalen günstig ist, und dass sie eine Information über die Reihenfolge der Wichtigkeit der Merkmale liefern. Zu den größten Nachteilen gehört, dass sie entweder nur diskrete Merkmale verarbeiten können oder dass bei kontinuierlichen Merkmalen die Klassengrenzen immer stückweise parallel zu den Koordinatenachsen verlaufen, so dass kompliziert geformte nichtlineare Klassengrenzen nicht effizient abgebildet werden können.

### Übungsaufgaben

- 8.1.** Betrachten Sie einen Datensatz für zwei Klassen mit  $X_1 = \{(0, 0)\}$  und  $X_2 = \{(1, 0), (0, 1)\}$ . Welche Klassenwahrscheinlichkeiten liefert ein naiver Bayes-Klassifikator für den Merkmalsvektor  $(0, 0)$ ?
- 8.2.** Welche Diskriminanzgerade liefert eine Supportvektormaschine ohne Kernel-Trick für diesen Datensatz?
- 8.3.** Welche Klassifikationsregel liefert ein Nächster-Nachbar-Klassifikator für diesen Datensatz?
- 8.4.** Welche Klassifikationsregel liefert ein Nächste-3-Nachbarn-Klassifikator für diesen Datensatz?
- 8.5.** Skizzieren Sie ein ROC-Diagramm für diese vier Klassifikatoren, wenn Klasse 2 als positiv betrachtet wird.

---

### Literatur

1. D. W. Aha. Editorial: Lazy learning. *Artificial Intelligence Review* (Special Issue on Lazy Learning), 11(1–5):7–10, June 1997.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, New York, 1999.
3. T. Bayes. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, 53:370–418, 1763.
4. J. C. Bezdek and N. R. Pal. Two soft relatives of learning vector quantization. *Neural Networks*, 8(5):729–743, 1995.
5. J. C. Bezdek, T. R. Reichherzer, G. S. Lim, and Y. Attikiouzel. Multiple-prototype classifier design. *IEEE Transactions on Systems, Man, and Cybernetics C*, 28(1):67–79, 1998.
6. L. Breiman, J. H. Friedman, R. A. Olsen, and C. J. Stone. *Classification and Regression Trees*. Chapman & Hall, New Work, 1984.

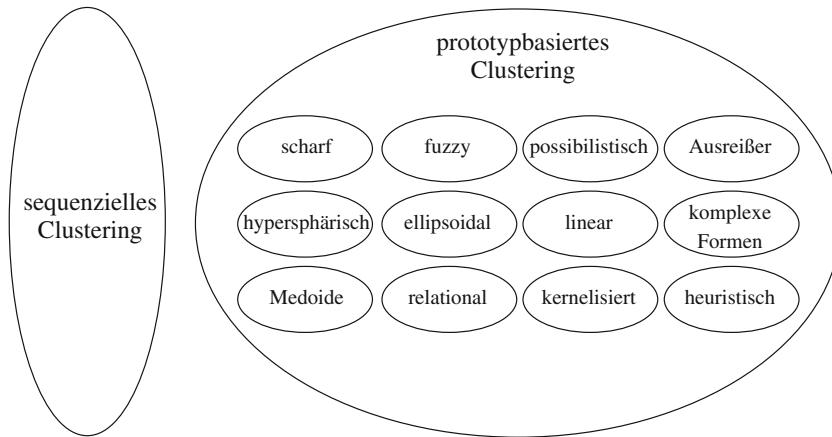
7. F. L. Chung and T. Lee. Fuzzy learning vector quantization. In *IEEE International Joint Conference on Neural Networks*, volume 3, pages 2739–2743, Nagoya, October 1993.
8. J. Davis and M. Goadrich. The relationship between precision–recall and ROC curves. In *International Conference on Machine Learning*, pages 233–240, 2006.
9. R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1974.
10. R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.
11. G. V. Kass. Significance testing in automatic interaction detection (AID). *Applied Statistics*, 24:178–189, 1975.
12. T. Kohonen. Learning vector quantization. *Neural Networks*, 1:303, 1988.
13. T. Kohonen. Improved versions of learning vector quantization. In *International Joint Conference on Neural Networks*, volume 1, pages 545–550, San Diego, June 1990.
14. J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society A*, 209:415–446, 1909.
15. J. Neyman and E. S. Pearson. Interpretation of certain test criteria for purposes of statistical inference, part I. *Joint Statistical Papers*, Cambridge University Press, pages 1–66, 1967.
16. M. J. D. Powell. Radial basis functions for multi-variable interpolation: a review. In *IMA Conference on Algorithms for Approximation of Functions and Data*, pages 143–167, Shrivenham, 1985.
17. J. R. Quinlan. Induction on decision trees. *Machine Learning*, 11:81–106, 1986.
18. L. Rokach and O. Maimon. *Data Mining with Decision Trees: Theory and Applications*. Machine Perception and Artificial Intelligence. World Scientific Publishing Company, 2008.
19. B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, 2002.
20. G. Shakhnarovich, T. Darrell, and P. Indyk. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. Neural Information Processing. MIT Press, 2006.
21. S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Academic Press, San Diego, 4th edition, 2008.

## Zusammenfassung

Clustering ist ein unüberwachtes Lernverfahren, bei dem unmarkierte Daten Clustern zugeordnet werden. Falls die zu clusternden Daten auch Klassen zugeordnet sind, so können die erhaltenen Clusterzugehörigkeiten möglicherweise den Klassenzugehörigkeiten entsprechen. Cluster- und Klassenzugehörigkeiten können jedoch auch verschieden sein. Cluster können mathematisch mit Hilfe von Mengen, Partitionsmatrizen und/oder Cluster-Prototypen spezifiziert werden. Sequenzielles Clustering (z. B. Single-Linkage, Complete-Linkage, Average-Linkage, Ward-Methode) lässt sich einfach implementieren, hat aber einen hohen Rechenaufwand. Partitionsbasiertes Clustering kann mit scharfen, unscharfen, possibilistischen oder robusten Clustermodellen definiert werden. Clusterprototypen können verschiedene geometrische Formen annehmen (z. B. Hypersphären, Ellipsoide, Linien, Hyperebenen, Kreise oder kompliziertere Formen). Relationale Clustermodelle finden Cluster in relationalen Daten. Dabei kann auch der Kernel-Trick angewendet werden. Die Clustertendenz gibt an, ob die Daten überhaupt Cluster enthalten. Clustervaliditätsmaße quantifizieren die Güte des Clusterergebnisses und ermöglichen, die Anzahl der Cluster abzuschätzen. Auch heuristische Methoden wie die selbstorganisierende Karte können zum Clustering verwendet werden.

## 9.1 Clusterpartitionen

Im vorangegangenen Kapitel über Klassifikation wurden markierte Datensätze mit Merkmalsvektoren  $X$  und Klassenzugehörigkeiten  $Y$  betrachtet. Häufig sind solche Klassenzugehörigkeiten nicht verfügbar oder nur aufwändig zu bestimmen, z. B. durch manuelle



**Abb. 9.1** Clusterverfahren

Zuordnung. Mit *Clusterverfahren* [18, 19] lassen sich Strukturen in unmarkierten Datensätzen, also reinen Merkmalsdaten, finden. Falls die zu clusternden Daten auch Klassen zugeordnet sind, so können die erhaltenen Clusterzugehörigkeiten möglicherweise den Klassenzugehörigkeiten entsprechen. Cluster- und Klassenzugehörigkeiten können jedoch auch verschieden sein. Abbildung 9.1 zeigt eine Übersicht der diesem Kapitel beschriebenen Clusterverfahren. Wir unterscheiden sequenzielle und prototypbasierte Verfahren. Bei den prototypbasierten Verfahren unterscheiden wir nach den Clustermodellen (scharf, fuzzy, possibilistisch, Ausreißer), den Prototypen (Hypersphären, Ellipsoide, Linien, Hyperebenen, Kreise oder kompliziertere Formen) und weiteren Spezifika wie medoid-basierte, relationale, Kernel- und heuristische Verfahren.

Abbildung 9.2 (links) zeigt ein Streudiagramm des Datensatzes

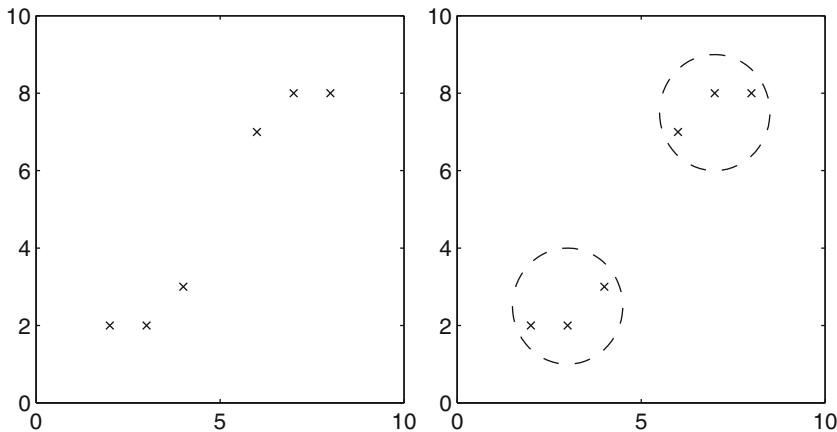
$$X = \{(2, 2), (3, 2), (4, 3), (6, 7), (7, 8), (8, 8), (5, 5)\} \quad (9.1)$$

der offenbar zwei Cluster enthält, und zwar links unten und rechts oben, deren Grenzen in Abb. 9.2 (rechts) durch gestrichelte Kreise markiert sind. Die Clusterstruktur partitioniert diesen Datensatz  $X$  in die paarweise disjunkten Teilmengen  $C_1 = \{x_1, x_2, x_3\}$  und  $C_2 = \{x_4, x_5, x_6\}$ . Im allgemeinen Fall ist die Zerlegung eines Datensatzes  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$  in seine (scharfe) Clusterstruktur definiert als die Partition von  $X$  in  $c \in \{2, 3, \dots, n - 1\}$  paarweise disjunkte Teilmengen  $C_1, \dots, C_c$ , so dass

$$X = C_1 \cup \dots \cup C_c \quad (9.2)$$

$$C_i \neq \{\} \quad \text{für alle } i = 1, \dots, c \quad (9.3)$$

$$C_i \cap C_j = \{\} \quad \text{für alle } i, j = 1, \dots, c, \quad i \neq j \quad (9.4)$$

**Abb. 9.2** Ein Datensatz und seine Clusterstruktur

1. Eingabe:  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$
2. Initialisiere  $\Gamma_n = \{\{x_1\}, \dots, \{x_n\}\}$
3. Für  $c = n-1, n-2, \dots, 1$ 
  - $(i, j) = \underset{(C_r, C_s) \in \Gamma_c}{\operatorname{argmin}} d(C_r, C_s)$
  - $\Gamma_c = (\Gamma_{c+1} \setminus C_i \setminus C_j) \cup (C_i \cup C_j)$
4. Ausgabe: Partitionen  $\Gamma_1, \dots, \Gamma_h$

**Abb. 9.3** Sequenzielle agglomerative hierarchische nichtüberlappende (SAHN) Clusteranalyse

## 9.2 Sequenzielles Clustering

Eine wichtige Familie *sequenzieller* Clustermethoden ist die *sequenzielle agglomerative hierarchische nichtüberlappende (SAHN)* Clusteranalyse [30], die in Abb. 9.3 dargestellt ist. Zu Beginn interpretiert SAHN jeden der  $n$  Punkte als einzelnen Cluster, so dass sich initial die Partition  $\Gamma_n = \{\{x_1\}, \dots, \{x_n\}\}$  ergibt. In jedem Schritt bestimmt SAHN aus der Menge der Cluster das Paar mit dem geringsten Abstand und fasst dieses Paar zu einem Cluster zusammen, so dass sich die Anzahl der Cluster in jedem Schritt um eins verringert. SAHN terminiert, wenn die gewünschte Anzahl von Clustern erreicht ist oder wenn alle Punkte zu einem einzigen Cluster zusammengefasst sind, was der Partition  $\Gamma_1 = \{\{x_1, \dots, x_n\}\}$  entspricht. Der Abstand zwischen einem Paar von Clustern kann auf verschiedene Weise aus den paarweisen Abständen der zugehörigen Punkte berechnet werden:

- Minimalabstand (engl. *Single Linkage*)

$$d(C_r, C_s) = \min_{x \in C_r, y \in C_s} d(x, y) \quad (9.5)$$

- Maximalabstand (engl. *Complete Linkage*)

$$d(C_r, C_s) = \max_{x \in C_r, y \in C_s} d(x, y) \quad (9.6)$$

- mittlerer Abstand (engl. *Average Linkage*)

$$d(C_r, C_s) = \frac{1}{|C_r| \cdot |C_s|} \sum_{x \in C_r, y \in C_s} d(x, y) \quad (9.7)$$

Diese drei Maße können sowohl für Merkmalsdaten als auch für relationale Daten verwendet werden. Zwei weitere Abstandsmaße, die jedoch nur für Merkmalsdaten geeignet sind, sind

- Abstand der Zentren

$$d(C_r, C_s) = \left\| \frac{1}{|C_r|} \sum_{x \in C_r} x - \frac{1}{|C_s|} \sum_{x \in C_s} x \right\| \quad (9.8)$$

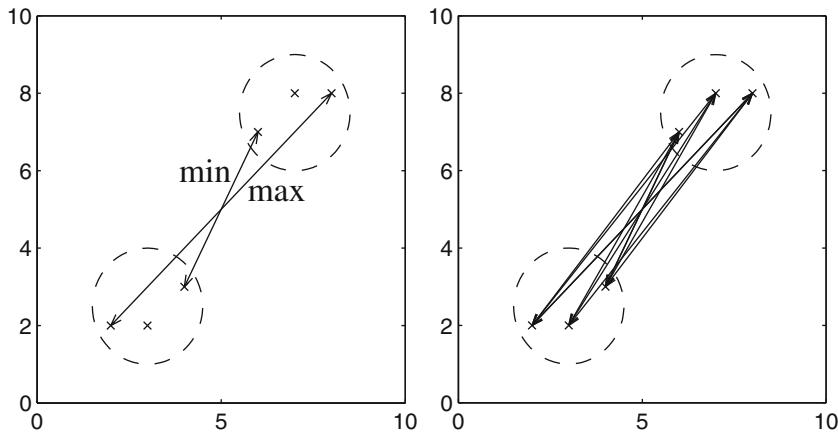
- Ward-Methode [31]

$$d(C_r, C_s) = \frac{|C_r| \cdot |C_s|}{|C_r| + |C_s|} \left\| \frac{1}{|C_r|} \sum_{x \in C_r} x - \frac{1}{|C_s|} \sum_{x \in C_s} x \right\| \quad (9.9)$$

Abbildung 9.4 zeigt ein Beispiel für die ersten drei Fälle (Single, Complete und Average Linkage). Das linke Diagramm zeigt die Minimal- und Maximalabstände (Single und Complete Linkage) zwischen den Clustern aus Abb. 9.2. Das rechte Diagramm zeigt alle Abstände, deren Mittelwert bei Average Linkage verwendet wird. Single Linkage (9.5) ist eines der verbreitetsten Abstandsmaße für sequenzielles Clustering, kann jedoch zu langen, kettenförmigen Clustern führen. SAHN-Algorithmen werden oft einfach nach dem verwendeten Abstandsmaß benannt, z. B. heißt SAHN mit (9.5) einfach *Single-Linkage-Clustering*.

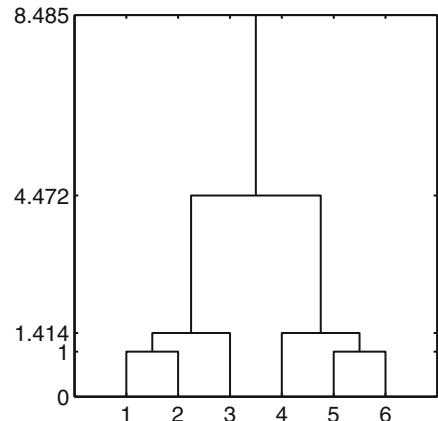
Die hierarchische Struktur der mit SAHN bestimmten Partitionen kann mit einem sogenannten *Dendrogramm* dargestellt werden. Abbildung 9.5 zeigt das Dendrogramm für Single-Linkage-Clustering unseres Datensatzes (9.1), das darstellt, wie die Punkte  $x_1, \dots, x_6$  (Indizes auf der horizontalen Achse) sukzessive zusammengefasst werden. Auf der vertikalen Achse sind die entsprechenden Single-Linkage-Abstände aufgetragen. Die Single-Linkage-Partitionen lauten

$$\Gamma_0 = \{\{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}, \{x_5\}, \{x_6\}\} \quad (9.10)$$



**Abb. 9.4** Abstände zwischen Clustern (links: Single und Complete Linkage, rechts: Average Linkage)

**Abb. 9.5** Dendrogramm



$$\Gamma_1 = \Gamma_2 = \{\{x_1, x_2\}, \{x_3\}, \{x_4\}, \{x_5, x_6\}\} \quad (9.11)$$

$$\Gamma_3 = \Gamma_4 = \{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6\}\} \quad (9.12)$$

$$\Gamma_5 = \{\{x_1, x_2, x_3, x_4, x_5, x_6\}\} = \{X\} \quad (9.13)$$

Der Algorithmus *DBSCAN* (engl. *Density Based Spatial Clustering of Applications With Noise*) ist eine Variante von SAHN, die sogenannte *Dichteschätzer* statt der Abstandsmaße benutzt, um Cluster zusammenzufassen [28].

Die umgekehrte Variante von SAHN ist die *sequenzielle divisive hierarchische nicht-überlappende (SDHN)* Clusteranalyse, die zunächst alle Punkte in einem Cluster zusammenfasst,  $\Gamma_0 = \{\{x_1, \dots, x_n\}\}$ ,  $c_0 = 1$ , und dann sukzessive Cluster in Teilcluster aufteilt. SDHN ist weniger verbreitet als SAHN. Ein Beispiel für SDHN ist in [11] beschrieben.

Zu den größten Vorteilen von SAHN gehört die direkte Anwendbarkeit auf relationale Daten (Single, Complete und Average Linkage) und dass eine hierarchische Clusterstruktur gefunden wird. Zu den größten Nachteilen gehört der hohe Rechenaufwand. Der Algorithmus in Abb. 9.3 hat die Komplexität  $o(n^3)$ , es sind jedoch auch effizientere Algorithmen für SAHN ( $o(n^2 \log n)$ ) und Single Linkage ( $o(n^2)$ ) bekannt [10].

### 9.3 Prototypbasiertes Clustering

Im vorigen Abschnitt wurde eine Partition von  $X$  durch eine *Partitionsmenge*  $\Gamma$  von disjunkten Teilmengen von  $X$  dargestellt. Eine äquivalente Darstellung ist eine *Partitionsmatrix*  $U$  mit den Elementen

$$u_{ik} = \begin{cases} 1 & \text{falls } x_k \in C_i \\ 0 & \text{falls } x_k \notin C_i \end{cases} \quad (9.14)$$

$i = 1, \dots, c$ ,  $k = 1, \dots, n$ . Jeder Zugehörigkeitswert  $u_{ik}$  bestimmt, ob  $x_k$  zu  $C_i$  gehört. Für nichtleere Cluster fordern wir

$$\sum_{k=1}^n u_{ik} > 0, \quad i = 1, \dots, c \quad (9.15)$$

und für paarweise disjunkte Cluster

$$\sum_{i=1}^c u_{ik} = 1, \quad k = 1, \dots, n \quad (9.16)$$

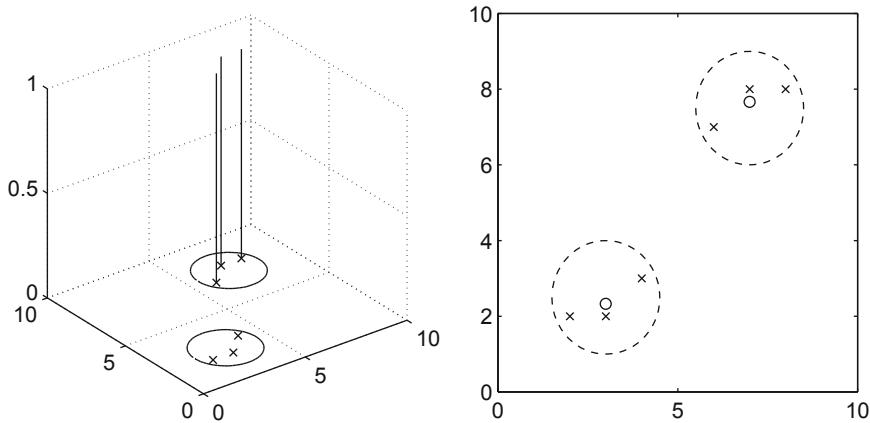
Abbildung 9.6 (links) zeigt unseren Datensatz (9.1) mit vertikalen Balken, die die zweite Zeile der Partitionsmatrix darstellen. Die zweite Zeile der Partitionsmatrix ist eins für alle Elemente des zweiten Clusters und null für alle anderen Punkte. Die drei Balken an den Punkten  $x_4, \dots, x_6$  haben daher die Höhe eins, und die übrigen Balken sind nicht sichtbar (Höhe null).

Neben Partitionsmengen und Partitionsmatrizen können Cluster von Merkmalsdaten auch durch *Prototypen* repräsentiert werden. Z. B. kann jeder Cluster durch ein (einzelnes) Zentrum  $v_i$ ,  $i = 1, \dots, c$ , repräsentiert werden, so dass die Clusterstruktur durch die Menge der Clusterzentren

$$V = \{v_1, \dots, v_c\} \subset \mathbb{R}^p \quad (9.17)$$

definiert wird. Für einen gegebenen Datensatz  $X$  können die Clusterzentren  $V$  und die Zuordnung der Datenpunkte  $X$  zu den  $c$  Clustern durch Optimierung des *c-Means-Clustermodells* (*CM*) [1] gefunden werden. Die Kostenfunktion des c-Means-Clustermodells ist die Summe der quadratischen Abstände zwischen den Clusterzentren und den zugehörigen Datenpunkten.

$$J_{CM}(U, V; X) = \sum_{i=1}^c \sum_{x_k \in C_i} \|x_k - v_i\|^2 = \sum_{i=1}^c \sum_{k=1}^n u_{ik} \|x_k - v_i\|^2 \quad (9.18)$$



**Abb. 9.6** Clusterzugehörigkeiten und Clusterzentren

Zur Minimierung von  $J_{CM}$  bestimmen wir für jedes  $k$  das Minimum  $\|x_k - v_i\|$ , setzen die entsprechende Zugehörigkeit  $u_{ik}$  auf eins und alle übrigen Zugehörigkeiten  $u_{jk}$ ,  $j \neq i$ , auf null.

$$u_{ik} = \begin{cases} 1 & \text{falls } \|x_k - v_i\| = \min_{j=1, \dots, c} \|x_k - v_j\| \\ 0 & \text{sonst} \end{cases} \quad (9.19)$$

Im Falle mehrfacher Minima wird nur einer der Cluster ausgewählt, z. B. zufällig. Die notwendige Bedingung für Extrema von (9.18)

$$\frac{\partial J_{CM}(U, V; X)}{\partial v_i} = 0, \quad i = 1, \dots, c \quad (9.20)$$

liefert die Clusterzentren

$$v_i = \frac{1}{|C_i|} \sum_{x_k \in C_i} x_k = \frac{\sum_{k=1}^n u_{ik} x_k}{\sum_{k=1}^n u_{ik}} \quad (9.21)$$

Die Clusterzentren sind also die Mittelwerte der Punkte des entsprechenden Clusters, und die Datenpunkte werden dem Cluster mit dem nächsten Zentrum zugeordnet. Die optimale Partition  $U$  hängt also von den Clusterzentren  $V$  ab, und die optimalen Clusterzentren  $V$  hängen von der Partition  $U$  ab.  $U$  und  $V$  müssen daher mit einer alternierenden Optimierung bestimmt werden, wie in Abb. 9.7 gezeigt. Diese Variante von AO initialisiert  $V$ , berechnet abwechselnd  $U$  und  $V$  und terminiert in Abhängigkeit von  $V$ . Die umgekehrte Variante initialisiert  $U$ , berechnet abwechselnd  $V$  und  $U$  und terminiert in Abhängigkeit von  $U$ . Zur Prüfung der Terminierungsbedingung müssen also in der ersten Variante  $c \cdot p$  Elemente von  $V$  verglichen werden, und in der zweiten Variante  $c \cdot n$  Elemente von  $U$ . Für niedrigdimensionale Daten ( $p \ll n$ ) ist die erste Variante also effizienter. Für sehr hochdimensionale Daten ( $p \gg n$ ) ist dagegen die zweite Variante effizienter.

1. Eingabe: Daten  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$ ,  
Clusteranzahl  $c \in \{2, \dots, n-1\}$ ,  
maximale Schrittzahl  $t_{\max}$ ,  
Abstandsmaß  $\|\cdot\|$ ,  
Abstandsmaß zur Terminierung  $\|\cdot\|_\varepsilon$ ,  
Terminierungsgrenze  $\varepsilon$
2. Initialisiere Prototypen  $V^{(0)} \subset \mathbb{R}^p$
3. Für  $t = 1, \dots, t_{\max}$ 
  - Berechne  $U^{(t)}(V^{(t-1)}, X)$
  - Berechne  $V^{(t)}(U^{(t)}, X)$
  - Falls  $\|V^{(t)} - V^{(t-1)}\|_\varepsilon \leq \varepsilon$ , dann Ende
4. Ausgabe: Partitionsmatrix  $U \in [0, 1]^{c \times n}$ ,  
Prototypen  $V = \{v_1, \dots, v_c\} \in \mathbb{R}^p$

**Abb. 9.7** Alternierende Optimierung von Clustermodellen

## 9.4 Fuzzy-Clustering

Das c-Means-Clustermodell liefert in der Regel gute Ergebnisse, wenn die Cluster klar getrennt sind und weder lokale noch globale Ausreißer enthalten. Zu dem im vorigen Abschnitt betrachteten Datensatz mit sechs Punkten fügen wir einen siebten Punkt  $x_7 = (5, 5)$  hinzu und erhalten den Datensatz

$$X = \{(2, 2), (3, 2), (4, 3), (6, 7), (7, 8), (8, 8), (5, 5)\} \quad (9.22)$$

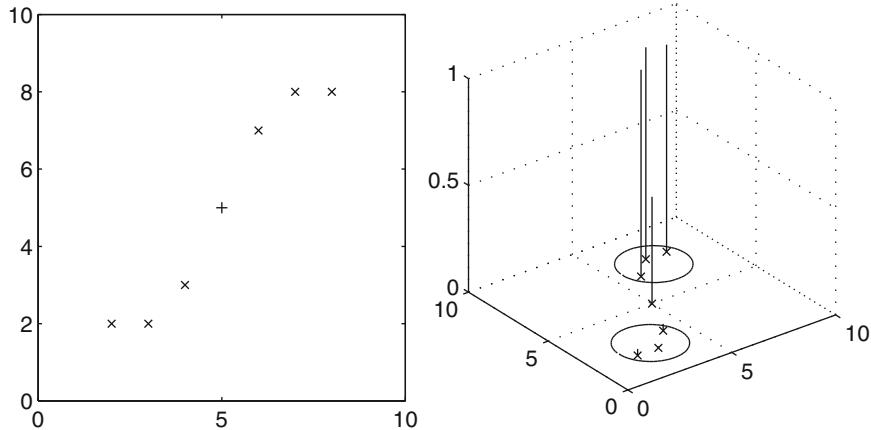
Dieser Datensatz ist in Abb. 9.8 (links) gezeigt, wobei der neue Punkt als Pluszeichen dargestellt ist. Gemäß c-Means-Clustermodell muss der zusätzliche Punkt (Pluszeichen) eindeutig einem der beiden Cluster zugeordnet werden. Dies widerspricht der intuitiven Erwartung. Stattdessen erwarten wir, dass dieser Punkt jeweils zur Hälfte beiden Clustern zugeordnet wird. Um eine teilweise Zuordnung von Punkten zu Clustern zu ermöglichen, werden für die Einträge der Partitionsmatrix alle Werte des Einheitsintervalls  $u_{ik} \in [0, 1]$  zugelassen,  $i = 1, \dots, c$ ,  $k = 1, \dots, n$ . Analog zu (9.15) wird gefordert, dass die Cluster nicht leer sind,

$$\sum_{k=1}^n u_{ik} > 0, \quad i = 1, \dots, c \quad (9.23)$$

und analog zu (9.16) wird gefordert, dass für jeden Punkt die Summe der Clusterzugehörigkeiten eins beträgt,

$$\sum_{i=1}^c u_{ik} = 1, \quad k = 1, \dots, n \quad (9.24)$$

Die Bedingung (9.24) erinnert an die Normalisierung von Wahrscheinlichkeitsverteilungen, jedoch sind die Clusterzugehörigkeiten nicht probabilistisch sondern deterministisch,



**Abb. 9.8** Datensatz mit einem Punkt, der zu beiden Clustern gehört (links), und entsprechende Fuzzy-Partition (rechts)

aber unscharf. Zum Unterschied zwischen Wahrscheinlichkeiten und Zugehörigkeitswerten verweisen wir auf [3]. Abbildung 9.8 (rechts) zeigt die Zugehörigkeitswerte einer Zeile einer Fuzzy-Partition. Die Zugehörigkeitswerte des neuen Punktes  $x_7$  betragen erwartungsgemäß  $u_{17} = u_{27} = 1/2$ . Die Zugehörigkeitswerte  $u_{21}, u_{22}, u_{23}$  sind klein, können aber ungleich null sein, und die Zugehörigkeitswerte  $u_{24}, u_{25}, u_{26}$  können ungleich eins sein, so dass die Bedingung (9.24) erfüllt ist. Jeder Punkt kann zu gewissen Graden mehreren Clustern zugeordnet werden.

Für einen gegebenen Datensatz  $X$  können die Clusterzentren  $V$  und die Fuzzy-Partitionsmatrix  $U$  durch Optimierung des *Fuzzy-c-Means-Clustermodells (FCM)* [2, 7] gefunden werden.

$$J_{\text{FCM}}(U, V; X) = \sum_{k=1}^n \sum_{i=1}^c u_{ik}^m \|x_k - v_i\|^2 \quad (9.25)$$

mit den Randbedingungen  $0 \leq u_{ik} \leq 1$ , (9.23) und (9.24). Der Parameter  $m > 1$  bestimmt den Grad der Unschärfe der Cluster. Für  $m \rightarrow 1$  wird FCM zu CM. Für  $m \rightarrow \infty$  erhält jeder Punkt aus  $X \setminus V$  die gleiche Zugehörigkeit  $u_{ik} = 1/c$ ,  $i = 1, \dots, c$ ,  $k = 1, \dots, n$ , zu jedem Cluster. Häufig wird der Wert  $m = 2$  verwendet. Zur Optimierung des FCM-Modells kann die Randbedingung (9.23) formal ignoriert werden, aber die Randbedingung (9.24) muss berücksichtigt werden, um die Triviallösung  $u_{ik} = 0$ ,  $i = 1, \dots, c$ ,  $k = 1, \dots, n$ , zu verhindern. Zur Optimierung von  $J_{\text{FCM}}$  unter der Randbedingung (9.24) betrachten wir die Lagrange-Funktion (siehe Anhang)

$$F_{\text{FCM}}(U, V, \lambda; X) = \sum_{k=1}^n \sum_{i=1}^c u_{ik}^m \|x_k - v_i\|^2 - \sum_{k=1}^n \lambda_k \cdot \left( \sum_{i=1}^c u_{ik} - 1 \right) \quad (9.26)$$

Die notwendigen Bedingungen für Extrema liefern

$$\left. \begin{aligned} \frac{\partial F_{\text{FCM}}}{\partial \lambda_k} &= 0 \\ \frac{\partial F_{\text{FCM}}}{\partial u_{ik}} &= 0 \end{aligned} \right\} \Rightarrow u_{ik} = 1 / \sum_{j=1}^c \left( \frac{\|x_k - v_i\|}{\|x_k - v_j\|} \right)^{\frac{2}{m-1}} \quad (9.27)$$

$$\frac{\partial J_{\text{FCM}}}{\partial v_i} = 0 \Rightarrow v_i = \frac{\sum_{k=1}^n u_{ik}^m x_k}{\sum_{k=1}^n u_{ik}^m} \quad (9.28)$$

Ein Vergleich von (9.25) mit (9.18), (9.27) mit (9.19) und (9.28) mit (9.21) bestätigt, dass FCM für  $m \rightarrow 1$  in CM übergeht. Auch in FCM hängt  $U$  von  $V$  ab und  $V$  von  $U$ , so dass  $U$  und  $V$  durch alternierende Optimierung (AO) gemäß Abb. 9.7 bestimmt werden können. In der Literatur wurden auch andere Verfahren zur Optimierung von FCM vorgeschlagen, z. B. evolutionäre Algorithmen [4] oder schwarmbasierte Verfahren [27].

FCM liefert in der Regel gute Ergebnisse, auch wenn die Cluster überlappen und die Daten verrauscht sind, ist aber empfindlich gegenüber Ausreißern. Ausreißer sind näherungsweise gleich weit von allen Clusterzentren entfernt und erhalten daher die Zugehörigkeiten  $u_{ik} \approx 1/c$  zu allen Clustern,  $i = 1, \dots, c$ . Ein Ausreißer erhält also die gleichen Zugehörigkeiten wie ein Datenpunkt, der genau zwischen allen Clusterzentren liegt, so wie der mittlere Punkt in Abb. 9.8. Intuitiv wäre dagegen zu erwarten, dass ein Ausreißer eine geringe Zugehörigkeit zu allen Clustern erhält, denn er kann für keinen Cluster als repräsentativ betrachtet werden. Dies kann erreicht werden, indem die Normalisierungsbedingung (9.24) aufgehoben wird und stattdessen in der Kostenfunktion ein Strafterm addiert wird, der die Triviallösung  $u_{ik} = 0, i = 1, \dots, c, k = 1, \dots, n$ , verhindert. Hierdurch erhalten wir das *possibilistische c-Means-Clustermodell (PCM)* [22] mit der Kostenfunktion

$$J_{\text{PCM}}(U, V; X) = \sum_{k=1}^n \sum_{i=1}^c u_{ik}^m \|x_k - v_i\|^2 + \sum_{i=1}^c \eta_i \sum_{k=1}^n (1 - u_{ik})^m \quad (9.29)$$

unter der Randbedingung (9.23) mit den Clustergrößen  $\eta_1, \dots, \eta_c > 0$ . Die notwendigen Bedingungen für Extrema liefern (9.28) und die sogenannte *Cauchy-Funktion*

$$u_{ik} = 1 / \sum_{j=1}^c \left( 1 + \left( \frac{\|x_k - v_i\|^2}{\eta_i} \right)^{\frac{1}{m-1}} \right) \quad (9.30)$$

Ein anderer Ansatz zum Umgang mit Ausreißern im Fuzzy-Clustering verwendet einen zusätzlichen Cluster speziell für Ausreißer. Die Zugehörigkeit jedes Punkts in diesem Ausreißer-Cluster ist gleich eins minus der Summe der Zugehörigkeiten in den regulären Clustern. Der Ausreißercluster wird nicht durch ein Zentrum beschrieben, sondern dadurch, dass alle Punkte zu diesem Cluster den gleichen Abstand  $\delta > 0$  haben, der größer

gewählt werden muss als die Abstände zwischen regulären Punkten (keine Ausreißer) und regulären Clusterzentren. Hierdurch erhalten wir das *Ausreißerclustering*, (engl. *Noise Clustering, NC*) [9] mit der Kostenfunktion

$$J_{NC}(U, V; X) = \sum_{k=1}^n \sum_{i=1}^c u_{ik}^m \|x_k - v_i\|^2 + \sum_{k=1}^n \left( 1 - \sum_{j=1}^c u_{jk} \right)^m \delta^2 \quad (9.31)$$

Die notwendigen Bedingungen für Extrema liefern (9.28) und

$$u_{ik} = 1 / \left( \sum_{j=1}^c \left( \frac{\|x_k - v_i\|}{\|x_k - v_j\|} \right)^{\frac{2}{m-1}} + \left( \frac{\|x_k - v_i\|}{\delta} \right)^{\frac{2}{m-1}} \right) \quad (9.32)$$

Für die Abstände  $\|\cdot\|$  in den Kostenfunktionen von CM (9.18), FCM (9.25), PCM (9.29), NC (9.31) und in den entsprechenden Gleichungen für  $u_{ik}$  (9.19, 9.27, 9.30, 9.32) kommen alle Abstandsmaße aus Kap. 2 in Frage, z. B. die Euklidische Norm oder die Mahalanobis-Norm. Eine Variante der Mahalanobis-Norm betrachtet die Kovarianzmatrizen der einzelnen Cluster anstatt des gesamten Datensatzes. Die Kovarianz innerhalb Cluster  $i = 1, \dots, c$  ist definiert als

$$S_i = \sum_{k=1}^n u_{ik}^m (x_k - v_i)^T (x_k - v_i) \quad (9.33)$$

und entspricht der Kovarianz innerhalb einer Klasse (8.24) bei der linearen Diskriminanzanalyse (Kap. 8). Diese Kovarianz wird verwendet, um für jeden Cluster eine lokale Matrix-Norm zu definieren mit

$$A_i = \sqrt[m]{\rho_i \det(S_i)} S_i^{-1} \quad (9.34)$$

Die Normalisierung in (9.34) stellt sicher, dass jeder Cluster das Hypervolumen  $\det(A_i) = \rho_i$  besitzt. Häufig wird  $\rho_1 = \dots = \rho_c = 1$  gewählt. FCM mit der Mahalanobis-Norm innerhalb einzelner Cluster heißt auch *Gustafson-Kessel-Clustermodell* (GK) [13] mit der Kostenfunktion

$$J_{GK}(U, V; X) = \sum_{k=1}^n \sum_{i=1}^c u_{ik}^m (x_k - v_i) A_i (x_k - v_i)^T \quad (9.35)$$

Die Mahalanobis-Norm innerhalb einzelner Cluster kann auch in Verbindung mit CM, PCM oder NC verwendet werden.

Die Verwendung der Euklidischen Norm führt zu hypersphärischen Clustern, und die Verwendung der Mahalanobis-Norm führt zu hyperellipsoiden Clustern. Wir erweitern den Begriff der Prototypen von Clustern auf kompliziertere geometrische Strukturen im Merkmalsraum. Eine Gerade, Ebene oder Hyperebene lässt sich definieren durch einen Punkt  $v_i \in \mathbb{R}^p$  und einen oder mehrere Richtungsvektoren  $d_{i1}, \dots, d_{iq}$ ,  $q \in \{1, \dots, p-1\}$ ,

$q = 1$  für Geraden und  $q = 2$  für Ebenen. Der Abstand eines Punkts  $x_k$  von einer solchen Geraden, Ebene oder Hyperebene ergibt sich durch orthogonale Projektion als

$$d(x_k, v_i, d_{i1}, \dots, d_{iq}) = \sqrt{\|x_k - v_i\|^2 - \sum_{l=1}^q (x_k - v_i) d_{il}^T} \quad (9.36)$$

Jedes oben beschriebene Clustermodell (CM, FCM, PCM, NC) lässt sich für die Erkennung von Linien, Ebenen oder Hyperebenen verwenden, indem die Abstände  $\|x_k - v_i\|$  durch (9.36) ersetzt werden. Für FCM erhalten wir beispielsweise das *Fuzzy-c-Varieties-Clustermodell (FCV)* [5] mit der Kostenfunktion

$$J_{\text{FCV}}(U, V, D; X) = \sum_{k=1}^n \sum_{i=1}^c u_{ik}^m \left( \|x_k - v_i\|^2 - \sum_{l=1}^q (x_k - v_i) d_{il}^T \right) \quad (9.37)$$

Für  $q = 1$  heißt FCV auch *Fuzzy-c-Lines-Clustermodell (FCL)*. In FCV/FCL wird  $U$  berechnet mit

$$u_{ik} = 1 / \sum_{j=1}^c \left( \frac{\|x_k - v_i\|^2 - \sum_{l=1}^q (x_k - v_i) d_{il}^T}{\|x_k - v_j\|^2 - \sum_{l=1}^q (x_k - v_j) d_{jl}^T} \right)^{\frac{1}{m-1}} \quad (9.38)$$

$V$  mit (9.28), und  $D$  sind die Eigenwerte der Kovarianzmatrizen innerhalb der Cluster (9.33).

$$d_{ij} = \text{eig}_j \sum_{k=1}^n u_{ik}^m (x_k - v_i)^T (x_k - v_i), \quad j = 1, \dots, q \quad (9.39)$$

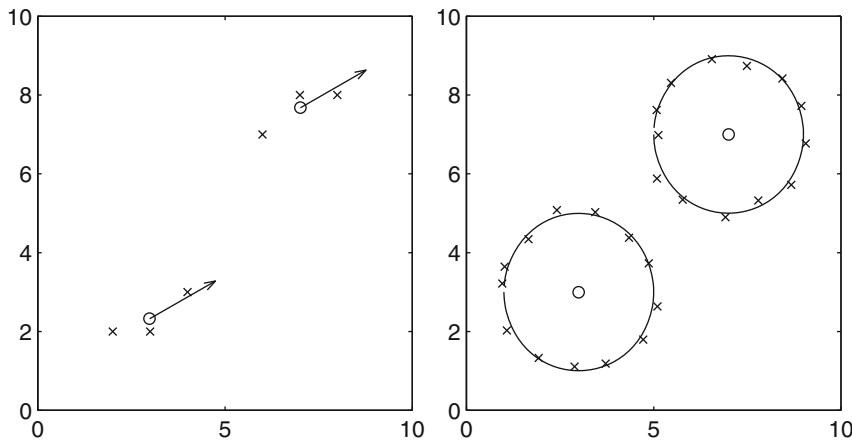
Als drittes Beispiel für nicht-sphärische Prototypen (neben GK und FCV/FCL) betrachten wir die sogenannten *Elliptotypes* [6] mit dem Abstand

$$d(x_k, v_i, d_{i1}, \dots, d_{iq}) = \sqrt{\|x_k - v_i\|^2 - \alpha \cdot \sum_{l=1}^q (x_k - v_i) d_{il}^T} \quad (9.40)$$

und dem Parameter  $\alpha \in [0, 1]$ . Die Elliptotypes sind für  $\alpha = 0$  Punkte und für  $\alpha = 1$  Geraden, Ebenen oder Hyperebenen. Mit Elliptotypes kann beispielsweise das *Fuzzy-c-Elliptotypes-Clustermodell (FCE)* [6] definiert werden. Abbildung 9.9 (links) zeigt ein Ergebnis von FCE mit den Parametern  $c = 2, m = 2, q = 1, \alpha = 0.5$  und  $t_{\max} = 100$ . Die Clusterzentren  $v_1$  und  $v_2$  sind als Kreise dargestellt und die Richtungsvektoren  $d_{11}$  und  $d_{21}$  als Pfeile. Die lokal linearen Strukturen werden gut erkannt.

Ein Beispiel für kompliziertere geometrische Prototypen sind Kreise. Jeder Kreis kann durch ein Zentrum  $v_i$  und einen Radius  $r_i$  definiert werden. Der Abstand zwischen einem Punkt  $x_k$  und einem solchen Kreis ist

$$d(x_k, v_i, r_i) = |\|x_k - v_i\| - r_i|. \quad (9.41)$$



**Abb. 9.9** Erkennung von Linien und Kreisen mit Clustering

womit beispielsweise das *Fuzzy-c-Shells-Clustermodell (FCS)* [8] definiert wird. Bei FCS werden die Zentren mit (9.28) berechnet, und die Radien ergeben sich zu

$$r_i = \frac{\sum_{k=1}^n u_{ik}^m \|x_k - v_i\|}{\sum_{k=1}^n u_{ik}^m} \quad (9.42)$$

Abbildung 9.9 (rechts) zeigt das Ergebnis von FCS für einen Datensatz mit zwei Kreisen. Durch Kombination der Clustermodelle CM, FCM, PCM und NC mit geeigneten Abstandsmaßen lassen sich zahlreiche weitere Clustermodelle für komplizierte geometrische Strukturen definieren.

---

## 9.5 Relationales Clustering

In diesem Abschnitt werden Clustermethoden für *relationale* Daten betrachtet, die durch symmetrische Abstandsmatrizen  $D$ ,  $d_{ii} = 0$ ,  $d_{ij} = d_{ji}$ ,  $i, j = 1, \dots, n$ , definiert sind [26]. Die sequenziellen Clustermethoden (SAHN, SDHN) aus Abschn. 9.2 basieren auf Abständen zwischen Paaren von Punkten und sind somit für relationale Daten einsetzbar. Die prototypbasierten Clustermethoden (CM, FCM, PCM, NC) aus den letzten beiden Abschnitten benötigen dagegen Merkmalsdaten  $X \subset \mathbb{R}^p$ . Relationale Daten können zunächst z. B. mit mehrdimensionaler Skalierung oder mit der Sammon-Abbildung auf Merkmalsdaten abgebildet werden (Kap. 4), auf die dann anschließend prototypbasierte Clustermethoden angewendet werden können. Mit geeigneten Modifikationen sind prototypbasierte Clustermethoden jedoch auch direkt auf relationale Daten anwendbar, ohne explizite Berechnung von Merkmalsdaten.

Prototypbasierte Clustermethoden basieren auf Abständen zwischen Merkmalsvektoren und Clusterzentren. Wenn wir verlangen, dass die Clusterzentren aus der Menge der Merkmalsvektoren gewählt werden müssen,  $V \subseteq X$ , sogenannte *Medoide* [23], so sind die Abstände zwischen Merkmalsvektoren und Clusterzentren in  $D$  enthalten und die Methoden können auf relationale Daten angewandt werden. Wir bezeichnen die entsprechenden Clustermodelle als *relationale c-Medoide (RCMdd)*, *relationale Fuzzy-c-Medoide (RFCMdd)*, *relationale possibilistische c-Medoide (RPCMdd)* und *relationales Ausreißerclustering mit Medoiden (RNCMdd)*. Durch die Beschränkung auf Medoide bleiben die jeweiligen Clustermodelle unverändert, es wird lediglich die Nebenbedingung  $V \subseteq X$  ergänzt, die sicherstellt, dass  $U$  aus  $D$  bestimmt werden kann. Die Bestimmung der Medoide erfolgt durch vollständige Suche, die hier am Beispiel von RFCMdd illustriert wird. Der Beitrag von Cluster  $i$  zur Kostenfunktion  $J_{\text{RFCMdd}} = J_{\text{FCM}}$  ist der Summand

$$J_i^* = \sum_{k=1}^n u_{ik}^m \|x_k - v_i\|^2 \quad (9.43)$$

und somit  $J = \sum_{i=1}^c J_i^*$ . Ohne Beschränkung der Allgemeinheit nehmen wir an, dass  $v_i = x_j$ , also  $\|v_i - x_k\| = d_{jk}$  und weiter

$$J_i^* = J_{ij} = \sum_{k=1}^n u_{ik}^m r_{jk}^2 \quad (9.44)$$

Die optimale Wahl der Medoide ist also  $v_i = x_{w_i}$ ,  $i = 1, \dots, n$ , mit

$$w_i = \operatorname{argmin}\{J_{i1}, \dots, J_{in}\} \quad (9.45)$$

Diese vollständige Suche führt zu einem hohen Rechenaufwand.

Ein anderer Ansatz zur Verwendung von prototypbasierten Clustermethoden für relationale Daten berechnet die Prototypen nur implizit, indem die Gleichung für optimale Prototypen in die Kostenfunktion eingesetzt wird, die sogenannte *Reformulierung* [15]. Die Clustermodelle, die durch Reformulierung aus CM, FCM, PCM und NC entstehen, bezeichnen wir als *relationale c-Means (RCM)*, *relationale Fuzzy-c-Means (RFCM)*, *relationale possibilistische c-Means (RPCM)* und *relationales Ausreißerclustering (RNC)* [16]. Die Reformulierung wird hier am Beispiel von RFCM illustriert. Falls für  $\|\cdot\|$  die Euklidische Norm gewählt wird, so liefert Einsetzen von (9.28) in (9.25) die Kostenfunktion

$$J_{\text{RFCM}}(U; D) = \sum_{i=1}^c \frac{\sum_{j=1}^n \sum_{k=1}^n u_{ij}^m u_{ik}^m d_{jk}^2}{\sum_{j=1}^n u_{ij}^m} \quad (9.46)$$

Die notwendigen Bedingungen für Extrema liefern

$$u_{ik} = 1 / \sqrt{ \sum_{j=1}^n \frac{\sum_{s=1}^n \frac{u_{is}^m d_{sk}}{\sum_{r=1}^n u_{ir}^m} - \sum_{s=1}^n \sum_{t=1}^n \frac{u_{is}^m u_{it}^m d_{st}}{2 \left( \sum_{r=1}^n u_{ir}^m \right)^2}}{\sum_{s=1}^n \frac{u_{js}^m d_{sk}}{\sum_{r=1}^n u_{jr}^m} - \sum_{s=1}^n \sum_{t=1}^n \frac{u_{js}^m u_{jt}^m d_{st}}{2 \left( \sum_{r=1}^n u_{jr}^m \right)^2}} } \quad (9.47)$$

$$i = 1, \dots, c, k = 1, \dots, n.$$

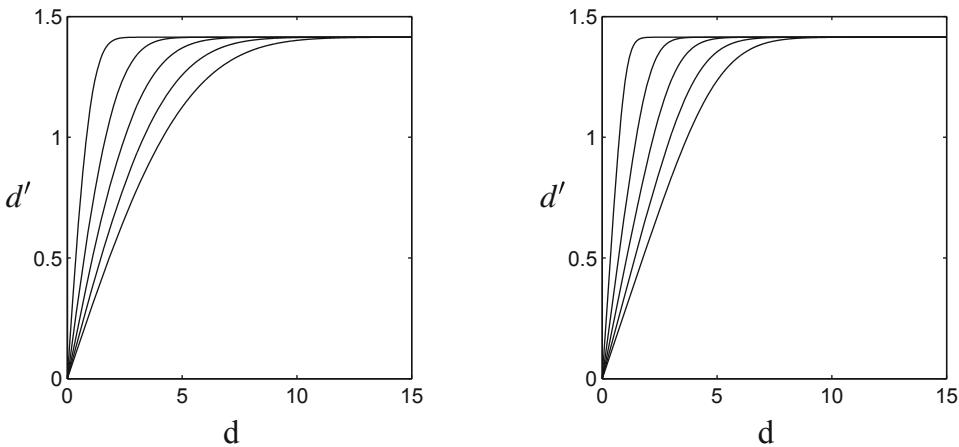
Wir stellen uns vor, dass zur relationalen Datenmatrix  $R$  ein Merkmalsdatensatz  $X \subset \mathbb{R}^p$  existiert, aus dem sich  $R$  unter Verwendung der Euklidischen Norm erzeugen lässt. In diesem Fall nennen wir  $R$  eine *Euklidische Abstandsmatrix*. Die Minimierung von  $J_{FCM}(U, V; X)$  wird dann die gleiche Partition liefern wie die Minimierung von  $J_{RFCM}(U; D)$ . Entsprechendes gilt für die Paare CM/RCM, PCM/RPCM, NC/RNC, CMdd/RCMdd, FCMdd/RFCMdd, PCMdd/RPCMdd und NCMdd/RNCMdd. Ist  $R$  dagegen keine Euklidische Abstandsmatrix, so kann die Minimierung der relationalen Kostenfunktion Zugehörigkeitswerte kleiner null oder größer eins liefern. Im dies zu vermeiden, kann eine nicht-Euklidische Abstandsmatrix  $D$  durch eine sogenannte *Spreizung* in eine Euklidische Abstandsmatrix  $D_\beta$  umgewandelt werden [14].

$$D_\beta = D + \beta \cdot B \quad (9.48)$$

mit einem geeigneten Parameter  $\beta \geq 0$  und der inversen  $n \times n$ -Diagonalmatrix  $B$  mit  $b_{ij} = 1$  für alle  $i, j = 1, \dots, n$ ,  $i \neq j$  und  $b_{ii} = 0$  für alle  $i = 1, \dots, n$ . Der Parameter  $\beta$  wird mit null initialisiert und sukzessive erhöht, bis sich alle Zugehörigkeitswerte im Einheitsintervall befinden. Bei Verwendung dieses Spreizungsverfahrens setzen wir das Präfix NE (nicht-Euklidisch) vor den Namen des relationalen Clustermodells, z. B. *nicht-Euklidische relationale Fuzzy-c-Means (NERFCM)* [14] oder *nicht-Euklidische relationale possibilistische c-Means (NERPCM)* [25].

Die bisher betrachteten relationalen Clustermodelle wurden aus Clustermodellen für Merkmalsdaten mit hypersphärischen Clustern abgeleitet. Im vorigen Abschnitt wurden auch Clustermodelle für Merkmalsdaten mit komplizierteren Clustergeometrien wie Hyperebenen, Ellipsoiden oder Kreisen präsentiert. Solche Modelle lassen sich nur schwierig auf relationale Daten übertragen, ohne explizit Merkmalsdaten zu approximieren. Bei der Supportvektormaschine (Kap. 8) wurde der Kernel-Trick benutzt, um eine Methode für einfache Geometrien (lineare Klassengrenzen) auf kompliziertere Geometrien (nichtlineare Klassengrenzen) anzuwenden. Beim Clustering wird der Kernel-Trick benutzt, um Methoden für einfache Clustergeometrien (hypersphärische Cluster) auf kompliziertere Clustergeometrien anzuwenden. Der Kernel-Trick wurde z. B. auf CM [12, 29, 38], FCM [34, 35, 37], PCM [36], NERFCM [17] und NERPCM [25] angewandt. Die Kernel-Varianten werden mit dem Präfix k bezeichnet, z. B. kCM, kFCM, kPCM, kNERFCM, oder kNERPCM. Für den Euklidischen Abstand liefert der Kernel-Trick die Transformation

$$d_{jk}^2 = \|\varphi(x_j) - \varphi(x_k)\|^2 \quad (9.49)$$



**Abb. 9.10** Kernel-Clustering: Transformation durch Gauß-Kernel (links) und hyperbolischen Tangens-Kernel (rechts)

$$= (\varphi(x_j) - \varphi(x_k)) (\varphi(x_j) - \varphi(x_k))^T \quad (9.50)$$

$$= \varphi(x_j)\varphi(x_j)^T - 2\varphi(x_j)\varphi(x_k)^T + \varphi(x_k)\varphi(x_k)^T \quad (9.51)$$

$$= k(x_j, x_j) - 2 \cdot k(x_j, x_k) + k(x_k, x_k) \quad (9.52)$$

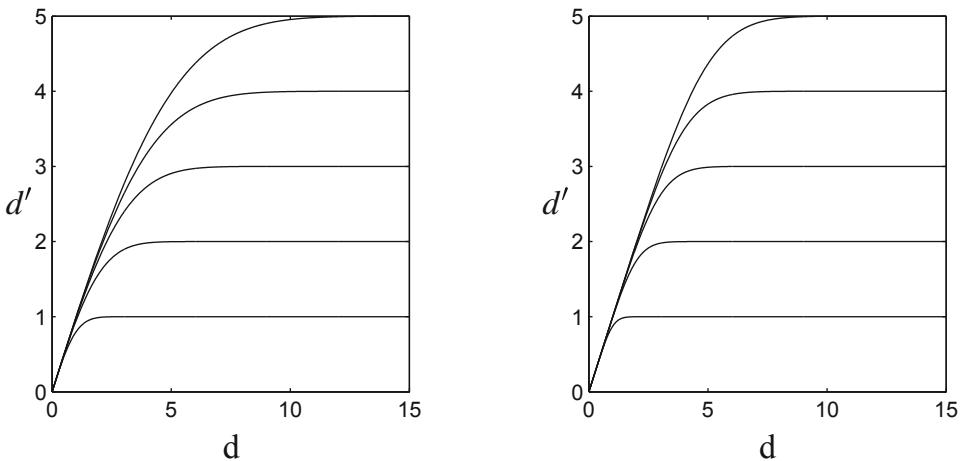
$$= 2 - 2 \cdot k(x_j, x_k) \quad (9.53)$$

wobei angenommen wird, dass  $k(x, x) = 0$  für alle  $x \in \mathbb{R}^p$ . Eine Kernel-Variante eines Clusteralgorithmus kann also einfach dadurch realisiert werden, dass  $D$  zunächst mit (9.53) in  $D'$  transformiert und der Clusteralgorithmus dann auf  $D'$  angewandt wird. Für den Gauß-Kernel und den hyperbolischen Tangens-Kernel erhalten wir die Transformationsvorschriften

$$d'_{jk} = \sqrt{2 - 2 \cdot e^{-\frac{d_{jk}^2}{\sigma^2}}} \quad (9.54)$$

$$d'_{jk} = \sqrt{2 \cdot \tanh\left(\frac{d_{jk}^2}{\sigma^2}\right)} \quad (9.55)$$

Abbildung 9.10 zeigt diese Funktionen für  $\sigma \in \{1, 2, 3, 4\}$ . Die beiden Transformationen sind sehr ähnlich. Kleine Abstände werden näherungsweise linear skaliert und größere Abstände mit  $d' = \sqrt{2}$  begrenzt. Alle der hier beschriebenen relationalen Clustermethoden sind invariant bezüglich einer Skalierung der Abstände mit einem Faktor  $\alpha > 0$ , also erhalten wir für  $D$  und  $D^*$ ,  $d'^*_{jk} = \alpha \cdot d_{jk}$ ,  $j, k = 1, \dots, n$ , die gleichen Ergebnisse. Jede Kurve in Abb. 9.10 kann daher so mit einem Faktor multipliziert werden, dass die Steigung im Ursprung eins beträgt (Abb. 9.11). Der Effekt der Kernel-Transformation ist offensichtlich eine Begrenzung am Abstand  $\sigma$ , so dass der Einfluss von großen Abständen, die z. B. durch Ausreißer verursacht werden, begrenzt wird [24].



**Abb. 9.11** Kernel-Clustering: skalierte Transformation durch Gauß-Kernel (*links*) und hyperbolischen Tangens-Kernel (*rechts*)

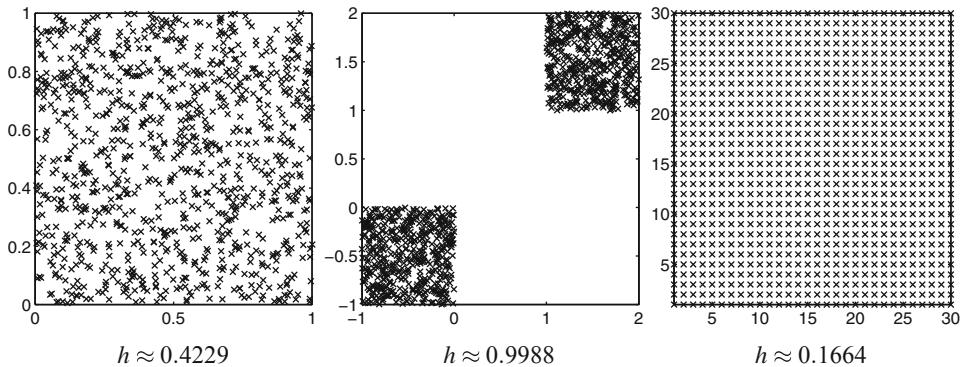
## 9.6 Clustertendenz

Clusteralgorithmen liefern Partitionen, auch wenn die Daten eigentlich gar keine Cluster enthalten. Die Clustertendenz quantifiziert, zu welchem Grad die Daten geclustert sind. Hierzu eignet sich der sogenannte *Hopkins-Index* [18, 19]. Zur Berechnung des Hopkins-Index eines Datensatzes  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$  werden zunächst  $m$  Punkte  $R = \{r_1, \dots, r_m\}$  zufällig aus der konvexen Hülle oder (zur Vereinfachung) aus dem minimalen Hyperwürfel von  $X$  gewählt,  $m \ll n$ . Anschließend werden zufällig  $m$  Datenpunkte  $S = \{s_1, \dots, s_m\}$  aus  $X$  gezogen, so dass  $S \subset X$ . Für die Punkte beider Mengen  $R$  und  $S$  werden die Abstände zum jeweils nächsten Nachbarn aus  $X$  bestimmt. Auf Basis dieser Abstände  $d_{r_1}, \dots, d_{r_m}$  und  $d_{s_1}, \dots, d_{s_m}$  wird der Hopkins-Index  $h \in [0, 1]$  berechnet als

$$h = \frac{\sum_{i=1}^m d_{r_i}^p}{\sum_{i=1}^m d_{r_i}^p + \sum_{i=1}^m d_{s_i}^p} \quad (9.56)$$

Es werden drei Fälle unterschieden

1. Für  $h \approx 0.5$  sind die Abstände innerhalb  $X$  näherungsweise gleich den Abständen zwischen  $R$  und  $X$ , also sind  $R$  und  $X$  ähnlich verteilt. Da  $R$  gleichverteilt ist, ist auch  $X$  gleichverteilt und enthält daher keine Cluster.
2. Für  $h \approx 1$  sind die Abstände innerhalb  $X$  viel kleiner als die Abstände zwischen  $R$  und  $X$ , also besitzt  $X$  eine Clusterstruktur.



**Abb. 9.12** Drei Datensätze und ihre Hopkins-Indizes

3. Für  $h \approx 0$  sind die Abstände zwischen  $R$  und  $X$  viel kleiner als die Abstände innerhalb  $X$ , also haben die Punkte in  $X$  näherungsweise gleiche Abstände zu ihren nächsten Nachbarn und liegen z. B. auf einem regelmäßigen Gitter.

Abbildung 9.12 illustriert diese drei Fälle mit drei unterschiedlichen Datensätzen und ihren Hopkins-Indizes ( $m = 10, n = 1000$ ). Der linke Datensatz ist zufällig verteilt, daher ist  $h \approx 0.5$ . Der mittlere Datensatz enthält zwei deutlich getrennte Cluster, daher ist  $h \approx 1$ . Der rechte Datensatz ist auf einem regelmäßigen orthogonalen Gitter, daher ist  $h$  sehr klein.

## 9.7 Clustervalidität

Bei allen hier beschriebenen Clustermethoden muss die Anzahl  $c \in \{2, \dots, n - 1\}$  der zu bestimmenden Cluster vorab gewählt werden. Oft ist die Anzahl der in den Daten zu erwartenden Cluster vorab nicht bekannt. Die Anzahl der in den Daten enthaltenen Cluster kann mit Hilfe von Clustervaliditätsmaßen geschätzt werden. Dabei wird ein Clusteralgorithmus mehrfach für verschiedene Werte von  $c$  ausgeführt und jeweils die Validität der erhaltenen Partition berechnet. Die Anzahl der Cluster entspricht dann der Partition mit der höchsten Validität. Häufig verwendete Validitätsmaße für Fuzzy-Partitionen sind der *Partitionskoeffizient* (*PC*) [2], der als Summe der quadratischen Zugehörigkeitswerte berechnet wird,

$$PC(U) = \frac{1}{n} \sum_{k=1}^n \sum_{i=1}^c u_{ik}^2 \quad (9.57)$$

und die *Klassifikationsentropie* (*CE*) [33], die als durchschnittliche Entropie der Zugehörigkeitswerte berechnet wird,

$$CE(U) = \frac{1}{n} \sum_{k=1}^n \sum_{i=1}^c -u_{ik} \cdot \log u_{ik} \quad (9.58)$$

**Tab. 9.1** Validitätsmaße für scharfe und indifferenten Partitionen

$U$	$PC(U)$	$CE(U)$
$\begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$	1	0
$\begin{pmatrix} \frac{1}{c} & \dots & \frac{1}{c} \\ \vdots & \ddots & \vdots \\ \frac{1}{c} & \dots & \frac{1}{c} \end{pmatrix}$	$\frac{1}{c}$	$\log c$

mit dem Sonderfall

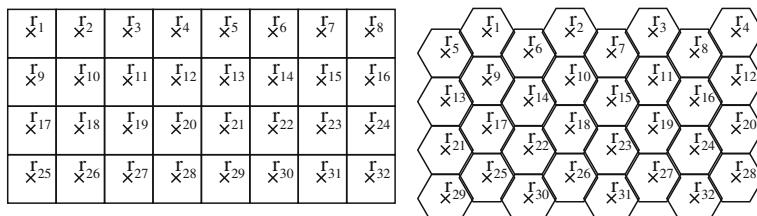
$$-0 \cdot \log 0 = \lim_{u \rightarrow 0} -u \log u = 0 \quad (9.59)$$

Tabelle 9.1 zeigt die Werte der Validitätsmaße PC und CE für eine scharfe und eine indifferente Partition. Für die scharfe Partition liefert PC den Wert eins und CE den Wert null. Eine gute Wahl der Clusteranzahl entspricht also einem hohen Wert von PC (nahe bei eins) oder einem niedrigen Wert von CE (nahe bei null).

Weitere Validitätsmaße, die sich auch für scharfe Cluster verwenden lassen, sind z.B. der Dunn-Index, der Davis-Bouldin-Index, der Calinski-Harabasz-Index, der Gap-Index oder der Silhouette-Index.

## 9.8 Selbstorganisierende Karte

Eine *selbstorganisierende Karte* [20, 21, 32] ist ein heuristisches Projektions- und Clusterverfahren. Die Karte ist eine regelmäßige  $q$ -dimensionale Struktur mit  $l$  Referenzknoten. Für zweidimensionale Karten werden oft rechteckige oder hexagonale Strukturen verwendet (Abb. 9.13). Jeder Knoten besitzt einen festen Ortsvektor  $r_i \in \mathbb{R}^q$ , der die Position auf der Karte festlegt, und einen Referenzvektor  $m_i \in \mathbb{R}^p$ , der sich auf die Daten  $X$  bezieht,  $i = 1, \dots, l$ . In der Trainingsphase werden die Referenzvektoren  $m_i \in \mathbb{R}^p$  zunächst zufällig initialisiert. Dann wird für jeden Datenpunkt  $x_k \in X$  der nächste Referenzvektor  $m_c$  gesucht, und es werden alle Referenzvektoren aktualisiert, die auf der Karte in der



**Abb. 9.13** Rechteckige und hexagonale selbstorganisierende Karte

1. Eingabe: Data  $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^p$ ,  
Kartendimensionalität  $q \in \{1, \dots, p-1\}$ ,  
Knotenpositionen  $R = \{r_1, \dots, r_l\} \subset \mathbb{R}^q$
2. Initialisiere  $M = \{m_1, \dots, m_l\} \subset \mathbb{R}^p$ ,  $t = 1$
3. Für jedes  $x_k$ ,  $k = 1, \dots, n$ ,
  - a. Bestimme Gewinnerknoten  $m_c$  mit
$$\|x_k - m_c\| \leq \|x_k - m_i\| \quad \forall i = 1, \dots, l$$
  - b. Aktualisiere Gewinner und Nachbarn
$$m_i = m_i + h_{ci} \cdot (x_k - m_c) \quad \forall i = 1, \dots, l$$
4.  $t = t + 1$
5. Wiederhole ab (3.), bis Terminierungsbedingung erfüllt
6. Ausgabe: Referenzvektoren  $M = \{m_1, \dots, m_l\} \subset \mathbb{R}^p$

**Abb. 9.14** Trainingsalgorithmus einer selbstorganisierenden Karte

Nachbarschaft von  $r_c$  liegen. Der Grad der Nachbarschaft in der Karte wird oft mit einer sogenannten *Blasenfunktion*

$$h_{ci} = \begin{cases} \alpha(t) & \text{falls } \|r_c - r_i\| < \rho(t) \\ 0 & \text{sonst} \end{cases} \quad (9.60)$$

oder einer Gauß-Funktion

$$h_{ci} = \alpha(t) \cdot e^{-\frac{\|r_c - r_i\|^2}{2\rho^2(t)}} \quad (9.61)$$

berechnet. Der Nachbarschaftsradius  $\rho(t)$  und die Lernrate  $\alpha(t)$  werden während des Lernvorgangs kontinuierlich verringert, z. B. gemäß

$$\alpha(t) = \frac{A}{B+t}, \quad A, B > 0 \quad (9.62)$$

Abbildung 9.14 fasst den Trainingsalgorithmus einer selbstorganisierenden Karte zusammen. Die Aktualisierung der Referenzvektoren erfolgt ähnlich wie bei der lernenden Vektorquantisierung (Kap. 8). Nach Abschluss der Trainingsphase werden die Referenzvektoren in der Karte (oder die Abstände zu ihren Nachbarn) visualisiert. Regionen mit ähnlichen Referenzvektoren entsprechen Clustern in den Daten. Darüber hinaus kann jeder Punkt des Datensatzes auf einen Referenz- und Ortsvektor abgebildet werden.

### Übungsaufgaben

**9.1.** Zeichnen Sie ein Dendrogramm für Single Linkage von  $X = \{-6, -5, 0, 4, 7\}$ .

**9.2.** Zeichnen Sie ein Dendrogramm für Complete Linkage von  $X = \{-6, -5, 0, 4, 7\}$ .

**9.3.** Bestimmen Sie die Sequenz der Clusterzentren von c-Means für  $X = \{-6, -5, 0, 4, 7\}$  mit der Initialisierung  $V = \{5, 6\}$ .

**9.4.** Bestimmen Sie eine Initialisierung  $V$ , bei der c-Means für  $X = \{-6, -5, 0, 4, 7\}$  ein anderes Ergebnis liefert als (c).

**9.5.** Was ist der Unterschied zwischen Klassifikation und Clustering?

---

## Literatur

1. G. B. Ball and D. J. Hall. Isodata, an iterative method of multivariate analysis and pattern classification. In *IFIPS Congress*, 1965.
2. J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.
3. J. C. Bezdek. Fuzzy models—what are they, and why? *IEEE Transactions on Fuzzy Systems*, 1(1):1–6, 1993.
4. J. C. Bezdek and R. J. Hathaway. Optimization of fuzzy clustering criteria using genetic algorithms. In *IEEE Conference on Evolutionary Computation, Orlando*, volume 2, pages 589–594, June 1994.
5. J. C. Bezdek, C. Coray, R. Gunderson, and J. Watson. Detection and characterization of cluster substructure, I. Linear structure: Fuzzy c-lines. *SIAM Journal on Applied Mathematics*, 40(2):339–357, April 1981.
6. J. C. Bezdek, C. Coray, R. Gunderson, and J. Watson. Detection and characterization of cluster substructure, II. Fuzzy c-varieties and convex combinations thereof. *SIAM Journal on Applied Mathematics*, 40(2):358–372, April 1981.
7. J. C. Bezdek, J. M. Keller, R. Krishnapuram, and N. R. Pal. *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Kluwer, Norwell, 1999.
8. R. N. Davé. Fuzzy shell clustering and application to circle detection in digital images. *International Journal on General Systems*, 16:343–355, 1990.
9. R. N. Davé. Characterization and detection of noise in clustering. *Pattern Recognition Letters*, 12:657–664, 1991.
10. W. H. E. Day and H. Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of Classification*, 1(1):7–24, 1984.
11. H. Fang and Y. Saad. Farthest centroids divisive clustering. In *International Conference on Machine Learning and Applications*, pages 232–238, 2008.
12. M. Girolami. Mercer kernel-based clustering in feature space. *IEEE Transactions on Neural Networks*, 13:780–784, 2002.
13. E. E. Gustafson and W. C. Kessel. Fuzzy clustering with a covariance matrix. In *IEEE International Conference on Decision and Control, San Diego*, pages 761–766, 1979.
14. R. J. Hathaway and J. C. Bezdek. NERF c-means: Non-Euclidean relational fuzzy clustering. *Pattern Recognition*, 27:429–437, 1994.
15. R. J. Hathaway and J. C. Bezdek. Optimization of clustering criteria by reformulation. *IEEE Transactions on Fuzzy Systems*, 3(2):241–245, May 1995.
16. R. J. Hathaway, J. W. Davenport, and J. C. Bezdek. Relational duals of the c-means algorithms. *Pattern Recognition*, 22:205–212, 1989.

17. R. J. Hathaway, J. M. Huband, and J. C. Bezdek. Kernelized non-Euclidean relational fuzzy c-means algorithm. In *IEEE International Conference on Fuzzy Systems*, pages 414–419, Reno, May 2005.
18. A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, 1988.
19. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.
20. T. Kohonen. Automatic formation of topological maps of patterns in a self-organizing system. In E. Oja and O. Simula, editors, *Scandinavian Conference on Image Analysis*, pages 214–220, Helsinki, 1981.
21. T. Kohonen. *Self-Organizing Maps*. Springer, Berlin, 2001.
22. R. Krishnapuram and J. M. Keller. A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, 1(2):98–110, May 1993.
23. R. Krishnapuram, A. Joshi, O. Nasraoui, and L. Yi. Low-complexity fuzzy relational clustering algorithms for web mining. *IEEE Transactions on Fuzzy Systems*, 9(4):595–607, August 2001.
24. T. A. Runkler. The effect of kernelization in relational fuzzy clustering. In *GMA/GI Workshop Fuzzy Systems and Computational Intelligence, Dortmund*, pages 48–61, November 2006.
25. T. A. Runkler. Kernelized non-Euclidean relational possibilistic c-means clustering. In *IEEE Three Rivers Workshop on Soft Computing in Industrial Applications*, Passau, August 2007.
26. T. A. Runkler. Relational fuzzy clustering. In J. Valente de Oliveira and W. Pedrycz, editors, *Advances in Fuzzy Clustering and its Applications*, chapter 2, pages 31–52. Wiley, 2007.
27. T. A. Runkler. Wasp swarm optimization of the c-means clustering model. *International Journal of Intelligent Systems*, 23(3):269–285, February 2008.
28. J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.
29. B. Schölkopf, A.J. Smola, and K. R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.
30. P. Sneath and R. Sokal. *Numerical Taxonomy*. Freeman, San Francisco, 1973.
31. J. H. Ward. Hierarchical grouping to optimize an objective function. *Journal of American Statistical Association*, 58(301):236–244, 1963.
32. D. J. Willshaw and C. von der Malsburg. How patterned neural connections can be set up by self-organization. *Proceedings of the Royal Society London*, B194:431–445, 1976.
33. M. P. Windham. Cluster validity for the fuzzy c-means clustering algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4(4):357–363, July 1982.
34. Z.-D. Wu, W.-X. Xie, and J.-P. Yu. Fuzzy c-means clustering algorithm based on kernel method. In *International Conference on Computational Intelligence and Multimedia Applications*, pages 49–54, Xi'an, 2003.
35. D.-Q. Zhang and S.-C. Chen. Fuzzy clustering using kernel method. In *International Conference on Control and Automation*, pages 123–127, 2002.
36. D.-Q. Zhang and S.-C. Chen. Kernel-based fuzzy and possibilistic c-means clustering. In *International Conference on Artificial Neural Networks*, pages 122–125, Istanbul, 2003.
37. D.-Q. Zhang and S.-C. Chen. Clustering incomplete data using kernel-based fuzzy c-means algorithm. *Neural Processing Letters*, 18:155–162, 2003.
38. R. Zhang and A.I. Rudnicki. A large scale clustering scheme for kernel k-means. In *International Conference on Pattern Recognition*, pages 289–292, Quebec, 2002.

---

# Anhang: Optimierungsverfahren

Viele der in diesem Buch beschriebenen Datenanalysemethoden basieren auf Optimierungsverfahren. In diesem Anhang werden die benutzten Verfahren kurz beschrieben: Optimierung mit Ableitungen, Gradientenverfahren und Lagrange-Verfahren. Für eine ausführlichere Betrachtung von Optimierungsverfahren wird auf die Literatur verwiesen, z. B. [1–3].

---

## A.1 Optimierung mit Ableitungen

Extrema oder Sattelpunkte einer differenzierbaren Funktion  $y = f(x)$  können gefunden werden, indem die notwendigen Bedingungen für Extrema von  $f$  zu null gesetzt werden.

$$\frac{\partial f}{\partial x^{(i)}} = 0 \quad (\text{A.1})$$

Ob dort ein Maximum, Minimum oder ein Sattelpunkt vorliegt, kann an der zweiten Ableitung

$$\frac{\partial^2 f}{\partial (x^i)^2} \quad (\text{A.2})$$

abgelesen werden. Sie ist negativ für Maxima, positiv für Minima und null für Sattelpunkte. Als Beispiel bestimmen wir die Extrema der Funktion

$$y = f(x^{(1)}, x^{(2)}) = (x^{(1)} - 1)^2 + (x^{(2)} - 2)^2 \quad (\text{A.3})$$

Die Ableitungen von  $f$  lauten

$$\frac{\partial f}{\partial x^{(1)}} = 2(x^{(1)} - 1), \quad \frac{\partial f}{\partial x^{(2)}} = 2(x^{(2)} - 2) \quad (\text{A.4})$$

$$\frac{\partial^2 f}{\partial (x^{(1)})^2} = 2, \quad \frac{\partial^2 f}{\partial (x^{(2)})^2} = 2 \quad (\text{A.5})$$

Nullsetzen der Ableitung von  $f$  liefert

$$2(x^{(1)} - 1) = 0 \quad \Rightarrow \quad x^{(1)} = 1, \quad 2(x^{(2)} - 2) = 0 \quad \Rightarrow \quad x^{(2)} = 2 \quad (\text{A.6})$$

Da die zweiten Ableitungen positiv sind, hat  $f$  ein Minimum bei  $x = (1, 2)$ .

## A.2 Gradientenverfahren

Das Gradientenverfahren ist eine iterative approximative Optimierungsmethode für differenzierbare Funktionen. Es wird häufig angewendet, wenn Nullsetzen der Ableitungen zu keinen explizit lösbarer Gleichungen führt. Das Gradientenverfahren minimiert eine Funktion  $y = f(x)$ , indem der Parametervektor  $x = x_0$  zufällig initialisiert und dann iterativ für  $k = 1, 2, \dots$  gemäß folgender Vorschrift aktualisiert wird:

$$x_k^{(i)} = x_{k-1}^{(i)} - \alpha \left. \frac{\partial f}{\partial x^{(i)}} \right|_{x=x_{k-1}} \quad (\text{A.7})$$

Die Schrittweite  $\alpha > 0$  muss geeignet gewählt werden. Für unser Beispiel (A.3) initialisieren wir  $x = (0, 0)$ , verwenden die Ableitungen (A.4) und erhalten mit  $\alpha = 1/2$

$$x_1^{(1)} = 0 - \frac{1}{2} \cdot 2 \cdot (0 - 1) = 1, \quad x_1^{(2)} = 0 - \frac{1}{2} \cdot 2 \cdot (0 - 2) = 2 \quad (\text{A.8})$$

$$x_2^{(1)} = 1 - \frac{1}{2} \cdot 2 \cdot (1 - 1) = 1, \quad x_2^{(2)} = 2 - \frac{1}{2} \cdot 2 \cdot (2 - 2) = 2 \quad (\text{A.9})$$

In diesem Fall findet das Gradientenverfahren also in einem einzigen Schritt das Minimum  $x = (1, 2)$ . Für das gleiche Beispiel mit  $\alpha = 1/4$  erhalten wir

$$x_1^{(1)} = 0 - \frac{1}{4} \cdot 2 \cdot (0 - 1) = \frac{1}{2}, \quad x_1^{(2)} = 0 - \frac{1}{4} \cdot 2 \cdot (0 - 2) = 1 \quad (\text{A.10})$$

$$x_2^{(1)} = \frac{1}{2} - \frac{1}{4} \cdot 2 \cdot \left(\frac{1}{2} - 1\right) = \frac{3}{4}, \quad x_2^{(2)} = 1 - \frac{1}{4} \cdot 2 \cdot (1 - 2) = \frac{3}{2} \quad (\text{A.11})$$

$$x_3^{(1)} = \frac{3}{4} - \frac{1}{4} \cdot 2 \cdot \left(\frac{3}{4} - 1\right) = \frac{7}{8}, \quad x_3^{(2)} = \frac{3}{2} - \frac{1}{4} \cdot 2 \cdot \left(\frac{3}{2} - 2\right) = \frac{7}{4} \quad (\text{A.12})$$

Der Algorithmus approximiert das Minimum also mit immer kleinerem Fehler, der erst nach unendlich vielen Schritten zu null konvergiert. Für  $\alpha = 1$  erhalten wir

$$x_1^{(1)} = 0 - 2 \cdot (0 - 1) = 2, \quad x_1^{(2)} = 0 - 2 \cdot (0 - 2) = 4 \quad (\text{A.13})$$

$$x_2^{(1)} = 2 - 2 \cdot (2 - 1) = 0, \quad x_2^{(2)} = 4 - 2 \cdot (4 - 2) = 0 \quad (\text{A.14})$$

Der Algorithmus alterniert also endlos zwischen  $(2, 4)$  und  $(0, 0)$ . Für  $\alpha = 2$  erhalten wir schließlich

$$x_1^{(1)} = 0 - 2 \cdot 2 \cdot (0 - 1) = 4, \quad x_1^{(2)} = 0 - 2 \cdot 2 \cdot (0 - 2) = 8 \quad (\text{A.15})$$

$$x_2^{(1)} = 4 - 2 \cdot 2 \cdot (4 - 1) = -8, \quad x_2^{(2)} = 8 - 2 \cdot 2 \cdot (8 - 2) = -16 \quad (\text{A.16})$$

$$x_3^{(1)} = -8 - 2 \cdot 2 \cdot (-8 - 1) = 28, \quad x_3^{(2)} = -16 - 2 \cdot 2 \cdot (-16 - 2) = 56 \quad (\text{A.17})$$

Der Algorithmus divergiert also zu  $(\pm\infty, \pm\infty)$ .

Die Wahl der Schrittweite  $\alpha$  kann vermieden werden, wenn auch höhere Ableitungen berücksichtigt werden, wie z. B. beim *Newton-Verfahren*, das für quadratische Funktionen das Extremum stets in einem einzigen Schritt findet.

$$x_k^{(i)} = x_{k-1}^{(i)} - \frac{\frac{\partial f}{\partial x^{(i)}} \Big|_{x=x_{k-1}}}{\frac{\partial^2 f}{\partial(x^{(i)})^2} \Big|_{x=x_{k-1}}} \quad (\text{A.18})$$

Eine Methode zur die Minimierung linearer Funktionen unter Randbedingungen ist der *Simplex- oder Nelder-Mead-Algorithmus*. *Sequentielle Programmierung* löst eine Optimierungsaufgabe, indem in jedem Schritt eine lokale Approximation der zu optimierenden Funktion betrachtet wird, z. B. eine lineare Approximation bei der *sequentiellen linearen Programmierung* oder z. B. eine Approximation durch quadratische Funktionen bei der *sequentiellen quadratischen Programmierung*.

### A.3 Lagrange-Optimierung

Wir betrachten das Problem, eine Funktion  $y = f(x)$  unter der Randbedingung  $g(x) = 0$  zu minimieren. Z. B. kann die Randbedingung  $x^{(1)} = x^{(2)}$  durch die Funktion  $g(x^{(1)}, x^{(2)}) = x^{(1)} - x^{(2)}$  repräsentiert werden. In manchen Fällen kann eine Minimierungsaufgabe ohne Randbedingungen dadurch gelöst werden, dass die Randbedingung in die zu optimierende Funktion eingesetzt und anschließend eine Optimierung ohne Randbedingungen durchgeführt wird. Um beispielsweise (A.3) unter der Randbedingung  $x^{(1)} = x^{(2)}$  zu minimieren, können wir  $x^{(2)}$  durch  $x^{(1)}$  ersetzen und erhalten

$$y = f(x^{(1)}) = (x^{(1)} - 1)^2 + (x^{(1)} - 2)^2 = 2(x^{(1)})^2 - 6x^{(1)} + 5 \quad (\text{A.19})$$

mit der ersten und zweiten Ableitung

$$\frac{\partial f}{\partial x^{(1)}} = 4x^{(1)} - 6, \quad \frac{\partial^2 f}{\partial(x^{(1)})^2} = 4 \quad (\text{A.20})$$

Nullsetzen der ersten Ableitung liefert

$$4x^{(1)} - 6 = 0 \quad \Rightarrow \quad x^{(1)} = \frac{3}{2} \quad (\text{A.21})$$

Das Minimum von  $f$  unter der Randbedingung  $x^{(1)} = x^{(2)}$  ist also bei  $x = (\frac{3}{2}, \frac{3}{2})$ . In vielen Fällen führt ein solches Einsetzungsverfahren jedoch zu keiner explizit lösbarer Funktion. Stattdessen kann dann die *Lagrange-Optimierung* verwendet werden. Bei der Lagrange-Optimierung wird ein Minimierungsproblem unter Randbedingungen in ein Minimierungsproblem ohne Randbedingungen transformiert. Die Minimierung von  $y = f(x)$  unter der Randbedingung  $g(x) = 0$  entspricht der Minimierung der sogenannten Lagrange-Funktion

$$L(x, \lambda) = f(x) - \lambda g(x) \quad (\text{A.22})$$

ohne Randbedingungen mit dem zusätzlichen Parameter  $\lambda \in \mathbb{R}$ . In unserem Beispiel (A.3) mit  $g(x^{(1)}, x^{(2)}) = x^{(1)} - x^{(2)}$  erhalten wir

$$L(x^{(1)}, x^{(2)}, \lambda) = (x^{(1)} - 1)^2 + (x^{(2)} - 2)^2 - \lambda(x^{(1)} - x^{(2)}) \quad (\text{A.23})$$

mit den Ableitungen

$$\frac{\partial L}{\partial x^{(1)}} = 2(x^{(1)} - 1) - \lambda, \quad \frac{\partial L}{\partial x^{(2)}} = 2(x^{(2)} - 2) + \lambda, \quad \frac{\partial L}{\partial \lambda} = -(x^{(1)} - x^{(2)}) \quad (\text{A.24})$$

Nullsetzen der Ableitungen liefert

$$x^{(1)} = 1 + \frac{\lambda}{2}, \quad x^{(2)} = 2 - \frac{\lambda}{2}, \quad x^{(1)} = x^{(2)} \quad (\text{A.25})$$

und weiter  $x = (\frac{3}{2}, \frac{3}{2})$ . Für mehrfache Randbedingungen  $g_1(x), \dots, g_r(x)$  lautet die Lagrange-Funktion

$$L(x, \lambda) = f(x) - \sum_{i=1}^r \lambda_i g_i(x) \quad (\text{A.26})$$

mit den Parametern  $\lambda_1, \dots, \lambda_r \in \mathbb{R}$ . Optimierung unter Ungleichheits-Randbedingungen wie z. B.  $g(x) \geq 0$  führt oft zu deutlich schwierigeren Problemen. Ein Ansatz für diesen Fall liefert die *Karush-Kuhn-Tucker-Theorie* (KKT) [2].

## Literatur

1. R. Fletcher. Practical Methods of Optimization. Wiley, Chichester, 2000.
2. J. Nocedal and S. J. Wright. Numerical Optimization. Springer, New York, 2006.
3. R. K. Sundaram. A First Course in Optimization Theory. Oxford University Press, Oxford, 1996.

---

# Lösungen der Übungsaufgaben

---

## Übungsaufgaben aus Kap. 2

- 1    **2.1** Kuchen: nominal, 180: Intervall, 45: proportional
- 2    **2.2** (a) 4, (b) 3, (c) 2.8
- 3    **2.3** (a) 4, (b) 16, (c) 16, (d) 2
- 4    **2.4** (2.42): Ähnlichkeit, (2.43): Ähnlichkeit, (2.44): Unähnlichkeit, (2.45): weder noch

---

## Übungsaufgaben aus Kap. 3

- 6    **3.1** kein Rauschen, ein globaler Ausreißer, kein lokaler Ausreißer
- 7    **3.2** (a) globaler Ausreißer 4, (b) lokaler Ausreißer mittlere 1, (c) globaler Ausreißer (2,1)
- 8    **3.3** (a)  $(0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0)$ , (b)  $(0, 0, 0, 0, 0)$ , (c)  $(0, 0, 0, 0, \frac{1}{2}, \frac{1}{4}, \frac{1}{8})$
- 9    **3.4** Hyperwürfel:  $\{(0, 0), (1, 1)\}$ ,  $\mu-\sigma$ :  $\{(-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}), (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})\}$

---

## Übungsaufgaben aus Kap. 4

- 11    **4.1**  $(\frac{\sqrt{2}}{2}, 0, \frac{\sqrt{2}}{2}), (-\frac{\sqrt{2}}{2}, 0, \frac{\sqrt{2}}{2})$
- 12    **4.2**  $Y = \{(-2\sqrt{2}, 0), (0, -\sqrt{2}), (0, \sqrt{2}), (2\sqrt{2}, 0)\}$ ,  $X' = X, 0$
- 13    **4.3**  $Y = \{-2\sqrt{2}, 0, 0, 2\sqrt{2}\}$ ,  $X' = \{(-3, -1, -1), (-1, -1, 1), (-1, -1, 1), (1, -1, 3)\}, 1$

**Übungsaufgaben aus Kap. 5**

15    **5.1**  $C = \begin{pmatrix} 6 & 0 \\ 0 & \frac{2}{7} \end{pmatrix}, S = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

16    **5.2** 0

17    **5.3** 8

**Übungsaufgaben aus Kap. 6**

18    **6.1**  $C = \begin{pmatrix} 2.5 & 2.5 \\ 2.5 & 2.5 \end{pmatrix}, x^{(1)} = x^{(2)}$

19    **6.2**  $C = \begin{pmatrix} 2.5 & 0 \\ 0 & 2.5 \end{pmatrix}, x^{(1)} = 3, x^{(2)} = 2$

**Übungsaufgaben aus Kap. 7**

20    **7.1** (1, 2; 3), (2, 3; 5), (3, 5; 8)

21    **7.2**  $y_k = y_{k-1} + y_{k-2}$

22    **7.3** 13, 21, 34, ...

**Übungsaufgaben aus Kap. 8**

23    **8.1**  $p(1 \mid (0, 0)) = \frac{2}{3}, p(2 \mid (0, 0)) = \frac{1}{3}$

24    **8.2**  $x \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \frac{1}{2} = 0$

25    **8.3**  $f(x) = \begin{cases} 1 & \text{falls } x^{(1)} < 0.5 \text{ und } x^{(2)} < 0.5 \\ 2 & \text{falls } x^{(1)} > 0.5 \text{ oder } x^{(2)} > 0.5 \\ \text{undefiniert} & \text{sonst} \end{cases}$

26    **8.4**  $f(x) = 2$

27    **8.5** naiver Bayes-Klassifikator, SVM, NN: TPR=1, FPR=0; 3-NN: TPR=1, FPR=1

---

**Übungsaufgaben aus Kap. 9**28    **9.1**  $\{\{-6, 5\}, \{0, \{4, 7\}\}\}$ 29    **9.2**  $\{\{-6, 5\}, 0\}, \{4, 7\}\}$ 30    **9.3**  $V = \left\{-\frac{7}{4}, 7\right\}, V = \left\{-\frac{11}{3}, \frac{11}{2}\right\}$ 31    **9.4** z.B.  $V = \left\{-\frac{11}{2}, \frac{11}{3}\right\}$

---

# Sachverzeichnis

- 2-Sigma-Regel, 25
- A**
- Abstand, 12
    - der Zentren, 112
    - maximaler, 112
    - minimaler, 112
    - mittlerer, 112
  - Abtastung, 18
  - Ähnlichkeit, 12
    - Dice, 15
    - Jaccard, 15
    - Kosinus, 15
    - Tanimoto, 15
    - Überlapp, 15
  - Ähnlichkeitsmaß, 14
  - Algorithmus, evolutionärer, 81
  - Amplitudenspektrum, 56
  - AO, 115
  - Approximator, universeller, 72
  - AR-Modell, 87
  - ARIMA-Modell, 87
  - ARMA-Modell, 87
  - ARMAX-Modell, 87
  - Ausfallrate, 91
  - Ausreißer, 24, 124
    - globaler, 25
    - lokaler, 25
  - Ausreißerclustering, 119
    - mit Medoiden, 122
    - relationales, 122
  - Auto-Assoziator, 51
  - Average Linkage, 112
- B**
- Bandbegrenzung, 19
  - Basisfunktion, radiale, 76
- Kernel, 100
- Baum, Entscheidungsbaum, 102
- Bayes-Klassifikator, 93
  - naiver, 93
- Bilddaten, 2
- Biologie, 2
- Butterworth-Filter, 31
- C**
- c-Elliptotypes, 120
  - c-Lines, 120
  - c-Means, 114
    - Fuzzy, 117
    - Kernel, 123
    - nicht-Euklidische relationale
      - possibilistische, 123
    - Kernel, 123
    - possibilistische, 118
    - Kernel, 123
    - relationale, 122
  - c-Medoide, 122
    - possibilistische, 122
    - possibilistische, 122
  - c-Shells, 121
  - c-Varieties, 120
  - C4.5, 106
  - C5.0, 106
  - CART, 105
  - Cauchy-Funktion, 118
  - CE, 126
  - CHAID, 106
  - Chi-Quadrat-Test, 63, 106
  - City-Block-Norm, 14
  - Clusteranalyse, sequenzielle divisive
    - hierarchische nichtüberlappende, 113
  - Clustering, 108

- Ausreißerclustering, 119  
  relationales, 122  
c-Means, 114, 118  
  possibilistische  
  relationale, 122  
DBSCAN, 113  
Fuzzy, 116  
Fuzzy-c  
  Elliptotypes, 120  
  Lines, 120  
  Means, 117  
  Shells, 121  
  Varieties, 120  
GK, 119  
Gustafson-Kessel, 119  
Kernel, 123  
nicht-Euklidisches  
  relationale Fuzzy-c-Means, 123  
  relationale possibilistische c-Means, 123  
Noise Clustering, 119  
prototypbasiertes, 114  
relationales, 121  
  c-Medoide, 122  
  Fuzzy-c-Means, 122  
  Fuzzy-c-Medoide, 122  
  possibilistische c-Means, 122  
  possibilistische c-Medoide, 122  
SAHN, 111  
SDHN, 113  
sequenzielles, 111  
Tendenz, 125  
Validität, 126  
Ward-Methode, 112  
Clustertendenz, 125  
Clustervalidität, 126  
CM, 114  
  Kernel, 123  
Complete Linkage, 112  
CRISP-DM, 3
- D**
- Data Analytics, 2  
Data Mining, 2  
Daten  
  Bild, 2  
  Biomedizin, 2  
  Fehler, 23  
  Geschäftsdaten, 1  
  industrielle, 1
- Iris, 5  
markierte, 89  
Matrix, 10  
Menge, 10  
relationale, 11  
Skalen, 6  
strukturierte, 2  
Text, 2  
Trainingsdaten, 79  
ungültige, 25  
unmarkierte, 109  
Validierungsdaten, 79  
Video, 2
- Datenbank, 2  
Datenfilterung, 23  
Datensäuberung, 23  
Datenvizualisierung, 37  
Datenvorverarbeitung, 23  
DBSCAN, 113  
Deltaregel, 75  
Dendrogramm, 112  
Diaconis, 53  
Diagonalmorm, 13  
Diagramm, 37  
Dice-Ähnlichkeit, 15  
Diskriminanzanalyse, lineare, 96  
Distanzmaß, 12  
Drift, 24  
Drittvariable, 62
- E**
- E-Mail, 2  
Edit-Abstand, 16  
Eigendekomposition, 43  
Eigenvektorprojektion, 39  
Empfindlichkeit, 91  
Entropie, 103  
Entscheidungsbaum, 102  
ETL-Prozess, 36  
Euklidische Norm, 13, 14  
Exot, 25
- F**
- F-Maß, 91  
Falsch negativ, 90  
Falsch positiv, 90  
Falsch-Negativ-Rate, 91  
Falsch-Positiv-Rate, 91  
Falschklassifikationsrate, 91

- negative, 91  
FCE, 120  
FCL, 120  
FCM, 117  
Kernel, 123  
FCS, 121  
FCV, 120  
Fehler  
Arten, 23  
erster Ordnung, 91  
systematischer, 23  
Trainingsfehler, 79  
Validierungsfehler, 79  
zufälliger, 23  
zweiter Ordnung, 91  
Filter, 23, 27  
diskrete lineare 30  
exponentielle 28  
Medianfilter, 28  
Mittelwertfilter, 27  
FIR, 30  
Fisher, 97  
Fisher-Z-Transformation, 35  
FLVQ, 101  
Fourier, 54  
Spektrum, 19  
Transformation, 55  
Freedman, 53  
Frobenius-Norm, 13  
Funktion  
logistische, 73  
sigmoide, 73  
Fuzzy LVQ, 101  
Fuzzy-c  
Elliptotypes, 120  
Lines, 120  
Means, 117  
Kernel, 123  
nicht-Euklidische relationale, 123  
Medoide, 122  
Shells, 121  
Varieties, 120  
Clustering, 116
- G**  
Gauß-Kernel, 100, 124  
Gaußsches Rauschen, 23  
Genauigkeit, 91  
Genauigkeit-Trefferquote-Diagramm, 93
- Genaugkeit-Trefferquote-Grenzwert, 93  
Geschäftsdaten, 1  
Geschäftsprozess, 1  
GK, 119  
Gradientenverfahren, 132  
Grenzwertoptimierungskurve, 92  
Gustafson-Kessel, 119
- H**  
Hamming-Abstand, 14  
Hauptkomponentenanalyse, 39  
Hilbert-Schmidt-Norm, 13  
Histogramm, 51, 63  
Fuzzy, 53  
unscharfes, 53  
Hopkins-Index, 125  
Hotelling-Transformation, 39  
Huber-Funktion, 72  
Hülle, konvexe, 125  
Hyperwürfel, 34
- I**  
ID3-Algorithmus, 105  
IIR, 30  
Impulsantwort  
endliche, 30  
unendliche, 30  
Infimum-Norm, 14  
Interpolation, 26  
Intervallskala, 9  
Iris, 5  
Iterative Dichotomiser, 105
- J**  
Jaccard-Ähnlichkeit, 15
- K**  
Kalibrierung, 24  
Karhunen-Loëve-Transformation, 39  
Karte, selbstorganisierende, 127  
Kausalität, 61  
kCM, 123  
KDD, 3  
Kernel  
Gauß, 100, 124  
linearer, 99  
polynomieller, 99  
RBF, 100  
Tangens Hyperbolicus, 100, 124  
Trick, 99, 123

- kFCM, 123  
Klasse, 6  
Klassifikation, 88  
    falsche, 91  
    irrelevante, 91  
    negative, 91  
    positive, 91  
    relevante, 91  
    richtige, 91  
Klassifikationsentropie, 126  
Klassifikator  
    Bayes, 93  
    C4.5, 106  
    C5.0, 106  
    CART, 105  
    CHAID, 106  
    Entscheidungsbaum, 102  
    Entwurf, 90  
    hierarchischer, 102  
    lernende Vektorquantisierung, 101  
    lineare Diskriminanz, 96  
    nächster Nachbar, 100  
    Supportvektormaschine, 98  
    unscharfe lernende  
        Vektorquantisierung, 101  
kNERFCM, 123  
kNERPCM, 123  
KNIME, 3  
Knowledge Discovery, 2  
Korrektklassifikationsrate, 91  
Korrelation, 59  
    bedingte, 62  
    bipartille, 62  
    lineare, 59  
    multiple, 62  
    nichtlineare, 63  
    partielle, 62  
    Pearson, 60  
    Scheinkorrelation, 62  
Kosinus-Ähnlichkeit, 15  
Kovarianz, 59  
kPCM, 123  
Kreuzvalidierung, 78  
Kundensegmentierung, 2
- L**  
Lagrange-Optimierung, 133  
Lazy Learning, 101  
Least Trimmed Squares, 72
- Leave-One-Out, 80  
Lebesgue-Norm, 13  
Lernen  
    faules, 101  
    überwachtes, 89  
    unüberwachtes, 109  
Levenshtein-Abstand, 16  
Linkage  
    Average, 112  
    Complete, 112  
    Single, 112  
LVQ, 101  
    Fuzzy, 101  
    unscharfe, 101
- M**  
Mahalanobis-Norm, 13, 119  
Manhattan-Norm, 14  
Maß  
    Distanz, 12  
    Unähnlichkeit, 12  
    Ähnlichkeit, 14  
Maßskala, 6  
MATLAB, 3  
Matrixdarstellung, 10  
Matrixdekomposition, 43  
Matrixnorm, 13  
Maximalabstand, 112  
MDS, 43  
Mealy-Maschine, 84  
Median, 9  
    gleitender, 28  
Medien, soziale, 2  
Medizin, 2  
Medoid, 122  
Mengendarstellung, 10  
Mercer, Satz von, 99  
Merkmals, 6  
    konstantes, 25  
Merkmalsselektion, 80  
Merkmalsvektor, 10  
Messbereich, 25  
Messfehler, 23  
Minimalabstand, 112  
Minkowski-Norm, 13  
Mittelwert, 9  
    geometrischer, 9  
    gleitender, 27  
    harmonischer, 9

- quadratischer, 9  
verallgemeinert, 9, 14
- MLP, 73
- Modalwert, 6
- Modell
- autoregressives, 86
  - lineares autoregressives, 87
  - rekurrentes, 85
- Modus, 6
- Moore-Maschine, 84
- Muster, 2
- interessante, Definition, 2
- N**
- Nachbar, nächster, 26, 100, 125  
Klassifikator, 100
- NaN, 26
- NC, 119
- Nelder-Mead-Algorithmus, 133
- NERFCM, 123  
Kernel, 123
- NERPCM, 123  
Kernel, 123
- Netz, neuronales, 72
- Neuron, 73
- Newton-Verfahren, 133
- Noise Clustering, 119
- Nominalskala, 6
- Norm, 12
- City-Block-, 14
  - Diagonal-, 13
  - Euklidische, 13, 14
  - Frobenius, 13
  - Hilbert-Schmidt, 13
  - hyperbolische, 12
  - Infimum, 14
  - Lebesgue, 13
  - Mahalanobis, 13, 119
  - Manhattan, 14
  - Matrix, 13
  - Minkowski, 13
  - Supremum, 14
- Not a number, 26
- Nyquist-Bedingung, 19
- O**
- Objekt, 6
- Ontologie, 2
- Optimierung
- alternierende, 115
- Gradientenverfahren, 132
- Lagrange, 133
- mit Ableitungen, 131
- Nelder-Mead-Algorithmus, 133
- Newton-Verfahren, 133
- sequenzielle Programmierung, 133
- lineare, 133
  - quadratische, 133
- Simplex, 133
- Ordinalskala, 6
- Ordnung, totale, 8
- P**
- Partition, 109
- Partitionskoeffizient, 126
- Partitionsmatrix, 114
- PC, 126
- PCA, 39
- PCM, 118  
Kernel, 123
- Pearson-Korrelation, 60
- Perzeptron, 73  
mehrschichtiges, 73
- Phasenspektrum, 56
- PR-Diagramm, 93
- Precision, 91
- Precision Recall
- Breakeven Point, 93
  - Diagramm, 93
- Prognose, 83
- Programmierung, sequenzielle, 133
- lineare, 133
  - quadratische, 133
- Projektion, achsenparallele, 39
- Proportionalskala, 9
- Prototyp, 114
- Prozessdaten, 1
- industrielle, 1
- Pruning, 106
- Pseudoinverse, 70
- Q**
- QlikView, 3
- Quantisierung, 18
- Quinlan-Algorithmus, 105
- R**
- R, 3
- Rapid Miner, 3

- Rauschen, 23  
 additives, 23  
 Filterung, 27  
 RBF-Kernel, 100  
 RCM, 122  
 RCMdd, 122  
 Recall, 91  
 Receiver Operating Characteristic, 92  
 Regression, 66  
 lineare, 67  
 nichtlineare Substitution, 71  
 polynomielle, 71  
 robuste, 72  
 Relation, 11  
 Ähnlichkeit, 14  
 Distanz, 12  
 Sequenzen, 16  
 Unähnlichkeit, 12  
 Relevanz, 91  
 RFCM, 122  
 Kernel, 123  
 RFCMdd, 122  
 Richtig negativ, 90  
 Richtig positiv, 90  
 Richtig-Negativ-Rate, 91  
 Richtig-Positiv-Rate, 91  
 RNC, 122  
 RNCMdd, 122  
 ROC-Kurve, 92  
 RPCM, 122  
 Kernel, 123  
 RPCMdd, 122  
 Rundung, 20  
 kaufmännische, 20
- S**  
 SAHN, 111  
 Sammon-Abbildung, 47  
 SAS, 3  
 Satz  
   von Fourier, 54  
   von Mercer, 99  
   von Shannon, 19  
 Säuberung, 23  
 Scheinkorrelation, 62  
 Schwarmintelligenz, 81  
 Scott, 53  
 SDHN, 113  
 Segreganz, 91
- Semantik, 2  
 Sensitivität, 91  
 Sequenzrelation, 16  
 Shannon, 19  
 Shepard-Diagramm, 45  
 Sigma-Regel, 25  
 Signal, 18  
 Simplex, 133  
 Simulated Annealing, 81  
 Single Linkage, 112  
 Singulärwertzerlegung, 39  
 Skala, 6  
 Skalierung, 24  
   mehrdimensionale, 43  
 Software, 3  
 Spektralanalyse, 54  
 Spektrum, 19, 56  
 Spezifität, 91  
 Spotfire, 3  
 Spreizung, 123  
 SPSS, 3  
 Standardabweichung, 25  
 Standardisierung, 34  
 STATISTICA, 3  
 Störfaktor, 62  
 Streudiagramm, 38  
 Sturgess, 53  
 Substitution, nichtlineare, 71  
 Supportvektormaschine, 98  
 Supremum-Norm, 14  
 SVD, 39  
 SVM, 98
- T**  
 Tableau, 3  
 Tangens, 35  
   hyperbolischer, 35, 74  
   Kernel, 100, 124  
 tanh, 35  
   Kernel, 100, 124  
 Tanimoto-Ähnlichkeit, 15  
 Tendenz (Clustering), 125  
 Textdaten, 2  
 Textdokument, 2  
 TIBCO, 3  
 Tool, 3  
 Torgerson, 46  
 Trainingsdaten, 79  
 Trainingsfehler, 79

- Trefferquote, 91  
Trennfähigkeit, 91  
True Negative Rate, 91  
True positive, 90  
True Positive Rate, 91
- U**  
Überlapp-Ähnlichkeit, 15  
Übertragungsfehler, 23  
Unähnlichkeit, 12  
Unähnlichkeitsmaß, 12  
Unschärfe, 117
- V**  
Validierungsdaten, 79  
Validierungsfehler, 79  
Validität (Clustering), 126  
Vektorquantisierung, 101  
  unscharfe, 101  
Videodaten, 2  
Visualisierung, 37  
Vorhersagewert, 91  
  negativer, 91
- positiver, 91  
Voronoi-Diagramm, 102  
Vorverarbeitung, 23
- W**  
Ward-Methode, 112  
Warenkorbanalyse, 2  
Webdokument, 2  
WEKA, 3  
Wert  
  analoger, 20  
  prädiktiver, 91  
  negativer, 91  
  positiver, 91  
Wirksamkeit, 91  
Wissen, Definition, 2
- Z**  
Zeitreihe  
  Filterung, 27  
Zugehörigkeitsfunktion, 53  
Zustandsautomat, 83  
  endlicher, 83