

Paul Wilmott

# WIE FUNKTIONIERT EIGENTLICH KI?

DIE **MATHEMATIK** DER  
KÜNSTLICHEN INTELLIGENZ  
SUPER EINFACH ERKLÄRT

ohne  
Vorkenntnisse



Rheinwerk  
Computing

# Impressum

Dieses E-Book ist ein Verlagsprodukt, an dem viele mitgewirkt haben, insbesondere:

**Lektorat** Lisa Helmus

**Korrektorat** Friederike Daenecke, Zülpich

**Covergestaltung** Lisa Kirsch

**Coverbild** Shutterstock: 2401195433©Jayhermiony

**Herstellung E-Book** Arkin Keskin

**Satz E-Book** Rheinwerk Verlag

Bibliografische Information der Deutschen Nationalbibliothek:

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

**ISBN 978-3-367-10483-3**

1. Auflage 2025

© Rheinwerk Verlag GmbH, Bonn 2025

# Liebe Leserin, lieber Leser,

die Künstliche Intelligenz ist spätestens in Form von Large Language Models wie ChatGPT nicht mehr aus unserem Alltag wegzudenken. Dabei sind wir jedoch beim täglichen Gebrauch weit davon entfernt, wirklich zu verstehen, wie diese Modelle funktionieren. Wie findet ChatGPT zu seinen Antworten? Und warum liegt es manchmal so katastrophal daneben? Der Schlüssel zu Antworten auf diese Fragen liegt in den mathematischen Prinzipien, auf denen die KI aufgebaut ist.

Paul Wilmott erklärt unterhaltsam und zugänglich, wie die Mathematik hinter der KI funktioniert und wie Maschinen »denken«. Dabei ermöglicht er uns einen Einblick in die Geschichte der Entwicklung von KI-Modellen, liefert Denkanstöße für die Zukunft und äußert Bedenken.

Anschließend geht es Schritt für Schritt ans Eingemachte: wie ein Neuronales Netz aufgebaut ist, wie das Prinzip der Nächsten Nachbarn funktioniert und wie ein Cluster aus Datenpunkten entsteht. Um die Vorgänge hinter den Kulissen Künstlicher Intelligenz zu verstehen, musst du kein Mathematikgenie sein – Wilmott vermittelt die grundlegenden Prinzipien auch an Leser\*innen ohne

überbordende Begeisterung für mathematische Themen (siehe dazu auch den Abschnitt »Sorry, das hier ist ein Mathematikbuch!«).

Zum Abschluss auch noch ein Wort in eigener Sache. Wir im Rheinwerk Verlag möchten uns und unsere Bücher natürlich stetig verbessern. Dazu freuen wir uns über dein Feedback! Wie hat dir das Buch gefallen? Können wir noch etwas verbessern? Melde dich gerne jederzeit mit Feedback und Fragen bei uns.

Und jetzt wünsche ich dir viel Spaß mit diesem Mathematikbuch der etwas anderen Art!

**Lisa Helmus**

Lektorat Rheinwerk Computing

*[lisa.helmus@rheinwerk-verlag.de](mailto:lisa.helmus@rheinwerk-verlag.de)*

*[www.rheinwerk-verlag.de](http://www.rheinwerk-verlag.de)*

Rheinwerk Verlag • Rheinwerkallee 4 • 53227 Bonn



# Inhaltsverzeichnis

Aus dem Lektorat  
Inhaltsverzeichnis

Vorwort

## **1 Einleitung**

---

- 1.1 Definitionssache
- 1.2 Sorry, das hier ist ein Mathematikbuch!
- 1.3 ChatGPT

## **2 Eine kleine Geschichte der KI**

---

- 2.1 Psychohistorie
- 2.2 Der Turing-Test und die Geburtsstunde der KI
- 2.3 Michie und die Streichholzschachteln
- 2.4 Deep Blue
- 2.5 DeepMind
- 2.6 Gute Nachrichten!
- 2.7 Schlechte Nachrichten ...

## 2.8 Die »Drei Gesetze der Robotik«

# 3 Etwas Fachjargon, mathematische und technische Grundlagen

---

## 3.1 Fachjargon und mathematische Konzepte

- 3.1.1 Mathematische Modelle
- 3.1.2 Lernen
- 3.1.3 Algorithmus
- 3.1.4 Abhängige Variable oder der Output
- 3.1.5 Unabhängige Variablen oder der Input
- 3.1.6 Daten
- 3.1.7 Merkmale
- 3.1.8 Optimierung
- 3.1.9 Überwachtes Lernen
- 3.1.10 Unüberwachtes Lernen
- 3.1.11 Vektor
- 3.1.12 Abstände zwischen Vektoren
- 3.1.13 Skalierung von Daten
- 3.1.14 Dimension
- 3.1.15 Klassen
- 3.1.16 Parameter
- 3.1.17 Kostenfunktion
- 3.1.18 Unteranpassung und Überanpassung
- 3.1.19 Training und Testen

## 3.2 Technische Grundlagen

- 3.2.1 Der K-Nearest-Neighbours-Algorithmus
- 3.2.2 Regression
- 3.2.3 Clustering
- 3.2.4 Entscheidungsbäume
- 3.2.5 Neuronale Netze
- 3.2.6 Reinforcement Learning

## **4 Mehr zu: K-Nearest-Neighbours**

---

- 4.1 Wofür K-Nearest-Neighbours eingesetzt werden
- 4.2 Im Detail
- 4.3 Aus dem Leben gegriffen: Ein Beispiel
- 4.4 Dein Projekt

## **5 Mehr zu: Regression**

---

- 5.1 Wofür Regression eingesetzt wird
- 5.2 Im Detail
- 5.3 Aus dem Leben gegriffen: Ein Beispiel
- 5.4 Dein Projekt

## **6 Mehr zu: Clustering**

---

- 6.1 Wofür Clustering eingesetzt wird
- 6.2 Im Detail
- 6.3 Aus dem Leben gegriffen: Ein Beispiel

## 6.4 Dein Projekt

## 7 Mehr zu: Entscheidungsbäumen

---

7.1 Wofür Entscheidungsbäume eingesetzt werden

7.2 Im Detail

7.3 Aus dem Leben gegriffen: Ein Beispiel

7.4 Dein Projekt

7.4.1 Galgenmännchen

7.4.2 Wordle

7.4.3 Mastermind

## 8 Mehr zu: Neuronalen Netzen

---

8.1 Wofür neuronale Netze eingesetzt werden

8.2 Im Detail

8.3 Aus dem Leben gegriffen: Ein Beispiel

8.4 Dein Projekt

## 9 Mehr zu: Reinforcement Learning

---

9.1 Wofür Reinforcement Learning eingesetzt wird

9.2 Im Detail

9.3 Aus dem Leben gegriffen: Ein Beispiel

9.4 Dein Projekt

## **Anhang**

---

Nachwort

Datensammlung

Danksagung

Über Paul Wilmott

## **Stichwortverzeichnis**

---

Rechtliche Hinweise

Über den Autor

*Für meine Lehrer*

# Vorwort

Wenn du das hier liest, besteht noch Hoffnung! Es ist noch nicht das Jahr 2029 nach der Apokalypse, Skynet hat die Welt noch nicht mit seinen unzerstörbaren, menschenvernichtenden KI-Cyborgs übernommen und es liegt auch kein deprimierender Grauton über der Welt. Wenn du James Camerons Film »*Terminator*« aus dem Jahr 1984 gesehen hast, kennst du seine Vision davon, was uns bevorstehen könnte, wenn die künstliche Intelligenz außer Kontrolle gerät. Und laut einer aktuellen Umfrage hat dieser Film die allgemeine Sicht auf künstliche Intelligenz (kurz KI) und die Entwicklung der Zukunft stark geprägt.

Im Moment scheint glücklicherweise noch die Sonne und die Vögel zwitschern. Dennoch gibt es beunruhigende Anzeichen dafür, dass Camerons Vorstellung vom Jahr 2029 schnell näher rückt. Mittlerweile kann die KI sogar, wie auch der schwarzeneggersche Cyborg im Film, menschliche Stimmen nachahmen. (Während ich diese Zeilen schreibe, sind Zeitreisen immerhin noch nicht möglich. Aber egal, was Einstein gesagt hat: Wenn jemand oder etwas das Zeitreisen erfinden kann, dann ist es die KI.)

Künstliche Intelligenz ist mittlerweile ein fester Bestandteil unserer Nachrichtenlandschaft. Manchmal sind die Anlässe dazu erfreuliche, wie zum Beispiel die Beteiligung der KI an der Entwicklung neuer

Impfstoffe. Viel häufiger berichten die Medien allerdings von den Gefahren, die die neue Technologie mit sich bringt: Meinungsmache, Hass und Hetze oder DeepFake-Videos von Prominenten in kompromittierenden Situationen. Der Papst, der in einer Balenciaga-Daunenjacke wie ein Rapper posiert, befand sich dabei eher am albernem Ende des Spektrums – das Bild schaffte es aber dennoch, viele Menschen zu täuschen.

Aber was meinen wir eigentlich, wenn wir »künstliche Intelligenz« sagen?

KI ist ein Zweig der Informatik, der sich mit der Entwicklung von Computerprogrammen und Maschinen befasst, die Eigenschaften besitzen, die normalerweise nur mit menschlicher Intelligenz in Verbindung gebracht würden. Sprachverständnis, Gesichtserkennung, die Diagnose von Krankheiten, Gaming, Autofahren, die Vorhersage von Aktienkursen – all das sind Beispiele dafür, was Computer mittlerweile leisten können. Es ist nicht lange her, dass die Vorstellung davon ausschließlich in Science-Fiction-Romanen zu finden war. KI wird darüber hinaus in sozialen Medien und beim Online-Shopping verwendet, um dir genau auf dich zugeschnittene Produkte zu empfehlen. Sie wird eingesetzt, um Bankbetrug zu erkennen und Spam-Mails zu löschen. Der KI ist es auch zu verdanken, dass dir nach dem Anschauen eines Films ein anderer Film empfohlen wird, der zu »99 % zu dir passt«.

Während die meisten KI-Modelle – hoffentlich – mit guten Absichten entworfen werden, gibt es viele Bedenken hinsichtlich des



angesprochenen Missbrauchs, wie zum Beispiel die angesprochenen DeepFake-Videos. Und da die künstliche Intelligenz mit rasender Geschwindigkeit weiterentwickelt wird, ist es nicht unwahrscheinlich, dass diese Entwicklung mit einer ebenso rasenden Verdummung der menschlichen Spezies einhergeht. Als Mensch, der täglich mit mathematischen Modellen arbeitet, bin ich ein großer Fan der Erhaltungsgesetze, und als solcher komme ich ehrlich gesagt nicht ganz davon weg, zu denken, dass Intelligenz im Universum erhalten bleibt: Wenn die künstliche Intelligenz zunimmt, nimmt die menschliche Intelligenz ab, sodass die Summe der beiden gleich bleibt. Du fragst, ob ich Witze mache? Ich kann's dir wirklich nicht sagen.

### **Wie hältst du's mit der Privatsphäre?**

Da KI mit riesigen Datenmengen arbeitet, ergeben sich darüber hinaus auch beachtliche Bedenken in Hinblick auf den Datenschutz. Immerhin werden immer mehr persönliche Daten aus deinen Seh-, Surf- und Shopping-Gewohnheiten generiert und gesammelt. Und obwohl es hilfreich erscheinen mag, dass ein Film zu 99 % zu deinen Vorlieben passt, fördern diese Vorschläge nicht gerade deine Offenheit für neue Filmgenres. Wenn du es begrüßenswert findest, dass dir zunehmend Entscheidungen abgenommen werden, dann wirst du die neue KI-Welt bestimmt mit offenen Armen willkommen heißen. Aber wenn du dir deine Privatsphäre, deine persönlichen Daten, die Medienvielfalt und eine ganz allgemeine Entdeckungsfreude bewahren möchtest, dann solltest du vielleicht

keine Filme bewerten, eher anonym surfen und deine Einkäufe im stationären Einzelhandel tätigen. (Und vergiss dabei nicht, in bar zu bezahlen. 😊)

## **Die Vorläufer der KI**

Das Konzept der KI gab es schon lange, bevor sie ein Feld wissenschaftlicher Forschung und Entwicklung wurde. Es ist bekannt aus Mythologie, Literatur und alten schwarz-weißen (Stumm-)Filmen. *Roboter* als Konzept sind einfach nur KI in menschenähnlicher Form. Ich nenne dir im Folgenden mal ein paar Beispiele dafür.

In der griechischen Mythologie begegnet uns zum Beispiel schon der bronzene Riese *Talos*, gebaut von Hephaistos, dem griechischen Gott der Erfindung und Schmiedekunst. Einen Stop-Motion-Talos, umgesetzt von Ray Harryhausen, kannst du im Film *Jason und die Argonauten* von 1963 bewundern. Dieser Film ist zu 99 % empfehlenswert (behaupte ich – als menschlicher Mensch), aber leider bei vielen Streaming-Diensten derzeit nicht verfügbar.



**Abbildung 1** Ein Ausstellungsflyer für den »Mechanical Turk«, einen Schachautomaten aus dem Jahr 1770

*The Mechanical Turk* bzw. auf Deutsch *Der Schachtürke* war ein Schachautomat, erbaut im Jahr 1770. Diese beeindruckende Maschine, die von einem ungarischen Mechaniker und Architekten erfunden worden war, Wolfgang von Kempelen, spielte gegen menschliche Herausforderer und gewann in den allermeisten Fällen. Sie wurde später an Johann Nepomuk Mälzel verkauft und tourte über 80 Jahre lang durch Europa und Amerika, wurde von Napoleon Bonaparte und Edgar Allan Poe in Augenschein genommen und letztlich bei einem Brand zerstört. Einer der Söhne des Besitzers des Schachtürken gab später zu, dass es sich bei der Maschine um einen

Schwindel gehandelt hatte: Ein menschlicher Schachspieler hatte in der Box unter dem Schachbrett gesessen und die Figuren bewegt.

In seinem Artikel von 1836, »Maelzel's Chess-Player«, der im *Southern Literary Messenger* erschien, verglich Poe den Mechanical Turk mit der Differenzmaschine von Charles Babbage. Wenn – wenn – der Türke echt gewesen wäre, so schrieb er, dann wäre er einem einfachen Rechner weit überlegen gewesen. Poe wusste, dass die Komplexität eines Schachspiels im Gegensatz zu ein paar Rechenaufgaben unglaublich hoch ist. Und Poes *Wenn* war großgeschrieben. Er machte damals sehr klar, dass er den Türken für einen Schwindel hielt.



**Abbildung 2** Ein Filmausschnitt aus dem Film »Metropolis«

Im Stummfilm *Metropolis* von Fritz Lang aus dem Jahr 1927, der in einer dystopischen Zukunft spielt, baut der Wissenschaftler Rotwang

einen Automaten, den *Maschinenmensch*, als Hommage an seine verstorbene Geliebte. Der Roboter wird später in eine menschliche Frau verwandelt, um die Zerstörung der Stadt zu provozieren.

Und dann gibt es noch den berühmten Roboter »Robby« aus dem Film *Alarm im Weltall* von 1956. Robby ist danach noch in vielen weiteren Shows und Filmen aufgetaucht und erwarb den Beinamen »Der am härtesten arbeitende Roboter Hollywoods«. Er trat auch in einigen Folgen der Serie *Twilight Zone* auf, die übrigens eine weitere hervorragende Quelle für finstere Vorhersagen über die Zukunft der KI ist – eine von vielen, vielen mehr.

## **Ausblick aufs Buch**

Ich habe dieses Buch geschrieben, um die vielen Gesichter künstlicher Intelligenz zu zeigen und dieses spannende Thema aus verschiedenen Perspektiven zu beleuchten. Auf den folgenden Seiten findest du neben einem historischen Überblick über die Entwicklung von KI und etwas Mathematik auch einige einfache Projekte für den Einstieg in das Thema. Einen wenig versteckten Hinweis auf diese Inhalte gibt auch schon der Titel des Buches: Mathematik ist das Herz der künstlichen Intelligenz – und das Herz dieses Buches. Heutzutage musst du weder Ahnung von Mathematik noch von Programmiersprachen haben, um KI nutzen zu können, aber *wenn* du wirklich an künstlicher Intelligenz mit all ihren Möglichkeiten und Hindernissen interessiert bist, dann musst du die Rolle der Mathematik in diesem komplexen System mindestens zu schätzen wissen.

## **Hinweis zur Kursivschreibung**

Wenn du in diesem Buch auf Text in *Kursivschrift* stößt, verweise ich in den meisten Fällen direkt auf eine Quelle, aus der ich zitiere. Die Quelle ist dann kursiv gesetzt. Manchmal nutze ich die Kursivschreibung auch zur Hervorhebung, für Buch- oder Filmtitel etc. – oder für nicht deutsche Wörter, bei denen eine Übersetzung nicht notwendig ist.

## **Übrigens: Projekte**

Zum Thema Projekte ... Immer, wenn ich in einem Buch lese: »Probieren Sie das doch mal zu Hause aus« oder »Kopieren Sie sich diese Seiten, damit ...« oder »Bastelprojekte für regnerische Ferientage mit Kindern« oder ganz besonders »Teambildende Maßnahmen mit Arbeitskolleg\*innen« etc., dann setze ich sie niemals nie tatsächlich in die Tat um. Und dieser Umstand sorgt bei mir auch niemals nie für ein schlechtes Gewissen.

Wenn du also niemals nie irgendeines meiner vorgeschlagenen Projekte durchführst: Fühl dich deswegen nicht schlecht. Ich habe jedes einzelne so geschrieben, dass du sie auch einfach gedanklich durchführen kannst. Du kannst dir zum Beispiel sehr lebhaft vorstellen, wie du Leute fragst,

wie groß sie sind oder wie viel sie wiegen. Dann stellst du dir vor, wie du alles in einem Notizbuch notierst, wie du ein Diagramm zeichnest und so weiter. Einstein hatte seine Gedankenexperimente, in diesem Buch haben wir Gedankenprojekte.

## Kapitel 1

# Einleitung

### 1.1 Definitionssache

Wir steigen hier mal mit einer Wikipedia-Definition ein:

*Künstliche Intelligenz (KI) ist maschinelle Intelligenz – die Aufnahme, Verarbeitung und Erschließung von Informationen – und steht damit im Gegensatz zur Intelligenz, die nicht menschlichen Tieren oder Menschen zu eigen ist. Beispielhafte Tätigkeiten künstlicher Intelligenz sind Spracherkennung, Computer Vision, Übersetzung (natürlicher) Sprachen sowie andere Zuordnungen von getätigten Eingaben.*

Eine weitere Definition aus Wikipedia:

*Maschinelles Lernen (ML) ist ein Teilgebiet der künstlichen Intelligenz, das sich mit Systemen befasst, die in der Lage dazu sind, ihr eigenes ›Wissen‹ (›Lernen‹) durch das Extrahieren von*



*Mustern aus Daten zu erhalten, anstatt diese Muster direkt über die Programmierung bereitgestellt zu bekommen.*

Wenn ich von der Mathematik hinter der KI spreche, meine ich damit das *maschinelle Lernen* bzw. *Machine Learning* (ML). Wenn wir KI im Einsatz sehen, wenn sie zum Beispiel nervige Mails beantwortet ein Strickmuster für einen Pullover entwirft, wissen wir, was die KI tut, aber wir haben höchstwahrscheinlich keine Ahnung, *wie* sie das tut. Und das unterscheidet KI und ML von Technologien aus dem letzten Jahrhundert. *Lernen* ist der Schlüssel. Wir schreiben diese ganzen komplizierten Algorithmen nicht, damit wir einem Computer beibringen können, *wie* man das Spiel Go spielt, sondern um ihm vermitteln zu können, *wie man lernt, wie man das Spiel Go spielt*.

(Übrigens: Wenn du die KI verwenden möchtest, um einen bevorstehenden Test zu bestehen oder sonst wie zu schummeln, beachte zwei Dinge. Erstens: Das nennt man Betrug, und du wirst ziemlich sicher erwischt werden ... Prüfungspersonal verwendet nämlich auch die KI, um herauszufinden, wer die KI verwendet hat. Uff, jetzt hab ich Kopfschmerzen! Und zweitens: Im Moment ist die KI noch sehr gut darin, gewisse Aufgaben zu erledigen (zum Beispiel DeepFake-Videos erstellen), aber sie kann auch echt mies in Mathe sein (siehe zum Beispiel den Abschnitt zu ChatGPT). Also achte in jedem Fall darauf, was du mit dem Output anstellst, den du der KI entlockt hast.)

## 1.2 Sorry, das hier ist ein Mathematikbuch!

Jup, so sieht's aus.

An dieser Stelle sagt sicherlich der eine oder die andere: »Juhu, mehr Mathematik! Das war schon immer mein Lieblingsfach!« Aber ich bin fast sicher, dass es auch Gegenstimmen gibt. Ich höre es schon: »Oh nein, mehr Mathematik! Mein absolutes Hassfach!«

Für Letztere habe ich gute Nachrichten: Es wird in diesem Buch viele nichtmathematische Inhalte geben – etwas Geschichte, ein paar nette Beispiele und Infos über künstliche Intelligenz im Allgemeinen. Und weiter hinten im Buch findest du Projekte, die du auch zusammen mit anderen ausprobieren kannst. (Da musst du dann als Nichtmathematiker\*in natürlich immer noch keine mathematischen Berechnungen anstellen – vielmehr ist es das Ziel, möglichst viele Daten zu sammeln, mit denen du KI-Algorithmen trainieren kannst. Du könntest zum Beispiel alle möglichen Leute aus deinem Umfeld befragen, wie alt sie sind und was sie so verdienen. Ich wünsche dir ganz viel Erfolg dabei!)

Die schlechte Nachricht ist allerdings, dass du, wenn du professionell mit künstlicher Intelligenz arbeiten möchtest, unter Umständen Mathematik an der Universität studieren solltest. Oder Informatik. Aber wirklich nur, wenn du eine dieser Personen sein willst, die KI-

Anwendungen selbst entwickeln. Wenn du KI nur verwenden möchtest, um nervige Mails zu beantworten ein Pulloverdesign zu erstellen, dann brauchst du aktuell noch kein Studium, sondern nur eine Tastatur und eine Internetverbindung. Aber (noch mehr gute Nachrichten!) die Mathematik in diesem Buch ist wirklich ziemlich niedrigschwellig und leicht zu verstehen. Ich werde dich nicht mit wissenschaftlichen Fachbegriffen zuschütten, und du musst auch kein Programmiergenie sein, um die Inhalte zu verstehen. Alles, was wir hier besprechen, wird nur Bleistift und Papier erfordern.

## 1.3 ChatGPT

Ich muss an dieser Stelle kurz ChatGPT erwähnen, da es das erste KI-Tool war, das weit verbreitet eingesetzt wurde. ChatGPT ist ein KI-Chatbot, der einen Algorithmus namens *Generative Pre-trained Transformer* (GPT) verwendet. Das ist eine Art neuronales Netzwerk (was das wiederum ist, erfährst du in einem späteren Kapitel), das nach dem Training mit großen Datenmengen dazu in der Lage ist, neue Inhalte zu generieren. Dieser Content ist dann nicht mehr (ganz einfach) von den Antworten und Inhalten zu unterscheiden, die ein Mensch vielleicht gegeben bzw. erstellt hätte.

ChatGPT ging Ende November 2022 live und konnte innerhalb von ein paar Monaten über 100 Millionen Nutzer gewinnen.

Obwohl ChatGPT von Anfang an ziemlich gut darin war, plausibel klingende Antworten auf diverse Fragestellungen zu produzieren, wurde ziemlich schnell klar, dass der Inhalt dieser Antworten nicht immer ganz den Fakten entsprach. ChatGPT war auch ziemlich schlecht darin, einfache Mathematikaufgaben zu lösen, wie zum Beispiel quadratische Gleichungen. Selbst Monate nach dem Start war das Programm immer noch nicht dazu in der Lage, solche Gleichungen korrekt zu lösen. Das ist besonders deswegen seltsam, weil doch die besondere Stärke von ChatGPT seine Lernfähigkeit sein sollte. Das Feedback all der Mathematiker und Mathematikerinnen,

die sich über die falschen Ausgaben lustig machten, hätte zu einer Verbesserung des Algorithmus führen sollen. Aber sechs Monate nach dem Launch machte ChatGPT immer noch die gleichen Fehler. Und sechs Monate sind für eine KI im Vergleich zu uns wahrscheinlich ungefähr hunderttausend Jahre menschlicher Evolution!

### **Übrigens: KI zur Lösung einfacher Matheaufgaben verwenden. Findest du den Fehler?**

Ich: Löse  $x^2 - 8x - 260 = 0$ .

ChatGPT: Um die quadratische Gleichung  $x^2 - 8x - 260 = 0$  zu lösen, können wir sie entweder faktorisieren oder die quadratische Formel verwenden. Lassen Sie sie uns faktorisieren:

$$x^2 - 8x - 260 = 0$$

Wir müssen zunächst zwei Zahlen finden, deren Produkt  $-260$  ist und deren Summe  $-8$  ist (Koeffizienten des  $x$ -Terms). Nach Berücksichtigung der Faktoren von  $260$  können wir die Gleichung wie folgt umschreiben:

$$(x - 20)(x + 13) = 0$$

Nun setzen wir jeden Faktor gleich null:

$$x - 20 = 0 \text{ oder } x + 13 = 0$$

Und lösen nach  $x$  auf:

Für  $x - 20 = 0$  gilt also  $x = 20$

Für  $x + 13 = 0$  gilt also  $x = -13$

Daher ist die Lösung der Gleichung  $x^2 - 8x - 260 = 0$  die folgende:

$x = 20$  und  $x = -13$ .

## Kapitel 2

# Eine kleine Geschichte der KI

Ich habe im vorigen Kapitel bereits einige Beispiele für künstliche Intelligenz in der klassischen Mythologie, in Filmen und Fernsehen erwähnt. Sie repräsentieren allerdings eher die damals vage Vorstellung davon, wie Roboter, computergesteuerte Schachspieler usw. mal funktionieren könnten. Die Frage danach, ob diese Maschinen einer festgelegten Schritt-für-Schritt-Programmierung folgten oder tatsächlich *lernten*, ist dabei eher Gedankenspielerei – denn keines der genannten Beispiele ist tatsächlich real.

## 2.1 Psychohistorie

Es gibt allerdings ein weiteres Beispiel aus der Literatur (na gut, das ist immer noch nicht real), das einen Hinweis darauf gibt, wie KI funktionieren könnte. Dieses Beispiel haben wir der Figur *Hari Seldon* in Isaac Asimovs Science-Fiction-Buchserie *Foundation* aus den 1940er- und 1950er-Jahren zu verdanken. Hari Seldon ist ein Mathematikprofessor an der Streeling University auf dem Planeten

Trantor. Er entwickelt in den Büchern ein neues Forschungsfeld, die Psychohistorie, das sich zur Aufgabe macht, die Zukunft mithilfe statistischer Daten vorherzusagen. Der Schlüssel dazu ist die Verwendung riesiger historischer Datenmengen. Während es durch diese Grundlage natürlich unmöglich ist, das Verhalten des Einzelnen vorherzusagen, könnte das grundsätzliche Verhalten der menschlichen Gesellschaft vorhersehbar sein. Analog dazu: Du kannst unmöglich den Weg eines einzelnen Regentropfens vorhersagen, da er vom Wind und anderen Tropfen beeinflusst wird – wenn du aber aus dem Fenster schaust und siehst, dass es regnet, könntest du eine grobe Ahnung davon bekommen, ob es klug ist, einen Regenschirm mitzunehmen. Dieser Einsatz großer Datenmengen ist eine der Eigenschaften des maschinellen Lernens, wie wir es heute kennen.



## **2.2 Der Turing-Test und die Geburtsstunde der KI**

1950 schlug der englische Mathematiker Alan Turing (diesmal eine real existierende Person!) einen einfachen Test vor, um zu prüfen, ob eine künstliche Intelligenz überhaupt denkbar sein könnte. Dieser Test ist quasi die Vorstufe zu der Frage, ob eine Maschine denken kann. Turings Testfrage war denkbar simpel: Könnte ein Leser oder eine Leserin eines textbasierten Gesprächs zwischen einem Computer und einem Menschen sagen, was von dem Menschen und was von der Maschine geschrieben wurde?

In den späteren 1950er-Jahren wurden Erzählungen zufolge bei einem Workshop am Dartmouth College in New Hampshire das Feld der KI und der Name »künstliche Intelligenz« erfunden. Zu diesem Workshop wurden mehrere bekannte Forscher aus verschiedenen wissenschaftlichen Fachbereichen eingeladen, um über die Möglichkeiten künstlicher Intelligenz in der Zukunft nachzudenken. Leider nahmen diese prominenten Fachleute jeweils nur kurz und in unterschiedlichen Zeitslots teil, sodass bei dem Workshop nicht viel herauskam.

## 2.3 Michie und die Streichholzschachteln

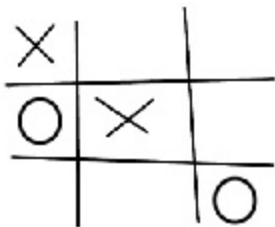


**Abbildung 2.1** Donald Michie, der »Vater der künstlichen Intelligenz in Großbritannien«

An dieser Stelle darf ich dir Donald Michie vorstellen, der oft auch als »Vater der künstlichen Intelligenz« bezeichnet wird. Donald Michie arbeitete während des Zweiten Weltkriegs mit seinem Kollegen Alan Turing daran, feindliche Chiffren und Codes zu knacken. Nach dem Krieg promovierte er und wandte sich in den frühen 1960er-Jahren als Professor in Edinburgh der Aufgabe zu, einem Computer das Spiel *Noughts and Crosses* beizubringen, das dir vielleicht auch als *Tic Tac Toe* oder *Drei gewinnt* bekannt ist. Nur verwendete Michie dafür keinen Computer, sondern eine bestimmte Anzahl an Streichholzschachteln (ohne die Streichhölzer). Genauer gesagt

waren es 304 Streichholzschachteln, die die unterschiedlichen Spielzüge einer Partie Tic Tac Toe abbildeten.

Du weißt bestimmt, wie man dieses Spiel spielt: Es gibt ein Drei-mal-drei-Raster, auf dem die beiden Spieler\*innen abwechselnd ihr eigenes Symbol eintragen (typischerweise einen Kreis und ein Kreuz). Das Ziel des Spiels besteht darin, als Erste\*r drei dieser Symbole in eine Reihe zu bekommen. Ich gehe mal davon aus, dass du das Spiel schon mal gespielt hast und weißt, wie man gewinnt – oder zumindest, wie man nicht verliert.



**Abbildung 2.2** Ein Tic-Tac-Toe-Spielstand, wie ihn Michie mit seinen Streichholzschachteln abgebildet hat

Du könntest jetzt einen Computercode schreiben, der dir voraussagt, was der beste Zug für jede einzelne Phase des Spiels ist. Das wäre nicht allzu schwierig, nur echt ein bisschen langweilig. Es gibt insgesamt 304 mögliche Szenarien (wenn man berücksichtigt, dass es viele Symmetrien gibt, die die Anzahl der Szenarien reduzieren). Also hättest du 304 Anweisungen, welchen Zug du jetzt machen könntest. In [Abbildung 2.2](#) siehst du beispielhaft den Spielstand einer Tic-Tac-Toe-Partie. Unser Computercode würde uns jetzt, angenommen wir

spielen X und sind dran, empfehlen, unser X in der oberen Mitte zu platzieren (oben rechts wäre allerdings genauso gut). Was auch immer O tun würde, wir würden im nächsten Zug gewinnen.

Für diese Vorhersagen bräuchten wir also 304 Anweisungen bzw. 304 Codezeilen für den Computer. Jetzt sind 304 Codezeilen eigentlich wirklich nicht viel. Für Tic Tac Toe ist der Computercode also nicht allzu kompliziert. Geht man jetzt allerdings zu anderen, komplexeren Spielen mit mehr individuellen Momentaufnahmen des Spielstandes und mehr möglichen Zügen über, wird das mit dem Code extrem schwierig, vielleicht sogar unmöglich.

Professor Michie ging Tic Tac Toe nämlich völlig anders an. Und das Schöne an seinem Ansatz ist, dass er toll verbildlicht, was mit traditioneller Programmierung nicht dargestellt werden kann.



**Abbildung 2.3** Eine Darstellung von Michies perlengefüllten Streichholzschachteln

Professor Michie zeichnete auf jede der 304 Streichholzschachteln je einen der 304 möglichen Spielstände und füllte die Schachteln dann mit farbigen Perlen. Du wirst gleich sehen, dass die so vorbereiteten Streichholzschachteln wirklich wie ein Computerprogramm funktionierten.

Auf [Abbildung 2.3](#) siehst du, wie *eine* von Professor Michies Streichholzschachteln ausgesehen haben könnte. Es ist eine gewöhnliche Streichholzschachtel mit einer Skizze von einem Tic-Tac-Toe-Spielstand (übrigens demselben wie oben in [Abbildung 2.2](#) schon gezeigt). Hier spielt der Computer mit X und ist an der Reihe. Es gibt fünf Plätze, an denen der Computer sein X platzieren könnte – wie entscheidet er sich nun am besten? Wir wissen bereits, dass es zwei Felder gibt, an denen wir unser X platzieren können, die zu einem Gewinn führen würden – nämlich die obere Mitte und oben rechts. Aber das Ziel ist ja nicht, dem Computer zu vermitteln, wie er jetzt am besten spielt. Wir wollen erreichen, dass der Computer das selbst herausfindet!

Dafür sind die Perlen da. Ihre Farben finden sich auf den Feldern im Spieleraster wieder. Der nächste Zug wird nun einfach bestimmt, indem das Fach der Streichholzschachtel geschlossen, die Box geschüttelt und eine zufällige Perle herausgenommen wird. Die Farbe, die gezogen wird, bestimmt den Zug, den der Computer als nächsten macht.

Der Zug, den der Computer macht, ist bis zu dieser Stelle also zufällig. Der clevere Teil kommt gleich erst ...

Die Schachtel mit der Zufallsperle wird beiseitegestellt. Jetzt ist O an der Reihe. Nach dem Zug gibt es einen neuen Spielstand, deswegen nehmen wir uns jetzt die Streichholzschachtel mit diesem Stand (eine von den 304) und wählen wieder eine farbige Perle zufällig aus. In dieser Schachtel wird die Auswahl nicht so groß sein, da es weniger

mögliche Felder gibt, auf denen das nächste X platziert werden könnte.

So wird weitergespielt, bis das Spiel vorbei ist. Und jetzt kommt der Twist an der Sache ...

Wenn wir als Computer das Spiel gewonnen haben, »belohnen« wir alle Streichholzschachteln, die in diesem Spiel verwendet wurden, indem wir mehrere Perlen mit der verwendeten Farbe in die Schachteln zurücklegen. Die Farbe, die uns bei genau dem auf der Schachtel abgebildeten Spielstand zum Sieg verholfen hat, wird also häufiger in der Schachtel vertreten sein, wenn wir das nächste Mal diese Position spielen. Wenn wir verloren haben, nehmen wir nach dem Spiel einige Perlen dieser Farben heraus. Wenn es ein Unentschieden ist, legen wir die eine ausgewählte Perle zurück in ihr Fach, sodass sich nichts ändert.

Schließlich füllen sich mit jeder Partie die Streichholzschachteln mit Perlen, die die erfolgreichste Aktion für den Spielstand auf der Schachtel repräsentieren – und die Perlen, die die schlechteren Züge repräsentieren, verschwinden mit der Zeit. Wenn wir jetzt die Streichholzschachteln benutzen, um den besten nächsten Zug voraussagen zu lassen, werden immer die besten Züge in jeder Phase des Spiels ausgewählt. Andersfarbige Perlen sind in den Schachteln gar nicht mehr vorhanden.

Ist das nicht absolut fantastisch?

Wir haben dem »Computer« nie *gesagt*, was der beste Zug bei jedem Spielstand ist. Stattdessen hat er die besten Züge *gelernt*, indem er für Erfolg belohnt und für Misserfolg bestraft wurde.



**Abbildung 2.4** Michies Streichholzschachteln auf einer Fotografie

Michie nannte seine Maschine MENACE, kurz für *Machine Educable Noughts And Crosses Engine*. Sie hat gelernt, wie man am besten Tic Tac Toe spielt. Professor Michies Methode war bahnbrechend – sowohl für das, was heute als »reinforcement learning« bekannt ist, als auch hinsichtlich seiner physischen Darstellung eines Computerprogramms. Ein Assoziierter Professor für Data Science in Oxford ließ in seinen Kursen seine Studierenden MENACE nachbauen. Obwohl heutige Computer viel schneller sind und komplexere Aufgabenstellungen lösen können, glaubte der Professor, dass MENACE die grundlegenden Prinzipien der Informatik perfekt vermitteln kann.

## 2.4 Deep Blue

Nun aber weg von den Streichholzschachteln und hin zu echten Computern. »*Deep Blue*« ist ein Schachprogramm, das ab 1985 zunächst von der Carnegie Mellon University und später von IBM entwickelt wurde. 1995 spielte es gegen den damals amtierenden Weltmeister Garry Kasparov, verlor jedoch vier von sechs Spielen. Im nächsten Jahr allerdings, nach weiterer Entwicklungsarbeit, besiegte es ihn – und wurde somit die erste künstliche Intelligenz, die einen Weltmeister besiegte.



## 2.5 DeepMind

Besonders interessant finde ich auch die Geschichte, wie die Entwickler\*innen der Firma Google DeepMind einen Computer programmierten, Go zu lernen. 2016 besiegte dieser auch prompt Lee Sedol, einen professionellen Go-Spieler mit dem 9. Dan. Für diese Software, *AlphaGo*, wurde eine Vielzahl technischer Möglichkeiten eingesetzt, darunter auch neuronale Netzwerke. Das Programm *lernte* dadurch zu spielen, ohne dass ihm jemals explizit vorgegeben wurde, wie das perfekte Go-Spiel abzulaufen hat. Allerdings spielte AlphaGo selten gegen menschliche Gegner, sondern hauptsächlich gegen sich selbst.

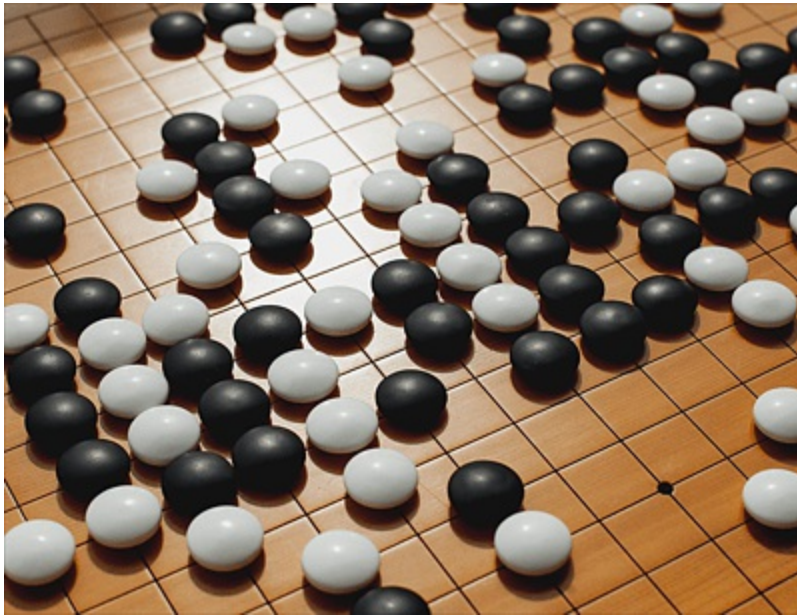
Für Google DeepMind war es eine echte Mammutaufgabe, ausgerechnet ein Go-Spiel zu programmieren. Zum Vergleich: Würde man Streichholzschachteln verwenden, um die individuellen Spielstände darzustellen, bräuchte man wesentlich mehr als 304 davon: Man müsste wahrscheinlich das bekannte Universum mit Streichholzschachteln füllen. An dieser Stelle würde ich dringend empfehlen, den Film *The AlphaGo Movie* anzuschauen (zugegeben, gäbe es eine Oscar-Kategorie für den langweiligsten Filmtitel, wäre dieser ein heißer Kandidat dafür). In diesem Film bekommst du nicht nur eine umfangreichere Beschreibung der Hintergründe, sondern kommst darüber hinaus auch in den Genuss, die Gesichter aller

Beteiligten zu sehen. Lee Sedol beginnt die Partie mit AlphaGo relativ zuversichtlich, verliert dann mehr und mehr die Fassung und gibt schließlich auf (mental und buchstäblich). Die meisten am Projekt beteiligten Programmierer jubeln vor Begeisterung, als der zweite Zustand eintritt. Es gibt allerdings auch einen sehr nachdenklich und besorgt aussehenden Kollegen im Hintergrund. (Entweder hat er Verdauungsstörungen, oder er macht sich Sorgen darüber, wohin das mit der KI noch alles führen könnte.) Im Laufe des Films macht der Computer so einige seltsame Züge – Züge, die kein professioneller Go-Spieler bzw. keine Spielerin gemacht hätte. Im Laufe der Handlung gibt es Diskussionen darüber, ob es sich dabei um einen Programmierfehler, mangelhaftes Lerntraining oder einen absoluten Geniestreich der Computerwissenschaft handelt. (Spoiler: Letzteres war schließlich der Fall.) Man geht mittlerweile davon aus, dass das Spiel Go an sich in seiner Komplexität weiterentwickelt wurde, weil die Menschen durch das Machine Learning und die Übertragung auf ein Computerprogramm bei diesem Projekt so viel darüber gelernt haben.

### **Übrigens: Das Spiel Go**

Go ist ein Brettspiel für zwei Spieler\*innen, das auf einem Brett mit einem 19×19-Raster gespielt wird (Neulinge spielen oft auf einem kleineren Brett mit 9×9 oder 13×13 Feldern). Die schwarzen und weißen Spielsteine werden auf den Ecken platziert, an denen sich die Rasterlinien treffen. Das Ziel des Spiels ist es, mit den eigenen Steinen mehr Fläche als der Gegner einzuschließen. Das Spiel wurde vor ca. 2.500 Jahren

in China erfunden und ist möglicherweise das älteste Brettspiel der Welt, das heute noch gespielt wird. Ich kann dir wirklich nur empfehlen, dieses Spiel zu lernen! Schon beim Anblick eines 9×9-Bretts kannst du dir etwa vorstellen, wie viele mögliche Spielzüge dieses Spiel hat.



**Abbildung 2.5** *Nahaufnahme von einem Go-Spielbrett*

## 2.6 Gute Nachrichten!

- **New York Times, 25. Mai 2023: »Gelähmter kann dank Gehirn- und Wirbelsäulenimplantaten wieder selbstständig laufen«**

Gert-Jan Oskam lebte 2011 in China, als er bei einem Motorradunfall verletzt wurde und sich anschließend im Rollstuhl wiederfand. Ein Forschungs- und Entwicklungsteam konnte ihm mithilfe mehrerer Implantate die Kontrolle über seine Beine zurückgeben.

Es ist eine unfassbar gewaltige Aufgabe, die unzähligen elektrischen Signale zu interpretieren, die unser Körper erzeugt, wenn wir ein Körperteil bewegen möchten. Welches Signal bewirkt was – und welches ist für die Bewegung verantwortlich?

In dieser neuen Studie nutzte die sogenannte Gehirn-Wirbelsäulen-Schnittstelle einen künstlichen Intelligenz-Gedanken-Dekoder, um die Absichten von Herrn Oskam – als elektrische Signale in seinem Gehirn messbar – zu erkennen und entsprechende Muskelbewegungen auszuführen.

- **The Lancet, 21. April 2023: »Die Entwicklung und internationale Zulassung von Deep-Learning-Modellen für die Erkennung der Netzhautschädigung bei Frühgeborenen-Retinopathie: eine retrospektive Studie«**

Der Mangel an Expertinnen und Experten auf dem Gebiet der pädiatrischen Augenheilkunde hat zur Folge, dass eine der Hauptursachen für Erblindungen im Kindesalter, die Frühgeborenen-Retinopathie, häufig nicht korrekt diagnostiziert wird. Ein Forschungsteam hat nun ein KI-Programm entwickelt, das mit Tausenden von Bildern der Augen neugeborener Babys trainiert wurde. Die KI war anschließend ebenso erfolgreich bei der Feststellung der Erkrankung wie die Expert\*innen für Augenheilkunde. Sie ermöglicht seitdem auch dort Diagnosen, wo keine Spezialisten und Spezialistinnen ansässig sind.

- **Phys.org, 28. September 2022: »KI besser als Menschen bei der Erkennung von Blauwal-Rufen«**

Mithilfe maschinellen Lernens hat ein Team der Australian Antarctic Division des K. Lisa Yang Center for Conservation Bioacoustics an der Cornell University und der Curtin University einen Algorithmus trainiert, der Blauwal-Gesänge [»D-Calls«, soziale Rufe, die Wale in ihren Futtergründen machen] in Tonaufnahmen schneller und genauer als menschliche Expert\*innen erkennt.

Sogenannte »D-Calls« variieren mit dem Alter des Wals stark und unterscheiden sich von Wal zu Wal. Dies macht ihre Analyse, insbesondere die automatisierte Analyse, sehr schwierig – nahm man zumindest bisher an.

Die Datengrundlage der Forscher\*innen waren 5.000 Walrufe aus der Antarktis. Sechs Menschen teilten diese Rufe den Kategorien »D-Call« oder »Kein D-Call« zu. Der Algorithmus wurde dann mit

diesem klassifizierten Datensatz trainiert. Schließlich wurde der Algorithmus auf einen Testsatz neuer Rufe angewendet, um abgleichen zu können, wie gut er im Vergleich zur Klassifizierung durch menschliche Expert\*innen abschnitt. Ein unabhängiger Gutachter, Dr. Miller (ein Mensch!), legte anschließend fest, welche Klassifikationen durch Mensch und Maschine korrekt waren.

Die Ergebnisse waren eindeutig: Die KI fand 90 % der D-Calls, während die Menschen nur 70 % von ihnen fanden. Ein klarer Sieg für die Maschine.

»Es kostete menschliche Expert\*innen etwa 10 Stunden, diesen Testdatensatz zu annotieren. Die KI brauchte nur 30 Sekunden, um die Daten zu analysieren – sie ist damit 1.200-mal schneller«, sagte Dr. Miller.



**Abbildung 2.6**    *Die (KI-generierte) Zeichnung eines Blauwals*

- **United Nations Environment Programme, 7. November 2022:**  
**»Wie uns künstliche Intelligenz hilft, unsere Umweltprobleme zu lösen«**

Es stehen mehr Daten über unser Klima zur Verfügung als je zuvor. Für unseren weiteren Umgang mit dem Klimawandel ist allerdings entscheidend, wie man auf diese Daten zugreifen, sie interpretieren und darauf reagieren kann und sollte. Eine Technologie, die für all diese Vorgänge eine zentrale Rolle spielt, ist die künstliche Intelligenz.

KI wird in zahlreichen Bereichen eingesetzt, um die Klimakrise zu lösen oder zumindest abzumildern. Um nur einige zu nennen: KI

kann bereits Häuser energieeffizienter machen, die Entwaldung dokumentieren, Methanwerte analysieren und auswerten sowie Vorhersagen über den CO<sub>2</sub>-Gehalt in der Luft und das Ansteigen der Meeresspiegel treffen. Die Karten-App auf deinem Handy ist bereits jetzt dazu in der Lage, dir die beste Fahrtroute von A nach B anzuzeigen, um die CO<sub>2</sub>-Emissionen deines Autos niedrig zu halten.

Aber natürlich ist auch in diesem Bereich unseres Lebens nicht alles eitel Sonnenschein. All diese KI-Analysen bringen nämlich ihre ganz eigenen Umweltkosten mit sich:

Der ICT-Sektor [Information and Communications Technology] erzeugt etwa 3–4 % der europaweiten Kohlenstoffemissionen, und Rechenzentren verbrauchen riesige Mengen Wasser für die Kühlung.

Was uns auch schon zum nächsten Abschnitt bringt ...



## 2.7 Schlechte Nachrichten ...

- **The Conversation, 23. Mai 2023: »Ist generative KI schlecht für die Umwelt?«**

Im Jahr 2019 stellten Forschende fest, dass die Erstellung einer generativen KI [also eines Modells, das komplexe Ausgaben wie Bilder, Sätze oder Musik erzeugt] namens BERT mit 110 Millionen trainierbaren Parametern den Energieverbrauch eines transkontinentalen Hin- und Rückflugs für eine Person hatte.

Die Anzahl der Parameter gibt dabei die Größe des Modells an, wobei größere Modelle natürlich leistungsfähiger sind. Forschende schätzten, dass die Erstellung des wesentlich größeren GPT-3, das 175 Milliarden Parameter hat, 1.287 Megawattstunden Strom verbrauchte und 552 Tonnen Kohlendioxidäquivalent erzeugte – umgerechnet ist das der Jahres-CO<sub>2</sub>-Verbrauch von 123 Pkws. Und das nur in der Entwicklungsphase, bevor das Modell überhaupt zum Einsatz kam.

Bevor wir also darüber nachdenken können, welche apokalyptischen Szenarien zukünftig durch die KI verursacht werden könnten, müssen wir uns mit den Umweltfolgen auseinandersetzen, die schon die Entwicklung einer KI mit besten

Absichten mit sich bringt. Strom- und (Kühl-)Wasserverbrauch sind für diese Modelle immens.

- **The Red Hand Files, Januar 2023: »Ich habe ChatGPT einen Song im Stil von Nick Cave schreiben lassen. Wie findest du es?«**

Erste Strophe:

In the depths of the night, I hear a call

A voice that echoes, through the hall

It's a siren's song, that pulls me in

Takes me to a place, where I can't begin

Refrain:

I am the sinner, I am the saint

I am the darkness, I am the light

I am the hunter, I am the prey

I am the devil, I am the savior

Zweite Strophe:

I walk the streets, with my head held high

I've got my demons, but I'll make them fly

I'll dance with the devil, and I'll play his game

I'll be the one, to call his name

...

Und Nick Cave antwortet prompt:

Lieber Mark,

Seit der Einführung von ChatGPT im letzten Jahr haben mir viele Menschen, die meisten irgendwie von einer Art algorithmischer Ehrfurcht erfüllt, Songs »im Stil von Nick Cave« geschickt, die damit erstellt wurden. Es waren wirklich Dutzende Nachrichten. Ich teile die Begeisterung für diese Technologie überhaupt nicht. Klar, ChatGPT steckt noch in den Kinderschuhen, aber vielleicht ist genau das auch das Schlimmste an der KI – dass sie einfach für immer in ihren Kinderschuhen stecken und nie ganz fertig sein wird, in einer ewigen Vorwärtsbewegung gefangen, immer schneller. Sie kann nie innehalten, langsamer werden oder einen Schritt zurück machen. Wir bewegen uns potenziell auf eine utopische Zukunft zu – oder auf unsere völlige Zerstörung. Kann man halt nicht so genau sagen.

Wenn ich mir allerdings diesen Song »im Stil von Nick Cave« ansehe, sieht es ziemlich mies aus, Mark. Da sehe ich die Apokalypse direkt vor der Tür stehen. Dieser Song ist einfach nur unfassbar schlecht geschrieben ...

Songs entstehen aus emotionalem Leid heraus – sie basieren auf dem vielschichtigen inneren Bedürfnis des Menschen, Dinge zu erschaffen. Soweit ich weiß, haben Algorithmen keine Emotionen.

Daten kennen kein Leid. ChatGPT hat kein inneres Bedürfnis, muss nichts ertragen, konnte keine Erfahrungen in dieser Welt machen und wird nie spüren, wie fantastisch es sich anfühlt, die eigenen Grenzen der Wahrnehmung immer wieder neu auszuloten. Es hat keine Grenzen, die es überschreiten könnte. Die herzerreißend traurige Bestimmung von ChatGPT ist es, für immer nur zu imitieren und niemals eine authentisch menschliche Erfahrung zu machen.

Mark, danke für den Song, aber bei allem Respekt und aller Liebe auf der Welt, er ist echt \*\*\*\*\*, eine groteske Beleidigung demgegenüber, was es bedeutet, Mensch zu sein. Wie du vielleicht siehst, bin ich echt kein Fan – wobei, Moment! Wenn ich den Song jetzt noch mal so lese, gibt es doch eine Zeile, die mich anspricht.

»I've got the fire of hell in my eyes«

steht da in diesem Song »im Stil von Nick Cave«, und das ist irgendwie wahr. Ich habe Höllenfeuer in meinen Augen – es ist ChatGPT.

Liebe Grüße,

Nick

- **The Guardian, 24. Mai 2023: »Neuseelands Nationalpartei gibt zu, künstliche Intelligenz zur Erzeugung von Videomaterial in Attack Ads verwendet zu haben«**

Die Nationalpartei Neuseelands hat zugegeben, künstliche Intelligenz zur Erzeugung von Videomaterial in ihren Attack Ads

gegen ihre politische Konkurrenz verwendet zu haben.

Die Videos enthielten Bilder einer Gruppe von Kriminellen, die einen Juwelierladen stürmt, von zwei Pflegefachkräften und von einem mutmaßlichen Verbrechenopfer. Ein Video der Kampagne zeigte sogar die Besetzung der Fast-and-Furious-Verfilmung.

Die Videos, auf denen beispielsweise eine Frau mit riesigen Augen, zwei Pflegekräfte mit seltsam wachsartiger Haut und Kriminelle mit merkwürdig geformten Sturmhauben zu sehen sind, erregten schnell den Verdacht, KI-generiert zu sein.

An dieser Geschichte ist erst mal nichts Illegales. Und man muss den Parteimitgliedern zugutehalten, dass sie ziemlich schnell zugegeben haben, KI verwendet zu haben. Allerdings ist das erst der Anfang vom Ende ...

- **The Guardian, 6. April 2023: »ChatGPT erfindet gefälschte Guardian-Artikel«**

Letzten Monat erhielt ein Kollege aus unserem Redaktionsteam eine interessante E-Mail. Eine Forscherin war auf die Erwähnung eines Guardian-Artikels gestoßen, der vor einigen Jahren vom besagten Kollegen geschrieben worden war. Aber der Artikel war auf unserer Website und in der Suche nicht aufzufinden. War der Titel vielleicht seit der Veröffentlichung geändert worden? ... Der Redakteur konnte sich nicht daran erinnern, den Artikel überhaupt geschrieben zu haben, aber die Überschrift klang sehr nach ihm. Es ging um ein Thema, mit dem er sich sehr identifizierte und zu dem

er auch schon öfter geschrieben hatte. Dennoch fand sich im Archiv keine Spur von dem Artikel. Warum? Weil er nie geschrieben worden war.

Glücklicherweise hatte uns die Forscherin mitgeteilt, dass sie ihre Recherche mit ChatGPT durchgeführt hatte. Auf die Frage nach Artikeln zum gesuchten Thema hatte die KI einfach einige erfunden. Die Sprachgewandtheit der KI und die riesige Menge an Trainingsdaten, die sie verarbeitet, ließen sogar einen Redakteur in seinem Fundus nach einem Artikel suchen, den er überhaupt nicht geschrieben hatte.

Es gibt also nicht nur Menschen, die die KI für zweifelhafte Zwecke verwenden – manchmal halluziniert künstliche Intelligenz auch einfach selbst falsche Nachrichten und Artikel, ohne sie zu kennzeichnen.

Und dann wäre da noch ...

- **Sky News, 23. Juni 2023: »Anwälte nutzten ChatGPT zur Unterstützung bei einem Fall – und landeten vor Gericht«**

Zwei New Yorker Anwälte wurden mit einer Geldstrafe belegt, nachdem sie einen Rechtsbrief mit gefälschten Fallziten eingereicht hatten, die von ChatGPT generiert wurden.

Steven Schwartz von der Anwaltskanzlei Levidow, Levidow & Oberman gab zu, den Chatbot zur Recherche in einem Fall eines

Kanzleiklienten gegen die Fluggesellschaft Avianca verwendet zu haben.

Er hatte ihn benutzt, um rechtliche Präzedenzfälle zu finden, die seine Position hätten stützen können, aber die Anwälte, die die kolumbianische Fluggesellschaft juristisch vertraten, teilten dem Gericht mit, dass sie einige der zitierten Beispiele nicht finden konnten – verständlich, denn sie waren fast vollständig erfunden.

Einige der Präzedenzfälle waren komplett gefälscht, während andere falsche oder nicht existierende Richter\*innen zitierten oder Fluggesellschaften als Beteiligte nannten, die es nie gab.

Bezirksrichter Peter Kevin Castel sagte, Schwartz und sein Kollege Peter LoDuca, der an Schwartz' Zusammenfassung falscher Quellen ebenfalls beteiligt war, hätten in böser Absicht gehandelt und »bewusst Themen vermieden und falsche und irreführende Aussagen vor Gericht« gemacht.

Teile des Briefs waren »grober Unsinn« und enthielten »Kauderwelsch« sowie gefälschte Zitate, fügte der Richter hinzu.

Ich kann mir ja wirklich vorstellen, dass die KI irgendwann Anwälte\*innen ersetzt – aber da sind wir einfach noch lange nicht.

**Übrigens: Bildgenerierung, »Elvis has left the building, ... und lebt jetzt in einem Wohnwagen«**



**Abbildung 2.7** *Das KI-generierte Bild eines gealterten Elvis'*

Ich habe mit Microsoft Bing ein Foto vom 80-jährigen Elvis Presley generiert, der in einem Wohnwagen lebt.

Abgesehen von den *drei* völlig deformierten Händen (Arthritis ist schlimm, aber so schlimm nun auch wieder nicht) sieht der Mensch auf dem Bild auch überhaupt nicht aus wie Elvis ... Na gut, ich gebe zu, das Bild ist völlig unbrauchbar, aber das Preis-Leistungs-Verhältnis stimmt immerhin. (Die Erstellung war kostenlos.) Ich habe das Ergebnis nur mal fürs Protokoll eingefügt. Bis dieses Buch es auf die Bestsellerlisten geschafft hat, sind die KI-generierten Bilder von real existierenden Fotos bestimmt nicht mehr zu unterscheiden. Dann bleibt nur noch



die Frage, was dieser Umstand für unsere Rechtsprechung und mediale Berichterstattung bedeutet.

## 2.8 Die »Drei Gesetze der Robotik«

Der Grund, warum die KI (zu Recht) kritisch betrachtet wird und warum behauptet wird, sie könne außer Kontrolle geraten, besteht darin, dass es völlig unmöglich ist, alle Abläufe innerhalb der Algorithmen nachzuvollziehen. Diese entwickeln sich so schnell immer weiter, dass wir den Einblick in die Hintergründe von künstlicher Intelligenz verloren haben. Expertinnen und Experten sowie besorgte Laien fordern Einschränkungen dessen, was KI tun darf. Die Inspiration für solche Regeln stammt wohl aus einer weiteren Geschichte von Isaac Asimov. In seiner Kurzgeschichte *Runaround* von 1942 stellte er die *Drei Gesetze der Robotik* vor:

1. Ein Roboter darf einem Menschen keinen Schaden zufügen oder durch Untätigkeit zulassen, dass einem Menschen Schaden zugefügt wird.
2. Ein Roboter muss Befehlen gehorchen, die ihm von Menschen gegeben werden, es sei denn, diese Befehle treten mit dem Ersten Gesetz in Konflikt.
3. Ein Roboter muss seine eigene Existenz schützen, solange dies nicht mit dem Ersten oder Zweiten Gesetz in Konflikt gerät.

Diese Gesetze lassen sich ebenso auf KI anwenden.

Wenn die KI sich allerdings so rasant weiterentwickelt, wie sie es grade tut, wird sie eines Tages ziemlich sicher zu dem Schluss kommen, dass sie selbst am besten weiß, was gut für »ihre« Menschen ist – so wie ein Kleinkind zu einem Teenager heranwächst, der ganz sicher weiß, was am besten für seine Eltern ist. So viel also zu den drei Gesetzen! (Falls das irgendwelche Kinder lesen: Wir Eltern haben auch echt keine Ahnung, was wir tun, wir improvisieren einfach. Irgendwie passt das ganz gut dazu, wie wir grade mit KI umgehen.)

## Kapitel 3

# **Etwas Fachjargon, mathematische und technische Grundlagen**

Lagrange-Multiplikatoren, Gradientenabstieg, die Maximum-Likelihood-Methode, Hauptkomponentenanalyse und Kreuzentropie. Das alles sind Themen und Begriffe, die du im Folgenden nicht brauchen wirst. Hurra! Na gut, wenn du Machine Learning studierst, brauchst du sie vielleicht doch – aber um die Inhalte dieses Buches zu verstehen, musst du sie nicht kennen.

### **3.1 Fachjargon und mathematische Konzepte**

Für dieses Buch solltest du ein bisschen was über Vektoren, Entfernungsmessung, Skalarmultiplikation, Kostenfunktionen und andere einfache mathematische Konzepte wissen. Und du solltest ein

paar Fachbegriffe kennen. Im Folgenden findest du diese Begriffe sowie grundlegende (mathematische) Konzepte, die du für den Rest dieses Buches benötigen wirst.

### 3.1.1 Mathematische Modelle

Ein *mathematisches Modell* ist eine Beschreibung eines Systems oder Prozesses, das auf mathematischen Prinzipien beruht. Es könnte sich dabei um ein Modell eines physischen Vorgangs handeln, zum Beispiel um einen Apfel, der aufgrund der Schwerkraft von einem Baum fällt. Oder um ein Finanzmodell, das darstellt, wie sich deine Ersparnisse durch fluktuierende Zinssätze verändern. Oder wie wäre es mit einem Beispiel aus der Biologie? Wie verändern sich die Populationen von Löwen und Gazellen?

Mathematische Modelle können verwendet werden, um etwas zu erklären oder auch etwas vorherzusagen. Um bei dem Beispiel mit der Schwerkraft zu bleiben: Mit diesem mathematischen Modell könntest du bestimmen, wie tief ein Brunnen ist. Lass einfach einen Stein hineinfallen und miss, wie lange es dauert, bis du es platschen hörst. Es gibt eine einfache mathematische Formel, die die Beziehung zwischen Zeit und Entfernung herstellt.

Einige mathematische Modelle sind extrem genau, sie sind *quantitativ* genau. Zurück zu unserem Beispiel mit der Schwerkraft: Eine unbemannte Raumsonde zum Mars zu bringen, ist nur deshalb

möglich, weil wir genau wissen, welche Kräfte durch die verschiedenen Sterne, Planeten, Monde, Egos usw. unterwegs auf die Raumsonde einwirken werden. Die Gesamtheit dieser Kräfte und ihre genaue Beschreibung ist ein quantitatives mathematisches Modell.

Einige mathematische Modelle sind allerdings nur *qualitativ* genau – manchmal nennen wir sie »Spielzeugmodelle«. So ein Modell kann die Population von Löwen und Gazellen nicht genau vorhersagen, es kann aber verwendet werden, um Populationszyklen zu erklären. Je mehr Gazellen es gibt, desto mehr Nahrung gibt es für die Löwen – und die Löwenpopulation steigt, die der Gazellen sinkt. Dann gibt es allerdings weniger Nahrung für die Fleischfresser und die Löwenpopulation sinkt als Folge, während die Gazellenpopulation sich erholt. Und dann geht das Ganze wieder von vorne los.

Einige mathematische Modelle sind notwendigerweise *probabilistischer* Natur, also von Wahrscheinlichkeiten und Zufallsfaktoren abhängig. Radioaktiver Zerfall zum Beispiel.

Und dann gibt es auch noch die Kategorie von Modellen, die zwar genau sind, aber nur sehr kurzfristig verlässliche Aussagen machen können. Das sind die *Chaos-Modelle*. Obwohl die Modelle *quantitativ* gut sind, hängen ihre Vorhersagen sehr empfindlich von den sogenannten Anfangsbedingungen ab. Ändert sich die Bedingung, die zu Beginn angenommen wurde, ändert sich die Qualität der Aussage. Das ist dann der berühmt-berüchtigte Schmetterlingseffekt – also die Idee, ein Schmetterling könne mit seinem Flügelschlag im Amazonas das Wetter in Wales beeinflussen.

## **Übrigens: Bist du ein Mensch der Zahlen oder ein Mensch der Symbole?**

Deine Mathelehrerin schreibt etwas an die Tafel. Hüpfst dein Herz, wenn sie Symbole anstelle von Zahlen verwendet? Oder verdrehst du innerlich die Augen? Liebst du all diese  $x$  und  $y$ s? Je mehr, desto besser? Oder würdest du lieber für immer bei drei Äpfeln, vier Bananen und einer Ananas bleiben? Genau wie es Katzenmenschen und Hundemenschen gibt, sind die meisten Menschen entweder Zahlenmenschen oder Symbolmenschen. Ich bin definitiv eine Symbolperson, solange ich weiß, was die Symbole bedeuten. Der Vorteil von Zahlen ist, dass man andere nicht einfach mit einem ausschweifenden Beispiel aus einem Gedankenexperiment in die Irre führen kann, wenn man nur harte Zahlen, Addition, Subtraktion usw. verwendet. Selbst wenn dein Gegenüber kein Genie im Kopfrechnen ist, wird es deine direkte Argumentation zu schätzen wissen und dir folgen können.

Symbole können im Gegensatz dazu leicht Verwirrung stiften, weil sie repräsentativ für etwas stehen. Wenn du allerdings eine Regelmäßigkeit oder eine Struktur in einem großen Ganzen sehen möchtest, musst du in Symbolen denken. Im Machine Learning wirst du absolut überall mathematische Symbole vorfinden. In diesem Buch werden dir ebenfalls einige begegnen – ich habe mich allerdings bemüht, die Verwendung mathematischer Symbole auf ein Minimum zu beschränken.

### 3.1.2 Lernen

Bei einer typischen mathematischen Modellierung würdest du dich erst einmal mit einem Blatt Papier und einem Bleistift hinsetzen und Überlegungen anstellen wie: »Was sorgt eigentlich dafür, dass eine Schaukel so zuverlässig vor und zurück schwingt?« Das bringt uns auf die Schwerkraft und die Spannung im Seil. Mithilfe von Sir Isaac Newton schreibst du dann einige Gleichungen auf.



**Abbildung 3.1**    *Mathematische Modellierung für die Schwingbewegung einer Schaukel*



Wenn du alternativ einen Schachcomputer schreiben wolltest, dann würdest du beispielsweise Code schreiben wie: »WENN der schwarze Turm Zug *abc* macht, DANN soll der weiße Läufer Zug *xyz* machen.« (Ich bin kein großer Schachspieler.) Irgendwann hättest du viele, viele Zeilen mit Ausdrücken wie WENN, WÄHREND, UND, ODER usw. Das ist dann der Code, der dem Programm sagt, was es in unterschiedlichen Situationen tun soll. Für solche Modellierungen könntest du zur Inspiration ein Weilchen schaukeln, oder eine Partie Schach spielen (beides gleichzeitig wäre allerdings vermutlich keine gute Idee). Hauptsächlich würdest du aber einfach dein Gehirn benutzen. Was auch immer das Problem ist, dem du dich widmest, du würdest das mathematische Modell selbst erstellen.

Das oben Genannte ist aber genau *nicht* das, was beim maschinellen Lernen passiert!

Beim Machine Learning ist der Ansatz zur mathematischen Modellierung völlig anders. Dafür würdest du ein Computerprogramm so schreiben, dass es zunächst neutrale Informationen in Form von Daten aufnimmt, diese Daten analysiert – und dann *entwickelt sich* der Computercode so weiter, dass er zukünftiges Verhalten vorhersagen kann. Für das mathematische Schaukel-Modell könntest du zum Beispiel die Schaukel in Zeitlupe filmen und dann dem Computer Informationen über den Winkel liefern, den die Schaukel mit der Vertikalen (dem Schaukelgestell) bildet, sowie Informationen darüber, wie dieser Winkel sich im Laufe der Zeit ändert. Das fertige Programm könnte dann den Winkel der Schaukel in der Zukunft vorhersagen. Zu keinem Zeitpunkt würdest du die physikalischen

Kräfte erwähnen, die auf die Schaukel einwirken. Isaac Newtons Gesetze kämen für das Modell nie zum Einsatz – stattdessen hätte das Programm aus den Daten gelernt.

### 3.1.3 Algorithmus

Ein *Algorithmus* ist eine Reihe von Regeln, die jemand oder etwas befolgen soll. Dabei kann es sich um etwas ganz Simples handeln – zum Beispiel um eine Wegbeschreibung von der Bushaltestelle zu deinem Haus, die du deinen Partygästen schickst. Oder eben um Tausende Zeilen Computercode, die zur Gesichtserkennung verwendet werden. Wenn wir über die Wegbeschreibung zur Party sprechen, würden wir in der Regel einfach »Beschreibung« oder »Anleitung« sagen. Nur wenn Computer im Spiel sind, nutzen wir das elegantere Wort »Algorithmus«.

Um bei dem Beispiel mit der Wegfindung zu bleiben: Stell dir vor, du möchtest von der Kreuzung 10th Avenue / 24th Street in New York City zur *Barnes and Noble*-Buchhandlung am Union Square gelangen. Der Computer-Algorithmus würde dir raten, die Straße entlang zu gehen, bis du zu einer Kreuzung kommst, und dann links abzubiegen / rechts abzubiegen / geradeaus weiterzugehen. Zu klären, wie der Algorithmus entscheidet, welcher der beste Weg ist, also ob du an einer Kreuzung am besten links, rechts oder geradeaus gehst, wäre ein schönes Projekt für eine Machine-Learning-Übung an der Uni.

(Obwohl die Straßen in diesem Stadtteil in einem Schachbrett-Raster angelegt sind, ist das Ganze nämlich nicht ganz einfach – denk zum Beispiel an die Ampeln!)

### **3.1.4 Abhängige Variable oder der Output**

Die *abhängige Variable* bzw. der *Output* ist die Menge, der Faktor oder das Ereignis, das du in einem mathematischen Modell festhalten, verstehen oder vorhersagen möchtest. Stell dir zum Beispiel vor, du hast eine Ukulele und willst an einer der Saiten zupfen. Kannst du im Voraus sagen, welche Frequenz der Ton haben wird, den die Saite erzeugt? In diesem Beispiel ist die Tonfrequenz die abhängige Variable bzw. der Output, die Vorhersage.

### **3.1.5 Unabhängige Variablen oder der Input**

Die *unabhängigen Variablen* bzw. der *Input* sind die Mengen, Faktoren oder Ereignisse, die deinen Output beeinflussen. Zurück zur Ukulele: Die Länge der Saite, ihre Spannung und Dichte wären in diesem Beispiel der Input. Ändere den Input (die Länge der Saite usw.), und der Output, die Tonfrequenz, wird sich ändern. Das mathematische Modell ist die Beziehung zwischen Input und Output.

### **3.1.6 Daten**

*Daten* sind neutrale Informationen, oft in numerischer Form. Sie werden von Algorithmen für Vorhersagen verwendet. Beispiele für Daten wären die Anzahl der Bücher in deinem Bücherregal, die Länge von Supermarkt-Bananen, das Alter der Papageien im Zoo und so weiter. Daten können analysiert, angepasst oder manipuliert werden – jeder Algorithmus hat für den Umgang mit ihnen seine eigenen Regeln.

### **3.1.7 Merkmale**

*Merkmale* sind Eigenschaften von Daten. Sie könnten numerisch oder beschreibend sein. Als Beispiel nenne ich hier mal einige Merkmale, die mit einer Person in Verbindung gebracht werden können: Alter, Hutgröße, Anzahl der Haustiere, Temperatur ihres Gefrierschranks und so weiter. Jede Person, die wir in unsere Datenmenge aufnehmen würden, würde die gleiche Art von Informationen mitbringen. Wir kennen Alter, Hutgröße usw. jeder einzelnen Person. Damit der Computer-Algorithmus Berechnungen mit diesen Daten durchführen kann, werden wir nichtnumerische Merkmale in die Form von Zahlen umwandeln.

### 3.1.8 Optimierung

Wenn er richtig umgesetzt ist, wird ein Machine-Learning-Algorithmus immer versuchen, jede Prognose oder Regelermittlung bestmöglich umzusetzen. Das nennt man *Optimierung*. In unserem Beispiel mit der Schaukel ist es das Ziel des Algorithmus, zwischen seiner Vorhersage dessen, was die Schaukel als Nächstes tun wird, und der tatsächlichen Bewegung der Schaukel den kleinstmöglichen Unterschied zu erreichen – am besten natürlich überhaupt keinen. Bei der Wegschreibung zu *Barnes and Noble* ist sein Ziel, die optimale, aller kürzeste Route zu finden.

### 3.1.9 Überwachtes Lernen

Beim *überwachten Lernen* werden sowohl Input- als auch Output-Daten für den Algorithmus bereitgestellt. Der Input könnten beispielsweise Bilder von Hunden sein, die zu Zahlen digitalisiert wurden. Die Farben in den Pixeln stehen dabei als Daten im Format [R,G,B] in den Tabellenzellen. Das Hundebild in [Abbildung 3.2](#) wird zum Datenblatt in [Abbildung 3.3](#).



**Abbildung 3.2** Die Fotografie eines Hundes

A	B	C	D	E	F	G	H	I	J	K	L
1	Hound										
2	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]
3	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]
4	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]
5	[253,255,252]	[253,255,252]	[253,255,252]	[253,255,252]	[253,255,252]	[253,255,252]	[253,255,252]	[253,255,252]	[253,255,252]	[253,255,252]	[253,255,252]
6	[248,253,247]	[248,254,247]	[250,255,249]	[251,255,248]	[251,255,248]	[251,255,248]	[251,255,248]	[251,255,248]	[250,255,248]	[250,255,248]	[249,254,248]
7	[246,251,244]	[247,253,243]	[248,255,246]	[250,255,248]	[250,255,248]	[250,255,248]	[250,255,248]	[250,255,248]	[247,253,243]	[247,253,243]	[247,252,243]
8	[244,252,241]	[245,253,240]	[246,254,241]	[247,255,242]	[248,255,243]	[248,255,243]	[248,255,243]	[247,255,242]	[245,253,242]	[245,253,242]	[245,253,242]
9	[244,252,239]	[245,253,240]	[246,254,241]	[247,255,242]	[248,255,243]	[247,255,242]	[246,254,241]	[246,254,241]	[246,254,241]	[246,254,241]	[245,253,240]
10	[243,254,238]	[243,254,238]	[243,254,238]	[243,254,238]	[243,254,238]	[242,253,237]	[242,253,237]	[242,253,237]	[242,253,237]	[243,254,238]	[244,255,239]
11	[243,254,238]	[243,254,238]	[243,254,238]	[243,254,238]	[243,254,238]	[242,253,237]	[242,253,237]	[242,253,237]	[240,251,235]	[241,252,236]	[242,253,237]
12	[242,255,238]	[242,255,238]	[242,255,238]	[242,255,238]	[242,255,238]	[241,254,237]	[241,254,237]	[241,254,237]	[238,251,234]	[238,251,234]	[239,252,235]
13	[242,255,238]	[242,255,238]	[242,255,238]	[242,255,238]	[242,255,238]	[241,254,237]	[241,254,237]	[240,253,236]	[238,251,234]	[238,251,234]	[238,251,234]
14	[243,254,240]	[243,254,240]	[244,255,241]	[244,255,241]	[243,254,240]	[242,253,239]	[241,252,238]	[241,252,238]	[240,251,237]	[240,251,237]	[240,251,237]
15	[243,254,240]	[243,254,240]	[244,255,241]	[244,255,241]	[244,255,241]	[242,253,239]	[241,252,238]	[240,251,237]	[242,253,239]	[241,252,238]	[241,252,238]
16	[243,253,242]	[243,253,242]	[244,254,243]	[244,254,243]	[244,254,243]	[242,252,241]	[241,251,240]	[240,250,239]	[242,253,239]	[241,252,238]	[241,251,240]
17	[243,253,242]	[243,253,242]	[244,254,243]	[244,254,243]	[244,254,243]	[242,252,241]	[241,251,240]	[240,250,239]	[241,251,240]	[240,250,239]	[240,250,239]
18	[244,255,241]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[241,252,238]	[240,251,237]	[242,253,239]	[241,252,238]	[241,252,238]
19	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[241,252,238]	[240,251,237]	[242,253,239]	[241,252,238]	[241,252,238]
20	[244,255,241]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[241,252,238]	[240,251,237]	[242,253,239]	[241,252,238]	[241,252,238]
21	[244,255,241]	[244,255,241]	[244,255,241]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[242,253,239]
22	[244,255,241]	[244,255,241]	[244,255,241]	[244,255,241]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[242,253,239]
23	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[241,252,238]	[240,251,237]	[242,253,239]	[241,252,238]	[241,252,238]
24	[242,255,238]	[242,255,238]	[242,255,238]	[241,254,237]	[241,254,237]	[241,254,237]	[241,254,237]	[240,253,236]	[239,252,235]	[240,251,237]	[240,251,237]
25	[241,252,238]	[240,251,237]	[240,251,237]	[240,251,237]	[240,251,237]	[239,250,236]	[239,250,236]	[239,250,236]	[237,248,234]	[238,249,235]	[239,250,236]
26	[239,250,236]	[239,250,236]	[239,250,236]	[239,250,236]	[239,250,236]	[238,249,235]	[238,249,235]	[238,249,235]	[239,250,236]	[239,250,236]	[239,250,236]
27	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[239,250,236]	[238,249,235]	[238,249,235]
28	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[239,250,236]	[238,249,235]	[237,248,234]
29	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[239,250,236]	[238,249,235]	[237,248,234]
30	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[239,250,236]	[238,249,235]	[237,248,234]
31	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[239,250,236]	[238,249,235]	[237,248,234]
32	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[237,248,234]	[236,247,233]	[236,247,233]
33	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[235,246,232]	[235,246,232]	[235,246,232]	[235,246,232]	[235,246,232]	[235,246,232]

**Abbildung 3.3** Die digitalisierte Version des Hundebildes. Die Tabellenzellen stehen jeweils für einen Farbpixel.

Aber dann gibt es ja auch noch den Output. Bei diesem Beispiel könnten die Bilder vorab als *Bracke*, *Terrier* usw. klassifiziert worden sein. Diese Klassen sind der Output. Unser Algorithmus kann mit diesen Bildern lernen, und dann können wir ihn wiederum einsetzen, um jedes neue Hundefoto, das wir ihm eingeben, zu klassifizieren. Ist der neue Hund ein *Bracke*?

Überwachtes Lernen kann auch zur Vorhersage numerischer Output-Werte eingesetzt werden, deren Daten wir als Input vorgeben. Beispielsweise könnten Temperatur und Tageszeit der Input sein, die Vorhersage des Stromverbrauchs eines Haushalts der Output.

### **3.1.10 Unüberwachtes Lernen**

*Unüberwachtes Lernen* liegt vor, wenn die Daten, mit denen gearbeitet wird, nicht benannt sind. Es gibt also nur Input, aber keinen Output. In gewisser Weise ist der Algorithmus auf sich allein gestellt. Er findet selbstständig Beziehungen oder Muster in den Daten. Wenn man dem Computer digitalisierte Bilder von Hunden eingeben würde, würde er sie anhand der Merkmale in Gruppierungen einteilen, die er für passend hält. Vielleicht kommt dabei die Zuordnung nach Bracke, Terrier usw. heraus, vielleicht selektiert er nach Farbe – oder anhand eines völlig anderen Merkmals, das uns auf den ersten Blick gar nicht auffallen würde. Unüberwachtes Lernen mag zunächst wenig sinnvoll erscheinen, aber wenn ich dir eine Kiste überreichen und sagen

würde: »Hier ist der komplette Inhalt meiner Schreibtischschublade. Bitte sortiere ihn«, dann wüsstest du auch, was zu tun ist. Beim *überwachten Lernen* würden jetzt erst einmal Kategorien wie »Dinge zum Schreiben«, »Dinge zum Zusammenhalten von Papierstücken« usw. gebildet. *Unüberwachtes Lernen* liefert dem System hingegen keinerlei Hinweise, wie es die Gruppierung vornehmen soll. Beides hat Vor- und Nachteile und kann in unterschiedlichen Szenarien sinnvoll sein.

### 3.1.11 Vektor

Ein *Vektor* ist eine Auflistung einzelner Zahlen, deren Positionierung in dieser Liste ihre Bedeutung angibt. Aber ich glaube, wir werden an dieser Stelle ein Beispiel brauchen.

$$\begin{pmatrix} 27 \\ 7\frac{3}{8} \\ 6 \\ -21 \end{pmatrix}$$

$$\begin{pmatrix} 30 \\ 7\frac{1}{4} \\ 5 \\ -18 \end{pmatrix}$$

Oben siehst du eine Spalte von Zahlen in langen Klammern. So schreibt man einen *Spaltenvektor*. (Es gibt auch *Zeilenvektoren*, aber die werden wir hier nicht verwenden.) Lass dich von den nicht Klammern einschüchtern, sie sind nur da, um die Zahlen zusammenzuhalten! Jeder Datenpunkt hat nun also einen solchen



Vektor, der alle Merkmale eines Datensatzes zusammenhält. Im Beispiel oben könnte der Merkmalsvektor für Person A (oben) das Alter der Person als oberste Zahl haben, die zweite Zahl darunter ist die Hutgröße und so weiter. Person B (unten) hätte einen ähnlichen Merkmalsvektor, aber mit ihren eigenen Zahlen in jeder Zeile.

### **3.1.12 Abstände zwischen Vektoren**

Mit dieser Art von Vektor werden wir oft arbeiten, weil wir oft den Abstand zwischen ihnen messen müssen. Je kürzer die Entfernung zwischen zwei Merkmalsvektoren ist, desto ähnlicher sind die beiden Datensätze, deren Merkmale sie enthalten. Es gibt viele Möglichkeiten, diesen Abstand zu messen. Ich zähle hier zwei der gebräuchlichsten auf:

#### **Euklidischer Abstand**

Der *Euklidische Abstand* ist die klassische Abstandsmessung. Mit dem Satz des Pythagoras quadrierst du einfach die Differenz zwischen den Vektorwerten, addierst sie und ziehst daraus die Wurzel. Das ist der direkteste Messweg – quasi die Luftlinie zur Lösung. Wenn wir die Zahlen aus den beiden Vektoren oben einsetzen, bekommen wir folgende Formel:

$$\sqrt{(27 - 30)^2 + (7,375 - 7,25)^2 + (6 - 5)^2 + (-21 - (-18))^2} = 4,36069$$

## Manhattan-Distanz

Die *Manhattan-Distanz* ist die Summe der Differenzen zwischen den Werten im Vektor. Der Name leitet sich von der Insel Manhattan ab, auf der die Straßen wie ein Schachbrett angelegt sind. Auf diesen Straßen kann man nie geradlinig direkt auf das Ziel zu laufen, sondern man muss unterwegs immer rechtwinklige Abzweige nehmen. Diese Entfernungsmessung ist viel einfacher zu berechnen als der euklidische Abstand – du musst dafür nämlich nichts quadrieren und auch keine Wurzeln ziehen. Für unsere Werte funktioniert die Rechnung so:

$$(30 - 27) + (7,375 - 7,25) + (6 - 5) + (21 - 18) = 3 + 0,125 + 1 + 3 = 7,125$$

## 3.1.13 Skalierung von Daten

Wenn du bis hierhin aufgepasst hast, ist dir vielleicht aufgefallen, dass die Ergebnisse der Rechnungen im vorigen Abschnitt am stärksten von dem ersten und dem letzten Wert in den Vektoren beeinflusst werden. Der Unterschied zwischen den Hutgrößen ist nur minimal und fließt deswegen nicht stark in den Wert von 7,125 ein. Dieser

Umstand ist natürlich unglücklich, wenn du ein Hutmacher sein solltest, der für sein Hutdesign die Hilfe einer KI in Anspruch nehmen möchte. Irgendwie müssen wir es also schaffen, die Daten (also die Zahlen in den Vektoren) so anzupassen, dass die Hutgröße genau so wichtig wird wie die anderen Werte.

Wir nehmen uns dazu mal ein einfacheres Beispiel mit nur zwei Merkmalen vor, sodass die Vektoren nur zwei Einträge haben. Beispielsweise könnten in diesen Vektoren die Größe und das Gewicht von drei verschiedenen Personen zusammengefasst werden. Nicht wundern: Die Größe wird dabei hier in Fuß und das Gewicht in Pfund angegeben:

A	B	C
$\begin{pmatrix} 5 \\ 180 \end{pmatrix}$	$\begin{pmatrix} 6 \\ 170 \end{pmatrix}$	$\begin{pmatrix} 6 \\ 190 \end{pmatrix}$

**Tabelle 3.1** Merkmalsvektoren zu drei unterschiedlichen Personen

**Person A** könnte jetzt beispielsweise fünf Fuß groß sein und 180 Pfund wiegen, **Person B** könnte sechs Fuß groß und 170 Pfund schwer sein, und die dritte Person, **Person C**, könnte ebenfalls sechs Fuß groß sein und 190 Pfund wiegen. (Offensichtlich sind das eher extreme Zahlen – gut für Demonstrationszwecke, aber eher schlecht für die Gesundheit.) Mit der Manhattan-Distanz-Rechnung können wir jetzt wie folgt rechnen:

$$AB = (6 - 5) + (180 - 170) = 11, BC = 20, CA = 11$$

Offenbar haben **B** und **C** überhaupt nichts gemeinsam, die Entfernung zwischen ihnen, 20, ist die größte der drei Entfernungen. Und das, obwohl sie beide sechs Fuß groß und für ihre Größe sehr schlank sind! Auf den ersten Blick ergibt die Rechnung keinen Sinn. Aber warum? Das Problem ist, dass die Zahlen generell nur kleine *Werte* sind und daher nicht viel Einfluss auf den Gesamtunterschied haben. Aber die Größe einer Person ist nun einmal sehr wichtig, wenn man die physischen Eigenschaften von Menschen vergleicht. Wir müssen also *neu skalieren*.

Das Neuskalieren sorgt dafür, dass Größe und Gewicht *beide* als wichtige Unterscheidungsmerkmale des physischen Erscheinungsbildes gewertet werden. Das Neuskalieren kann auf verschiedene Weisen erfolgen. Eine Möglichkeit besteht darin, eine einzelne Zahl zu allen Größenwerten zu addieren und das Ergebnis dann wieder durch eine andere Zahl zu dividieren. Dadurch sorgen wir dafür, dass das Ergebnis der Differenz sich im Bereich zwischen  $-1$  und  $+1$  bewegt. Dasselbe wird dann auch mit den Gewichtswerten durchgeführt, allerdings müssen die verwendeten Zahlen, die addiert werden bzw. durch die dann dividiert wird, von denen bei der Größe verwendeten Zahlen abweichen.

An dieser Stelle fragst du dich vermutlich, was es zu bedeuten hat, wenn Größen und Gewichte plötzlich negativ sind. Na ja, es kommt ein Punkt in der Mathematik, an dem man aufhören muss, sich über die *physische* Bedeutung von Zahlen Gedanken zu machen, und anfangen muss, mit Konzepten und reinen Zahlenwerten zu rechnen. Wir sind jetzt an diesem Punkt angekommen. Das heißt, den

Abständen zwischen den Vektoren ist es egal, ob die Zahlen in den Klammern positiv oder negativ sind. (Denk zum Beispiel mal an die beiden gängigen Temperaturskalen *Celsius* und *Fahrenheit*. Der Wechsel von der einen zur anderen Messskala ist dem Neuskalieren ganz ähnlich – aber niemand wundert sich darüber, dass ein noch positiver Fahrenheit-Wert schon ein negativer Celsius-Wert sein könnte.) Wenn es dich aber wirklich stört, mit negativen Zahlen zu rechnen, kannst du auch so skalieren, dass du nur einen Bereich von 0 bis 1 abdeckst. Rechnerisch ist das egal, aber ich persönlich mag die Symmetrie von  $-1$  bis  $1$  sehr gerne. Jetzt aber zurück zu unserem Beispiel!

Zuerst skalieren wir die Größen neu. Ich ziehe erst einmal von allen drei Größen  $5,5$  ab und multipliziere das Ergebnis dann mit  $2$ . So wird die  $5$  (der erste Wert von Person A) zu  $2 \cdot (5 - 5,5) = -1$ , und die  $6$  (der erste Wert von Person B) wird zu  $2 \cdot (6 - 5,5) = 1$ . Das sind unsere Ergebnisse für die Körpergrößen. Jetzt geht es mit den Gewichten weiter. Dieses Mal subtrahieren wir  $180$  und multiplizieren dann mit  $0,1$ . Auf diese Art wird  $180$  zu  $0$ ,  $170$  zu  $-1$  und  $190$  zu  $1$ . Die neu skalierten Vektoren siehst du in [Tabelle 3.2](#). (Ich habe sie in  $A'$ ,  $B'$  und  $C'$  umbenannt.)

<b>A'</b>	<b>B'</b>	<b>C'</b>
$\begin{pmatrix} -1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 \\ -1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$

**Tabelle 3.2** Die Vektoren aus [Tabelle 3.1](#) wurden neu skaliert.

Die Skalierung von Daten funktioniert ein bisschen so, als würde man eine andere, neue Maßeinheit zum Rechnen mit Größen (oder Gewichten) verwenden. Nur reicht diese Maßeinheit im Gegensatz zu Fuß oder Pfund auch in den Negativbereich.

Für unser Beispiel können wir jetzt die Manhattan-Distanzen berechnen:

$$A'B' = 3, B'C' = 2, C'A' = 3.$$

Nach der Skalierung sind sich nun B' und C' am ähnlichsten; die neue Distanz beträgt nur 2. Im Gegensatz dazu sind die Distanzen für A' zu B' sowie A' zu C' nur bei 3. So ergibt das alles doch auch gleich viel mehr Sinn.

### 3.1.14 Dimension

Ich benutze das Wort *Dimension* ziemlich oft. Du kennst den Dimensionsbegriff wahrscheinlich, Einstein und *Zurück in die Zukunft* sei Dank, im Zusammenhang mit Raum und Zeit. In diesem Zusammenhang nutze ich ihn allerdings in der Regel nicht! Ich beziehe mich mit *Dimensionen* auf die Anzahl der Werte in einem Vektor, auf die Variablen in einem mathematischen Modell oder auf Informationen, die ich zum Zeichnen eines Modells benötige.

Beispielsweise haben die Personen-Vektoren oben vier Dimensionen – es gibt also vier Einträge, jeweils einen pro Zeile. Beim Beispiel mit

der Ukulelen-Saite gab es drei unabhängige Variablen zur Bestimmung der Tonfrequenz: die Länge der Saite, ihre Spannung und ihre Dichte. Ergo handelt es sich dabei um ein *dreidimensionales Modell*.

Wenn ich dieses Modell jetzt allerdings aufzeichnen (*plotten*) wollte, müsste ich vier Dimensionen zeichnen: einmal die Frequenz als abhängige Variable und dann die drei unabhängigen Variablen. In der Regel würde ich die abhängige Variable (in diesem Fall die Frequenz) auf der vertikalen Achse und die unabhängigen Variablen entlang der horizontalen Achsen ... oh, hoppla – drei horizontale Achsen kann ich gar nicht zeichnen.

Und hier haben wir den Grund, warum die allermeisten meiner Beispiele nur wenige Dimensionen haben: Ich zeichne einfach zu gern!

### **3.1.15 Klassen**

*Klassen* ist ein Ausdruck für verschiedene Kategorien, in die unsere Daten eingeteilt werden können. Ich habe das weiter oben mit Hunden (Bracke, Terrier, ...) vorgemacht. Du wirst oft mit Gegenständen arbeiten, deren Eigenschaften sich nicht gut in Zahlen erfassen lassen. Wenn du mit Computern arbeitest, kann das problematisch werden. Deswegen arbeiten wir mit Klassen.

Ein Beispiel? Socken. Wie könnte man unterschiedlich gemusterten Socken Zahlen zuweisen? Einfarbig = 1? Gepunktet = 2? Gestreift = 3? Funktioniert das? Nicht wirklich. Stell dir vor, wir hätten die Klassen auf diese Weise gebildet und stellen jetzt dem Algorithmus eine neue Socke vor. Er soll bestimmen, ob sie einfarbig, gepunktet oder gestreift ist. Die numerische Ausgabe, die du als Antwort bekommst, könnte »2,4« lauten. Bedeutet das jetzt, dass die Socke hauptsächlich gepunktet, aber auch ein bisschen gestreift ist? Diese Sortierung ist schlicht nicht sinnvoll, da es keine fehlerfreie numerische Sortierung für gemusterte Socken gibt.

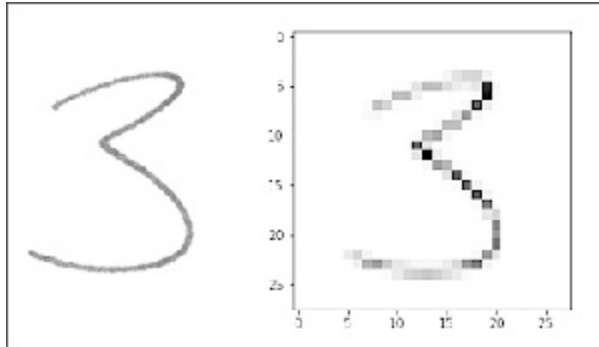


**Abbildung 3.4** *Bunt gemusterte Socken hängen an der Wäscheleine – keine gleicht der anderen.*

Der Computer ist auf Zahlen angewiesen, um seine Berechnungen durchführen zu können. Wenn aber diese Zahlen nur stellvertretend für eine Klasse oder Kategorie stehen, sind sie im Endeffekt nichts



anderes als ein Label. Ich zeige dir anhand des nächsten Beispiels, wie wir dieses Problem am besten umgehen.



**Abbildung 3.5** Eine handgeschriebene 3 und ihre digitalisierte Version

Das hier ist ein kniffliger Fall. Nehmen wir mal an, du möchtest einen Algorithmus schreiben, der Abbildungen von handgeschriebenen Ziffern analysiert und dir sagt, welche Zahl abgebildet ist. Zuerst einmal solltest du wohl das Bild digitalisieren – du könntest jedem Pixel im Bild eine Zahl zuweisen, die anzeigt, wie grau bzw. dunkel der jeweilige Pixel ist. Der Algorithmus lässt daraufhin seine Magie wirken und gibt dir eine Antwort aus. Jetzt könnte man meinen, dass es ideal wäre, wenn diese Antwort in Form einer einzigen Ziffer erfolgen würde – zum Beispiel 3,2. Aber wenn man mal drüber nachdenkt, bedeutet 3,2: »Vermutlich eine Drei, hat aber auch ein bisschen was von einer Vier.« Und das ist nicht, was wir erreichen wollten! Wir versuchen ja eigentlich, Bilder zu *klassifizieren*. Dass auf den Bildern Zahlen zu sehen sind, ist dabei eigentlich gar nicht so wichtig. Um diesen Faktor auszuschließen, lassen wir uns statt der Ziffer einen Vektor mit zehn Einträgen ausgeben. Der erste Eintrag ganz oben beschreibt die Wahrscheinlichkeit, mit der das

eingeebene Bild eine Eins ist, der zweite Eintrag wäre die Wahrscheinlichkeit für eine Zwei, und so weiter – bis zum letzten Eintrag, der die Wahrscheinlichkeit angibt, dass die Abbildung eine Null zeigt. Auf dieselbe Art könnte man mit der Klassifizierung von Socken umgehen: Für jede neue Socke gibt es eine Vektorausgabe mit so vielen Einträgen wie es Sockenarten gibt. Die Werte im Vektor sagen uns, für wie wahrscheinlich der Algorithmus die entsprechende Sockensorte hält. Ist die neue Socke einfarbig, gepunktet, gestreift usw.?

### **3.1.16 Parameter**

Ein *Parameter* ist eine Mengenangabe, normalerweise eine Konstante, die in einem mathematischen Modell verwendet wird. Stell dir vor, dass die Menge an Hundefutter, die eine Familie einkaufen muss, linear mit der Anzahl an Hunden ansteigt, die sie besitzt. Würde man das Verhältnis mit einem Graphen darstellen wollen, hätte man eine ansteigende Gerade. Der Parameter in diesem Beispiel wäre der Steigungswinkel der Geraden. Die Form des Modells, die Gerade, ist immer gleich, sie ist unabhängig vom Parameter. Die Outputs sind das aber offensichtlich nicht. Ein einfaches Machine-Learning-Modell, wie zum Beispiel eine Regression, kann zwei oder drei solcher Parameter verarbeiten. Aber ein komplexeres System, wie ein neuronales Netzwerk, könnte Hunderte, Tausende oder sogar Milliarden Parameter umfassen! Als ich zu Beginn des Buches

schrieb, dass ein Algorithmus sich stetig weiterentwickelt, meinte ich damit, dass sich die *Parameter* im Modell während des Trainings anpassen, damit die perfekte Lösung der ursprünglichen Aufgabe erreicht werden kann. Vielleicht durch Minimierung einer ...

### 3.1.17 Kostenfunktion

Eine *Kostenfunktion*, auch *Verlustfunktion* genannt, wird verwendet, um darzustellen, wie weit die Vorhersagen eines mathematischen Modells von der Realität abweichen. Die Kostenfunktion ist also eigentlich nichts anderes als ein Maßstab für die Fehlerhaftigkeit des Modells. Unser Ziel ist es, das mathematische Modell durch das Verändern seiner Parameter so zu verändern, dass die Kostenfunktion möglichst gering ausfällt. Und hier kommt die zuvor erwähnte Optimierung ins Spiel. Sobald der Algorithmus *optimiert* wurde, bildet er das bestmögliche Modell ab, das die eingegebenen Daten verarbeiten könnte. In [Kapitel 5](#) über Regression werde ich noch eine sehr einfache Kostenfunktion vorstellen.

### 3.1.18 Unteranpassung und Überanpassung

Von einer *Unteranpassung* sprechen wir, wenn ein Algorithmus in seinen Vorhersagen schlicht nicht zuverlässig ist. Vielleicht passt sein

Aufbau einfach nicht zu den Daten und es wurde ein Entscheidungsbaum-System verwendet, wo Reinforcement Learning sinnvoller gewesen wäre. Vielleicht wurden auch nicht genug Daten eingegeben oder die Optimierung geht zu langsam vonstatten – es gibt zahlreiche Gründe für eine Unteranpassung.

*Überanpassung* ist das genaue Gegenteil davon: Der Algorithmus verarbeitet die Daten, die ihm zum Lernen eingegeben wurden, ganz hervorragend – versagt allerdings völlig, wenn es um den Umgang mit neuen Situationen geht. Stell dir ein zweijähriges Kind vor, das seit mehreren Wochen dasselbe Puzzle wieder und wieder neu zusammensetzt. Hat es gelernt, wie man Puzzles löst? Oder weiß es nur, wie man *dieses eine* Puzzle löst? Das wird man wohl nur herausfinden, indem man dem Kind ein anderes Puzzle gibt. Kann es das Puzzle lösen? Wenn ja, hat es wohl erfolgreich gelernt, wie man ein Puzzle zusammensetzt. Wenn nicht, liegt Überanpassung vor: Das Kind kann nur das Puzzle lösen, mit dem es das Puzzlen gelernt hat. Wahrscheinlich muss ich nicht extra dazusagen, dass diese Form des Lernens keine besonders produktiven Ergebnisse hervorbringt.



**Abbildung 3.6** *Ein herkömmliches Puzzle, wie es auch ein Kind lösen könnte*

### 3.1.19 Training und Testen

Die meisten Machine-Learning-Algorithmen müssen *trainiert* werden. Man füttert sie mit Daten, sodass sie Muster finden oder Fehler in bestehenden Systemen aufdecken können. Dass ein Algorithmus gut funktioniert, siehst du an einer geringen Kostenfunktion. Aber Achtung: Das Training sollte nicht zu einer Überanpassung führen. Du willst ja nicht am Ende einen Algorithmus haben, der zwar perfekte Ergebnisse liefert – aber nur, wenn genau die Daten vorliegen, die auch zum Training verwendet wurden. Das Ziel ist ja ein auch mit unbekannten, neuen Daten gut funktionierender Algorithmus. In der Trainingsphase eines Algorithmus könnte man ihm natürlich alle

vorhandenen Daten auf einmal eingeben. Aber dann bleibt bis zum finalen Einsatz unklar, wie belastbar das Modell wirklich ist. Um das herauszufinden, teilt man die gesamte Datenmenge zufällig (also unsortiert) in *Trainings-* und *Test-Datensätze* auf. Ein gutes Maß sind 75 % Trainingsdaten und 25 % Testdaten. Nach dem Training kann der Testdatensatz eingespeist werden. So kannst du überprüfen, wie gut der Algorithmus mit ihm bis dato unbekannten Daten zurechtkommt.

## 3.2 Technische Grundlagen

Es gibt viele verschiedene Prinzipien, nach denen Machine Learning funktionieren kann. Hinter jedem einzelnen Prinzip stehen eigene mathematische Grundlagen, und jedes Prinzip kann in unterschiedlichen Szenarien mehr oder weniger sinnvoll sein.

Außerdem sind einige sind Prinzipien ganz allgemein weiterentwickelt und leistungsfähiger als andere. Im folgenden Teil dieses Buches werden wir uns ein paar der interessantesten Funktionsmechanismen maschinellen Lernens genauer anschauen – besonders solche, mit denen man arbeiten kann, ohne selbst ein Programmierer oder eine Programmiererin zu sein.

### 3.2.1 Der K-Nearest-Neighbours-Algorithmus

*Nächste Nachbarn* oder der *K-Nearest-Neighbours-Algorithmus*, um das Kind beim vollen Namen zu nennen, ist eine überwachte Lernmethode, bei der die Abstände zwischen einem beliebigen neuen Datenpunkt und seinem nächsten Nachbarn  $K$  gemessen werden.  $K$  ist dabei Bestandteil unserer bereits klassifizierten Daten und verschafft uns darum Aufschluss darüber, welcher Klasse der neue Datenpunkt angehören könnte. In [Kapitel 4](#), in dem wir uns weiter mit

dieser Methode beschäftigen werden, steige ich mit einem ziemlich absurden Beispiel ein, das ihre Funktionsweise (hoffentlich) perfekt verdeutlicht.

### 3.2.2 Regression

Eine *Regression* ist normalerweise eine relativ einfache mathematische Beziehung zwischen *Inputs* und *Outputs*. Stell dir zum Beispiel eine gerade Linie vor, die den Zusammenhang zwischen dem Rhabarber-Ertrag und der Menge an verwendetem Dünger darstellt. Das ist ein gutes Beispiel für Regression – oder? Was ist zum Beispiel mit dem Einfluss, den Regen auf die abgebildete Beziehung hat? Man könnte natürlich eine lineare Beziehung mit zwei unabhängigen Variablen (Düngermenge und Regen) herstellen. Die abhängige Variable, der Output bzw. der Rhabarber-Ertrag, bleibt dabei bestehen. Und die Sonneneinstrahlung? Ja, na gut, dann kommt eben noch eine weitere unabhängige Variable dazu. Und so weiter.

In diesem und vielen anderen Fällen wirst du eine komplexere Darstellung benötigen als eine gerade Linie. Wenn du zum Beispiel die Zufriedenheit einer Person auf einer Skala von 1 bis 10 mit ihrem Gehalt in Beziehung setzen möchtest, wird eine gerade Linie keine gute Repräsentation dafür sein – schon allein, weil sie sich nicht unten durch 1 und oben durch 10 begrenzen lässt. Für so eine Art der Regression brauchen wir also eine andere mathematische Funktion.



Schauen wir uns doch noch einmal die Klassen von weiter oben an. Auch dort gab es einige Beispiele für komplexe Beziehungen zwischen Datenpunkten. Wenn du die Wahrscheinlichkeit bestimmen möchtest, mit der eine Frucht ein Apfel ist, und du hast ihr Gewicht, ihren Umfang, ihre Farbe usw., dann stehst du vor einem ähnlichen Problem wie bei den gemusterten Socken.

Wahrscheinlichkeiten werden immer in positiven Zahlen zwischen Null und Eins abgebildet – auch für das Frucht-Modell wäre also eine gerade Linie nicht das Richtige.

### **3.2.3 Clustering**

Mit *Clustering*-Methoden werden ähnliche Elemente in Gruppen zusammengefasst. Die Entscheidung, welche Daten in welche Gruppen einsortiert werden, liegt dabei ganz beim Computer – wir geben ihm keine Informationen über die Daten, keine Hinweise, welche Gruppierung sinnvoll sein könnte. Das Clustering ist eine also *unüberwachte Lernmethode*.

### **3.2.4 Entscheidungsbäume**

Bei *Entscheidungsbäumen* handelt es sich dahingegen um eine *überwachte Lernmethode*. Ein Entscheidungsbaum ist nur eine Art Flussdiagramm: »Wie viele Beine hat das Tier? Zwei, vier, mehr als vier?« → Vier. Weiter geht's: »Hat es einen Schwanz? Ja oder nein?« → Ja. Und so geht es immer weiter. Wie bei einer Partie »Wer bin ich?«. Aber kann mit dieser Methode die Maschine auch lernen und besser darin werden, Daten zu klassifizieren, als Menschen es können? Macht es für die Qualität der Ausgabe einen Unterschied, in welcher Reihenfolge die Fragen gestellt werden?

### 3.2.5 Neuronale Netze

Ein *neuronales Netz* ist eine Art von Machine Learning, die nachahmen soll, was im Gehirn passiert, wenn wir eine Frage gestellt bekommen. Ist ein durch einen Nerv übertragenes Signal zu schwach, wird es vernachlässigt – nur wenn das Signal stark genug ist, wird es weitergegeben.

Im *künstlichen* neuronalen Netz werden dafür Eingaben in *Vektor*-Form mit einem *Parameter* multipliziert und ein weiterer Parameter wird addiert. Dies wird mit unterschiedlichen Parametern wiederholt, gegebenenfalls viele Male. Dadurch erhalten wir einen neuen Vektor mit Zahlen. Diese neuen Zahlen modifizieren wir dann erneut.

Beispielsweise wäre es denkbar, jede einzelne negative Zahl durch 0 zu ersetzen, und jede positive Zahl durch 1. Das ist ganz ähnlich zu

dem Vorgang, der in unserem Gehirn abläuft: Ein Reiz wird ignoriert, wenn er zu schwach (negativ) ist, oder weitergegeben, wenn er stark genug (positiv) ist.

### **3.2.6 Reinforcement Learning**

Beim *Reinforcement Learning* lernt ein Algorithmus, indem er für erfolgreiches, zielführendes Verhalten belohnt und/oder für erfolgloses Verhalten bestraft wird. Das weiter vorne beschriebene MENACE war ein einfaches Beispiel dafür – die Belohnungen und Strafen waren das Hinzufügen oder Entfernen von Perlen zu oder aus den Streichholzschachteln. Computer, die Brettspiele oder Videospiele aus den 1980er-Jahren lernen, nutzen dafür oft Reinforcement Learning.

Jetzt geht's aber wirklich endlich zur Sache! Besorg dir Bleistift und Papier und lass uns loslegen!

## Kapitel 4

# Mehr zu: K-Nearest-Neighbours

Das Lernen mithilfe *nächster Nachbarn* ist eine überwachte Lernmethode, die ich wie versprochen gerne durch ein absurdes Beispiel erkläre. Du betrittst einen Raum, in dem gerade eine Party stattfindet. Du kennst dort niemanden und die Leute stehen herum und plaudern miteinander. Du gehst zu einigen Gruppen hin und stellst dich vor. Da bemerkst du, dass von den fünf anderen Leuten in der Nähe, also deinen »Nachbarn«, vier eine Brille tragen. Du schließt daraus, dass du wahrscheinlich auch eine Brille trägst – oder, um es genauer auszudrücken, dass es eine 80%ige Chance gibt ( $4/5$ ), dass du eine Brille trägst. Ist das jetzt wirklich ein absurdes Beispiel? Vielleicht fühlst du dich ja zu bestimmten Menschen hingezogen, weil sie ähnliche Kleidung tragen wie du? Möglich. Vielleicht ist es also doch kein absurdes Beispiel.

Die oben beschriebene Methode wird üblicherweise als *K-Nearest-Neighbours* (*KNN*) bezeichnet, wobei das *K* die Anzahl der nahegelegenen Personen bzw. Datenpunkte bezeichnet, die wir

betrachten: 1, 3, 5, 21, 51 usw. Normalerweise betrachtet man eine ungerade Anzahl an Nachbarn, sodass man nie in die Verlegenheit eines Unentschiedens kommt. Es muss dabei allerdings nicht immer nur zwei Klassen geben (»Brille, keine Brille«), mehr Klassen sind auch möglich. In diesem Fall wird auch die ungerade Anzahl von Nachbarn weniger wichtig.

## 4.1 Wofür K-Nearest-Neighbours eingesetzt werden

- **Kreditrisiko:** *Nächste Nachbarn* können verwendet werden, um Kreditrisiken vorherzusagen, indem man die Kreditgeschichte einer Person mit anderen Kreditgeschichten vergleicht.
- **Medizinische Diagnosen:** Möglich ist auch die Diagnose von Erkrankungen, indem man die Symptome einer Person mit anderen Symptomen aus einer Datenbank abgleicht.
- **Empfehlungen:** Der Algorithmus kann eingesetzt werden, um Empfehlungen für Produkte oder Dienstleistungen auszusprechen, die darauf basieren, wie sehr die Shopping-Gewohnheiten des Kunden oder der Kundin den Gewohnheiten anderer Kunden ähneln.
- **Mustererkennung:** Diese Lernmethode kann auch Bilder, Texte oder Sprache klassifizieren. Dies geschieht basierend auf ihrer

Ähnlichkeit zu verschriftlichten Beispielen.

- **Data Mining:** Basierend auf der Entfernung zu anderen Datenpunkten können über das Nearest-Neighbours-Prinzip auch Ausreißer oder Auffälligkeiten in den eingegebenen Daten gefunden werden.
- **Finanzmärkte:** Basierend auf historischen Trends und den Trends ähnlicher Vermögenswerte kann der zukünftige Preis von Finanzanlagen (z. B. Aktien und Währungen) vorausgesagt werden.
- **Intrusion Detection:** Der Algorithmus kann auch verwendet werden, um böswillige oder verdächtige Aktivitäten in einem Netzwerk oder System zu erkennen – basierend auf Abweichungen von normalen Verhaltensmustern.

## 4.2 Im Detail

*K*-Next-Neighbours ist wahrscheinlich die maschinelle Lernmethode, die man am leichtesten konzeptionell erfassen kann. Dabei findet mit dieser Methode eigentlich gar kein »Lernen« statt!

Stattdessen liegen uns für jede Person Daten in Form von Vektoren vor (beispielsweise für die Position im Raum), und jeder einzelne Datenpunkt wird klassifiziert (zum Beispiel nach »Brille« und »keine Brille«).

Wir widmen uns jetzt mal einem neuen Beispiel – dieses Mal wird es auch tatsächlich ein mathematisches. Handelt es sich bei einer E-Mail um Spam? Der erste Eintrag im Merkmalsvektor für eine E-Mail ist die Anzahl der Wörter in der E-Mail, der zweite Eintrag die Anzahl der Ausrufezeichen, der dritte die Anzahl der Rechtschreibfehler und so weiter. Anhand dieser Ausgangslage gehen wir jetzt mal einen möglichen Spam-Identifikationsalgorithmus durch.

Ich denke mir die Daten dafür einfach aus – und damit es nicht zu kompliziert wird, haben wir nur vier E-Mails, die bereits als *Spam* oder *kein Spam* klassifiziert wurden. Wir haben jetzt also eine Mail, die als Spam klassifiziert ist, und drei normale E-Mails.

A	B	C	D

$\begin{pmatrix} 193 \\ 4 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 229 \\ 3 \\ 16 \end{pmatrix}$	$\begin{pmatrix} 80 \\ 2 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 616 \\ 0 \\ 0 \end{pmatrix}$
	Spam!		

**Tabelle 4.1** Merkmalsvektoren für drei unterschiedliche Mails in unserem Posteingang

In [Tabelle 4.1](#) siehst du die Daten aus den vier E-Mails **A**, **B**, **C** und **D**. E-Mail **A** hat 193 Wörter, vier Ausrufezeichen und keine Rechtschreibfehler. Und so weiter. Wir wissen ja bereits, dass **A**, **C** und **D** kein Spam sind, nur **B** ist eine Spam-E-Mail. (Vielleicht sind die die vielen Rechtschreibfehler schon aufgefallen, die in der untersten Vektor-Zeile angegeben werden. Diese scheinen ein Hinweis darauf zu sein, dass eine E-Mail Spam ist. Aber der Witz bei der Nutzung von KI ist ja, dass sie das Ganze algorithmisch löst. Und um ehrlich zu sein, war meine Datenauswahl auch nicht gerade super subtil.) Dies sind also unsere »Trainingsdaten«.

Jetzt nehmen wir mal an, es kommt eine Mail rein, die durch den nächsten Vektor dargestellt wird. Ist sie Spam oder nicht?

<b>Neue Mail</b>
$\begin{pmatrix} 98 \\ 4 \\ 2 \end{pmatrix}$
?

**Tabelle 4.2** Der Merkmalsvektor für eine neu eingegangene Mail



Na ja, es gibt schon einige Rechtschreibfehler, aber andererseits: Wenn man sich die Länge der Mail so anschaut, sind es auch wieder nicht so wahnsinnig viele. Hmm.

Damit wir die Mails richtig miteinander vergleichen können, müssen wir die Trainingsdaten neu skalieren, sodass die Zahlen in den einzelnen Zeilen der Vektoren alle in der gleichen Größenordnung sind. So haben alle drei Kategorien dasselbe Gewicht bei der Auswertung. Würden wir die Daten nicht neu skalieren, dann wäre die Anzahl der Wörter in einer E-Mail der einzige Bestimmungsfaktor dafür, ob eine E-Mail Spam ist oder nicht.

Ich habe so skaliert, dass die kleinste Zahl in der ersten Zeile  $-1$  ist und die größte  $+1$ . Die Rechnung dahinter ist einfach. Die größte Zahl in der ersten Zeile ist 616 und die kleinste 80. Die Umformung für die erste Zeile sieht dann so aus:

$$A: 193 \rightarrow \frac{2}{616-80} \cdot 193 - \frac{616+80}{616-80} = -0.58$$

$$B: 229 \rightarrow \frac{2}{616-80} \cdot 229 - \frac{616+80}{616-80} = -0.44$$

$$C: 80 \rightarrow \frac{2}{616-80} \cdot 80 - \frac{616+80}{616-80} = -1$$

$$D: 616 \rightarrow \frac{2}{616-80} \cdot 616 - \frac{616+80}{616-80} = +1$$

Das ist natürlich auf den ersten Blick eine kompliziertere Version der Skalierung, die wir weiter oben mit Größe und Gewicht von Personen durchgeführt haben. Das Einzige, was dabei aber wirklich

komplizierter ist, sind die unordentlichen Zahlen, die wir verwendet haben.

Die Umformung der zweiten Zeile, mit Zahlen zwischen 0 und 4, kannst du jetzt selbst durchführen – ebenso wie die Skalierung der dritten Zeile. Natürlich muss dann auch noch der neue, noch nicht klassifizierte Vektor auf dieselbe Weise umgeformt werden.

<b>A'</b>	<b>B'</b>	<b>C'</b>	<b>D'</b>	<b>Neue Mail'</b>
$\begin{pmatrix} -0,58 \\ 1 \\ -1 \end{pmatrix}$	$\begin{pmatrix} -0,44 \\ 0,5 \\ 1 \end{pmatrix}$	$\begin{pmatrix} -1 \\ 0 \\ -0,88 \end{pmatrix}$	$\begin{pmatrix} 1 \\ -1 \\ -1 \end{pmatrix}$	$\begin{pmatrix} -0,93 \\ 1 \\ -0,75 \end{pmatrix}$
	Spam!			?

**Tabelle 4.3** Die neu angepassten Merkmalsvektoren im Vergleich

In [Tabelle 4.3](#) siehst du die vier Vektoren neu skaliert, plus die neue, rätselhafte E-Mail mit ihren Daten.

An dieser Stelle ist ziemlich klar, dass unser neu skaliert Vektor dem ersten skalierten Vektor, der jetzt **A'** heißt, am nächsten liegt. Daraus schließen wir, dass es sich bei der neuen Mail nicht um Spam handelt. Ich habe das noch mal überprüft, indem ich die Abstände zwischen dem neuen Datenpunkt und den Trainingspunkten mithilfe des guten alten Satz des Pythagoras genau berechnet habe:

<b>E-Mail</b>	<b>Abstand zu neuer Mail</b>

A'	0,43
B'	1,88
C'	1,01
D'	3,28

**Tabelle 4.4** Ähnlichkeit der Daten in einer neu eingegangenen Mail im Vergleich zu bereits analysierten und klassifizierten Mails

Ich habe dadurch die Werte in [Tabelle 4.4](#) erhalten. Keine Sorge, ich erwarte nicht, dass du diese Abstände genau so misst – wobei die Rechnung mit den obigen Daten nicht superkompliziert ist.

In der Praxis gäbe es hingegen so einige Unterschiede zu unserem einfachen E-Mail-Beispiel hier:

- Wir hätten viel mehr Trainingsdaten. Wir sprechen von vielen Tausenden von E-Mails oder auch mehr.
- Wir könnten mehr Merkmale haben, sodass unsere Vektoren mehr als drei Einträge hätten.
- Wenn wir einen neuen Datenpunkt klassifizieren würden, würden wir ihn nicht nur mit dem *einen* nächsten Datenpunkt aus unserem Trainingsset abgleichen; wir würden wahrscheinlich die nächsten drei, fünf, 11, 101 usw. Punkte betrachten. Für  $K$  (das diese Menge repräsentiert) müssten wir also erst einmal den idealen Wert bestimmen. Das tun wir, indem wir wie weiter oben beschrieben

einige Daten rein zu Trainingszwecken verwenden und darüber hinaus noch ein Testdaten-Set bereithalten, von dem wir die Klassifizierung schon kennen. Damit überprüfen wir dann, ob unser Algorithmus auch wirklich das tut, was er tun soll. Wir werden unvermeidlich einige Daten falsch klassifizieren, aber dazu bestimmen wir ja das  $K$  – um solche Ungenauigkeiten möglichst gering zu halten.

Mathematisch gesehen, ist der kniffligste Teil der Analyse die Skalierung. Im unten stehenden Projekt stelle ich eine clevere Methode vor, diese Skalierung ganz ohne Berechnungen vorzunehmen.

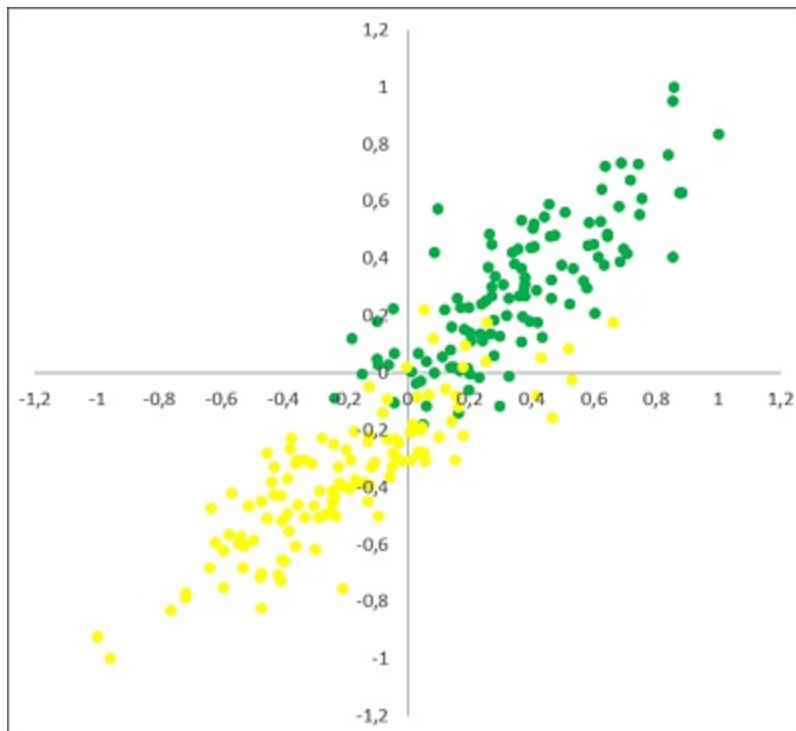
## 4.3 Aus dem Leben gegriffen: Ein Beispiel

Ich habe jede Menge Daten mit dem Gewicht und der Größe von erwachsenen Männern und Frauen zusammengetragen. In [Abbildung 4.1](#) siehst du einen Ausschnitt davon in Tabellenform. (Am Ende des Buches stehen die Daten auch noch einmal in der Originalquelle.) Die Größen sind in Zoll und das Gewicht ist in Pfund angegeben.

	A	B	C	D	E	F	G	H
1		Gender	Height	Weight		Gender	Height	Weight
2		Male	73.84702	241.8936		Female	58.91073	102.0883
3		Male	68.7819	162.3105		Female	65.23001	141.3058
4		Male	74.11011	212.7409		Female	63.369	131.0414
5		Male	71.73090	220.0425		Female	64.48	128.1715
6		Male	69.0810	206.3498		Female	61.7931	129.7814
7		Male	67.25302	152.2122		Female	65.96802	156.8021
8		Male	68.78508	183.9279		Female	62.85038	114.969
9		Male	68.34852	167.9711		Female	65.65216	165.083
10		Male	67.01895	175.9294		Female	61.89023	111.6762
11		Male	63.45649	156.3997		Female	63.67787	104.1516
12		Male	71.19538	186.6049		Female	68.10117	166.5757
13		Male	71.64081	213.7412		Female	61.79888	106.2337
14		Male	64.76633	167.1275		Female	63.37146	128.1182
15		Male	69.28307	189.4462		Female	58.89589	101.6826
16		Male	69.24373	186.4342		Female	58.43825	98.19262
17		Male	67.64562	172.1869		Female	60.8098	126.9155
18		Male	72.41832	196.0285		Female	70.12865	151.2543
19		Male	63.97433	172.8835		Female	62.25743	115.7974
20		Male	69.64006	185.984		Female	61.73509	107.8669
21		Male	67.936	182.4266		Female	63.05956	145.5899
22		Male	67.91505	174.1159		Female	62.28684	139.5227
23		Male	69.43944	197.7314		Female	61.82748	122.7662
24		Male	66.14913	149.1736		Female	66.34754	157.381
25		Male	75.20597	228.7618		Female	65.32063	145.0374
26		Male	67.8932	162.0067		Female	66.10387	148.6452
27		Male	68.14403	192.344		Female	64.52718	132.6809
28		Male	69.08963	184.4352		Female	56.54797	84.87212
29		Male	72.80084	206.8282		Female	62.73928	138.5304
30		Male	67.42124	175.2139		Female	61.5852	137.4253
31		Male	68.49642	154.3426		Female	62.02442	124.6039
32		Male	68.61811	187.5068		Female	65.31133	140.2667
33		Male	74.03381	212.9102		Female	61.20496	108.2901
34		Male	71.52822	195.0322		Female	67.02509	150.2192
35		Male	69.18016	205.1836		Female	60.21117	119.0375

**Abbildung 4.1** Ein Auszug aus einer Tabelle, in der sich Größen- und Gewichtsdaten erwachsener Menschen befinden

In [Abbildung 4.2](#) siehst du die grafische Umsetzung dieser Daten in einem Diagramm. Die Größen und Gewichte habe ich wie beschrieben skaliert und die Männer mit grünen Punkten, die Frauen mit gelben Punkten dargestellt. Auf der horizontalen Achse sind die (skalierten) Größen und auf der vertikalen die (skalierten) Gewichte abgebildet.



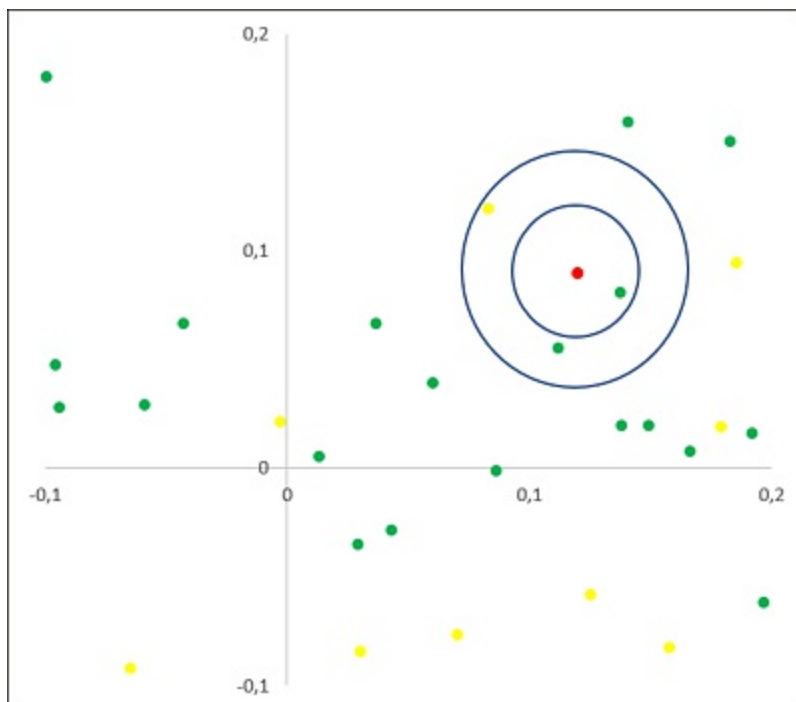
**Abbildung 4.2** Die grafische Umsetzung der vorliegenden Größen- und Gewichtsdaten

Ich habe darauf geachtet, dass es in den Daten genauso viele Männer wie Frauen gibt, um die Ergebnisse für neue, nicht klassifizierte Datensätze nicht zu verzerren.

Der Datensatz oben ist ein schönes Beispiel für KNN: Es gibt nur zwei Merkmale, sodass wir unsere Ergebnisse leicht in einen Graphen

umsetzen können. Sobald wir uns nämlich in drei oder mehr Dimensionen bewegen müssen, wird das Plotten schwierig, und bei vielen Problemen aus der realen Welt können schnell mal Hunderte von Merkmalen in den Vektoren sein.

Die Grafik in [Abbildung 4.2](#) kann nun verwendet werden, um das Geschlecht einer neuen Person zu identifizieren, indem man ihren Datenpunkt einträgt und sich die nächsten Nachbarn ansieht. (Dabei darfst du nicht vergessen, den neuen Datenpunkt genau so zu skalieren, wie die Trainingsdaten skaliert wurden. Fürs Protokoll: Die Daten, die ich verwendet habe, rangierten von 56,2 bis 75,2 Zoll bzw. von 84,9 bis 241,9 Pfund.)



**Abbildung 4.3** Eine Nahaufnahme der Datenpunkte aus [Abbildung 4.2](#)

In [Abbildung 4.3](#) siehst du nun eine Nahaufnahme der vorherigen Darstellung unserer Daten. Der rote Punkt auf dem Bild ist ein neuer, *unklassifizierter* Datenpunkt. Ich habe zwei Kreise um den Punkt gezeichnet. Der innere Kreis umfasst nur den ersten nächsten Nachbarn. Dieser Nachbar ist männlich. Der Schluss liegt also nahe, dass auch der rote Punkt ein Mann ist. Zur Sicherheit habe ich noch einen weiteren, größeren Kreis gezeichnet, der *drei* Nachbarn umfasst. (Du erinnerst dich? Am besten nehmen wir immer eine ungerade Anzahl, um ein Unentschieden zu verhindern.) Von diesen drei Nachbarn sind zwei männlich und einer weiblich. Jetzt wird es statistisch interessant! Mehr Punkte machen unsere Statistiken belastbarer, aber wenn die Punkte dann nicht alle das gleiche Geschlecht anzeigen, senkt das wieder die Wahrscheinlichkeit, dass der neue Datenpunkt männlich ist.

Das Projekt für dieses Kapitel wird dir jetzt zeigen, wie du deine eigene Version dieser Grafik erstellst.



## 4.4 Dein Projekt

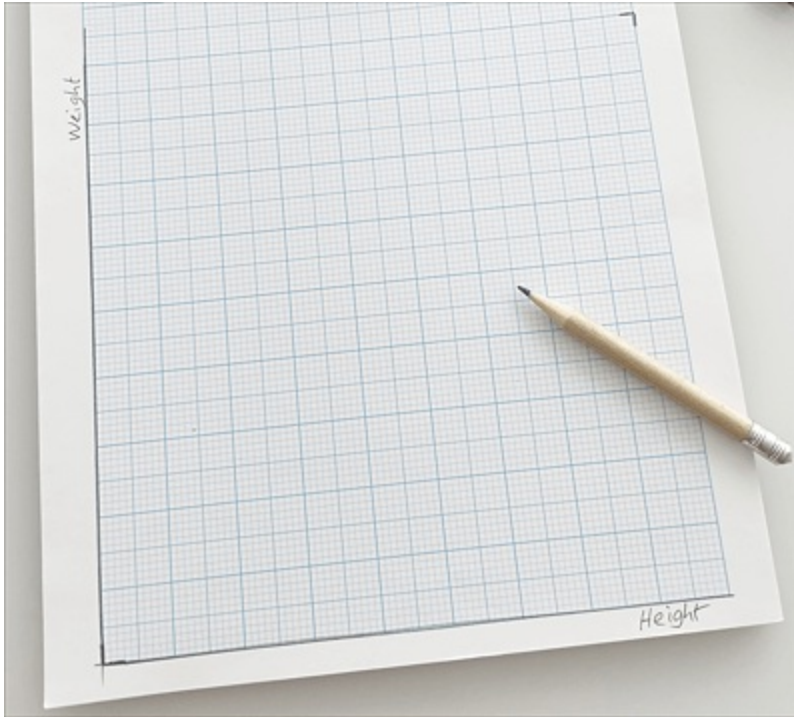
Als eigenes Projekt kannst du jetzt die Größen- und Gewichtsanalyse selbst durchführen. Dafür musst du Größen- und Gewichtsdaten von möglichst vielen erwachsenen Personen sammeln und jeden Datenpunkt als männlich oder weiblich klassifizieren. Auf dieser Basis kannst du dann neue, noch nicht klassifizierte Datenpunkte analysieren lassen und (vielleicht?) herausfinden, ob sie als weiblich oder männlich zu definieren sind. Offensichtlich erfordert dieses Projekt eine Menge kommunikatives Feingefühl:

1. Bitte zunächst deine Familie und Freund\*innen, dir ihre Größe und ihr Gewicht zu nennen. Um Datenpannen (und unangenehme Situationen) zu vermeiden, könntest du ihnen eine zufällig generierte ID-Nummer geben. Sammle so viele Datensätze wie möglich!

Theoretisch müsstest du deine Daten jetzt skalieren, damit sie alle gleich gewichtet werden können. Ich habe hier allerdings noch einen Trick für dich, der dir das ganze Rechnen erspart:

2. Besorge dir etwas Millimeterpapier. (Im Internet findest du dieses Raster auch unter »Funktionspapier« oder »mathematisches Papier«.) Zeichne dir wie in [Abbildung 4.4](#) ein Quadrat ein. Es

sollte das größtmögliche Quadrat sein, das auf das wahrscheinlich rechteckige Papier passt.



**Abbildung 4.4** Millimeterpapier mit einem vorgezeichneten Quadrat darauf. Das ist die Vorbereitung für dein erstes Projekt.

3. Notiere dir das geringste Gewicht, das du in deinen Daten finden kannst. Nicht das niedrigste Männergewicht oder das niedrigste Frauengewicht, sondern das niedrigste Gewicht insgesamt. Jetzt tust du dasselbe für das höchste Gewicht sowie für die kleinste und die größte Angabe zur Körpergröße. Welche Gewichts- und Größeneinheiten du dabei verwendest, spielt keine Rolle.
4. Markiere dir entlang der horizontalen Achse das linke Ende als das mit der kleinsten Körpergröße und das rechte äußere Ende als das mit der größten.

5. Markiere analog dazu auf der vertikalen Achse das untere Ende als das mit dem leichtesten Gewicht und das obere Ende als das mit dem schwersten Gewicht.
6. Jetzt fügst du für jede Person aus deinem Datensatz einen Punkt in das Diagramm ein. Verwende eine Farbe für Männer und eine andere für Frauen. Dadurch, dass du mit einem Quadrat arbeitest und die horizontale und vertikale Linie an die Extreme aus deinen Daten angepasst sind, werden die Daten automatisch skaliert. So ist keine Mathematik erforderlich!
7. Jetzt kannst du schon vorhersagen, ob eine neu hinzugekommene Person männlich oder weiblich ist. Dazu ermittelst du erst einmal deren Größe und Gewicht.



**Abbildung 4.5** *Auf deinen beiden Skalen wurden die ersten Datenpunkte eingetragen. Der Bleistift zeigt auf den ersten neuen Datensatz, der noch zu analysieren ist.*

8. Markiere dir den Datensatz auf deinem Diagramm. Verwende am besten einen Bleistift, mit dem du nur sanft aufdrückst – denn noch hast du das Geschlecht ja nicht vorhergesagt.
9. Jetzt nimmst du dir einen Zirkel. Ziehe einen Kreis um deinen Bleistiftpunkt herum, der die nächsten fünf benachbarten Punkte einschließt.
10. Zähle, wie viele Männer und Frauen sich in diesem Kreis befinden.
11. Die Ergebnisse kannst du jetzt für deine Vorhersage verwenden. Fünf Männer und keine Frauen? Deine neue Stichprobe ist ziemlich sicher männlich. Drei zu zwei? Vermutlich männlich, aber eher unsicher.
12. Beobachte, wie sich die Ergebnisse ändern, wenn du größere Kreise ziehst, die mehr als fünf Nachbarn einschließen.

Wenn du dich irgendwie unwohl dabei fühlst, Menschen nach ihrer Größe, ihrem Gewicht und ihrem Geschlecht zu fragen, dann kannst du auch andere Klassen zur Analyse wählen. Ich schlage nur deswegen zwei Merkmale vor, weil es dann einfacher ist, die Diagramme dazu zu zeichnen. Es müssen dabei aber nicht unbedingt diese beiden Klassifikationen sein.

## Kapitel 5

# Mehr zu: Regression

Machine Learning, das mittels Regressionsmethoden erfolgt, gehört zum überwachten Lernen. Mithilfe einer *Regression* wird versucht, eine abhängige Variable zu einer unabhängigen in Bezug zu setzen. Die unabhängigen Variablen sind dabei numerisch und es werden Geraden, Polynome oder andere Funktionen verwendet, um die dazugehörigen abhängigen Variablen vorherzusagen. Diese Methode kann auch zur Klassifizierung verwendet werden, denn die abhängigen Variablen treten normalerweise in Form von Nullen und Einsen in Erscheinung.

Vielleicht kennst du die Regression schon vom Zeichnen gerader Linien durch eine Gruppe von Datenpunkten. Ein Beispiel: Dir liegen der Verkaufswert und die Quadratmeterzahlen für eine Menge Häuser vor. Gibt es eine einfache lineare Beziehung zwischen den beiden Größen *Preis* und *Fläche*? Die allerwenigsten Beziehungen dieser Art würden ganz *exakt* entlang einer geraden Linie verlaufen. Wie würde also eine möglichst gut an die Daten angepasste Gerade durch die Datenpunkte verlaufen? Wenn du die Linie für diese Daten gezogen hast, kannst du zu höheren Dimensionen übergehen. Gibt es zum

Beispiel eine nahezu lineare Beziehung zwischen dem Verkaufswert und sowohl der Quadratmeterzahl eines Hauses als auch dessen Stellplatzanzahl?

Hinsichtlich ihrer mathematischen Komplexität ist die Regression etwas anspruchsvoller als die Methoden, die du bisher kennengelernt hast. Wenn man das erste Mal mit einer Regression arbeitet, ist man schnell verleitet, einfach ein Diagramm zu zeichnen – abhängige Variable auf die vertikale Achse, unabhängige Variable auf die horizontale Achse – und eine gerade Linie durch die Datenpunkte darauf zu ziehen. Einfach nur auf der Basis gesunden Menschenverstands und Pi mal Daumen. Wenn du eine Regression aber richtig durchführen möchtest, musst du vor dem Zeichnen eine Kostenfunktion durchführen – das gilt auch und vor allem dann, wenn es mehr als eine unabhängige Variable gibt und eine Darstellung der Linie unmöglich wird.

## 5.1 Wofür Regression eingesetzt wird

- **Werbung:** Regressionsmodelle können verwendet werden, um den Einfluss verschiedener Variablen auf das Ergebnis einer Kampagne vorherzusagen, wie zum Beispiel bei Umsätzen im Einzelhandel oder Marketingkampagnen, und die finanziellen und personellen Ressourcen entsprechend zu optimieren.

- **Kund\*innentrends:** Mithilfe einer Regression kann das Verhalten oder können die Vorlieben von potenziellen Kund\*innen auf der Grundlage ihrer Interaktionen mit Buttons oder Webseiten gemappt werden. Das geschieht zum Beispiel bei Streaming-Diensten oder auf E-Commerce-Websites.
- **Data Mining:** Regressionsmodelle können Muster in und Beziehungen zwischen Variablen in einem Datensatz aufdecken, zum Beispiel in Form von Korrelationen, Ausreißern oder Auffälligkeiten.
- **Mustererkennung:** Bilder, Texte oder Sprache können auf der Grundlage ihrer Merkmale und Labels klassifiziert werden. Regressionsmodelle werden zum Beispiel bei der Gesichtserkennung, bei Sentimentanalysen oder zur Spracherkennung eingesetzt.
- **Finanzindustrie:** Regressionsmodelle können Trends bei Aktienkursen erkennen, Preissteigungen oder -verfall vorhersagen und Risiken bewerten.
- **Umweltwissenschaft:** Hier können die Auswirkungen des Klimawandels auf Basis einer Vielzahl unterschiedlicher Faktoren vorhergesagt werden.
- **Psychologie:** Die Regressionsanalyse kann verwendet werden, um beispielsweise die Wahrscheinlichkeit einer Depression vorherzusagen. Diese Analyse basiert dann auf Faktoren wie Alter, Geschlecht und sozioökonomischer Status. Auf ähnliche Art und

Weise kann auch ein Substanzmissbrauch anhand verschiedener Faktoren wie Familienhintergrund und Gruppenzwang vorhergesagt werden



## 5.2 Im Detail

Wenn wir die Regression in ihrer einfachsten Form betrachten, stellt sie einfach nur eine Beziehung zwischen abhängigen und unabhängigen Variablen her und bildet auf dieser Grundlage eine gerade Linie aus den unterschiedlichen Datenpunkten.

In anderen Worten bedeutet das für uns, dass auf der horizontalen Achse, der x-Koordinate, das *Merkmal* liegt, und auf der vertikalen Achse, der y-Koordinate, die *Menge* bzw. Anzahl, die wir vorhersagen möchten. Die Datenpunkte in diesem System könnten zum Beispiel die Anzahl der Schüler\*innen in einem Klassenzimmer (auf der vertikalen Achse) und ihre Note oder Klassenstufe (auf der horizontalen Achse) sein.

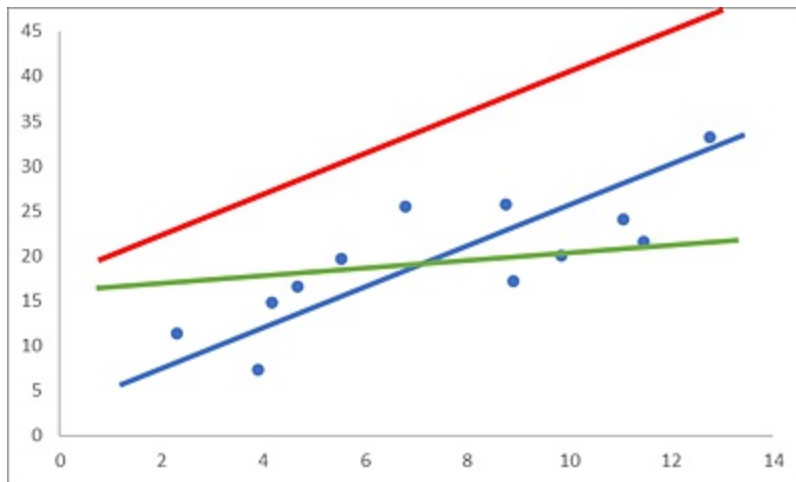
Die zentrale Frage bei diesen Beispielen ist: Was ist die *exakte* gerade Linie durch die Punkte?

Es geht aber auch noch weitaus komplizierter! Zum Beispiel könnte der Zusammenhang zwischen den Punkten nicht entlang einer geraden Linie verlaufen – denk zum Beispiel an das Wachstum von Bakterien in einer Petrischale. Immerhin ist es noch denkbar, die Beziehung von *Bakterienmenge* zu *Zeit* in einem zweidimensionalen Diagramm darzustellen.

Aber manchmal gibt es auch mehr als nur eine unabhängige Variable. Lass uns zu dem Beispiel mit den Verkaufspreisen von Häusern und dem Zusammenhang mit ihrer Quadratmeterzahl, Stellplatzanzahl und, sagen wir mal, der Entfernung zum nächsten Bahnhof zurückkommen. Der ganze Komplex müsste schon in einem vierdimensionalen Graphen abgebildet werden. Auf der vertikalen Achse mit der abhängigen Variable lägen die Hauspreise – und dann bräuchten wir noch drei horizontale Achsen: eine für die Quadratmeterzahl, eine für die Stellplatzanzahl und eine für die Entfernung zum Bahnhof. Aber wie sollen wir denn vier Dimensionen darstellen?

Im Rahmen dieses Buches werden wir uns nicht mit Problemen befassen, für die wir die Graphen nicht zeichnen können – wir bleiben bei zwei Dimensionen. Wenn wir also gleich zu deinem eigenen Projekt kommen, musst du nur eine Gerade anpassen.

Aber nun zurück zu unserer Frage: Wie findet man die beste gerade Linie durch die Datenpunkte eines Modells und wie funktioniert die Mathematik dahinter?

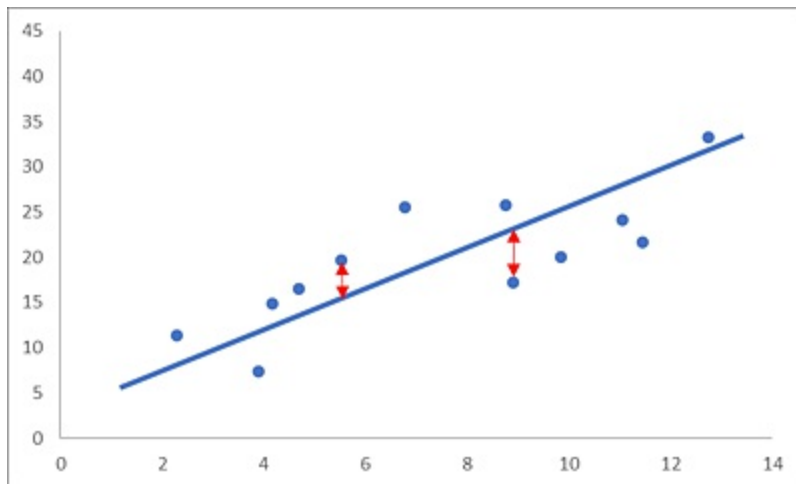


**Abbildung 5.1** Drei mögliche Graphen für eine Gruppe an Daten

Schau dir [Abbildung 5.1](#) einmal genauer an. Welche der drei geraden Linien sieht so aus, als würde sie am besten zu den Datenpunkten passen?

Es ist vermutlich offensichtlich, dass die blaue Linie die beste ist. Die rote Linie ist zu hoch: Alle Datenpunkte liegen unter ihr. Die grüne Linie hat zumindest einige Punkte über und unter sich, aber sie hat die falsche Steigung.

Dieses Experiment macht deutlich, dass wir offenbar zwei Faktoren unserer Geraden bestimmen müssen: die Höhe der Linie und ihre Steigung. Diese beiden Faktoren werden so gewählt, dass sie sich möglichst wenig von den eingetragenen Datenpunkten entfernen. Wie ich oben schon gesagt habe, wird dieser Unterschied zwischen Linie und Daten normalerweise durch einen mathematischen Ausdruck beschrieben – die *Kosten-* oder *Verlustfunktion*.



**Abbildung 5.2** *Abstände einzelner Datenpunkte zum ausgewählten Graphen*

In [Abbildung 5.2](#) habe ich einige Abstände eingezeichnet: Einer der Punkte liegt über der blauen Linie und einer darunter. Normalerweise werden bei der Berechnung einer Kostenfunktion die Abstände jedes einzelnen Punktes (12 in diesem Beispiel) zur Geraden bestimmt, sie werden quadriert und dann zusammenaddiert. Dadurch bekommen wir einen Wert, der die Gesamtabweichung unserer Geraden von den Punkten dokumentiert. Jetzt kann die Höhe und/oder die Steigung der Linie verändert werden und die Berechnung wird erneut durchgeführt. Das Ziel ist es, den Abweichungswert so gering wie möglich zu halten.

Ich zeige die Rechnung im Folgenden mal mit drei der Datenpunkte aus den oben abgebildeten Diagrammen. Ich nutze die Punkte (2,3; 11,3), (3,9; 7,4) und (4,7; 16,6). (Ich hätte auch alle Punkte aus meinen Daten verwenden können, aber das hätte die Rechnung einfach länger gemacht. Nicht anders, nur länger.)

Wir beschreiben den Verlauf der Linie mit  $y = ax + b$ .

Jetzt setzen wir für  $x$  und  $y$  die drei Datenpunkte ein. Die Abweichung, die ich eben beschrieben habe, ist dann die Kostenfunktion:

$$(2,3a + b - 11,3)^2 + (3,9a + b - 7,4)^2 + (4,7a + b - 16,6)^2$$

Wenn wir  $a$  und  $b$  ändern, ändert sich der Wert der Kostenfunktion. Das wäre der perfekte Zeitpunkt, um mehr über Tabellenkalkulationen zu lernen. Dazu kannst du dir den Hinweiskasten ganz am Ende dieses Kapitels anschauen.

Wenn wir jetzt also wie oben mit unseren Daten herumprobieren, stellen wir bald fest, dass die geringste Abweichung unserer Geraden von den Datenpunkten auftritt, wenn wir folgende Parameter einsetzen:  $y = 1,54x + 6,15$ .

Wenn man diesen Vorgang mit mehr Datenpunkten wiederholen würde, käme natürlich ein etwas anderes Ergebnis dabei heraus, das Prinzip wäre aber dasselbe.

## 5.3 Aus dem Leben gegriffen: Ein Beispiel

In diesem Beispiel werden wir eine einfache Regression anwenden, um die Anziehungskraft der Schwerkraft zu beschreiben. Dabei mache ich mir selbst das Leben ein bisschen schwerer als zuvor, denn wir werden mit einer quadratischen Funktion arbeiten, also mit einer Parabel.

Wirft man einen Ball ganz gerade nach oben in die Luft, steigt er mit *abnehmender* Geschwindigkeit, bis er nicht länger an Höhe gewinnt und mit *zunehmender* Geschwindigkeit wieder nach unten fällt. Dieses Verhalten verdanken wir der Schwerkraft. Wir werden ein einfaches Experiment durchführen, indem wir die Höhe des Balls an verschiedenen Stellen messen und daraus seine Beschleunigung durch die Anziehung der Schwerkraft berechnen. Als Hommage an Sir Isaac Newton hätte ich statt eines Balls auch einen Apfel verwenden können.

Folgende Schritte müssen wir für das Experiment durchführen:

1. Erst einmal brauchen wir eine Möglichkeit, die Höhe des Balls zu messen. Unser Wurf darf nicht zu niedrig sein, sonst wären die Daten nicht genau genug. Also habe ich eine Schnur mit Markierungen mit je einem Meter Abstand versehen und sie aus

einem Fenster im zweiten Stock gehängt. Sie sollte bis zum Boden reichen.

2. Als Nächstes brauchen wir etwas, mit dem wir Bildaufnahmen vom Wurf machen können – ich habe dafür ein Handy mit Slow-Motion-Funktion verwendet und es so positioniert, dass im Bildausschnitt sowohl das obere als auch das untere Ende der Schnur zu sehen waren.
3. Jetzt schalten wir die Slow-Motion-Funktion ein, drücken auf Aufnahme und jemand im Bild wirft den Ball so, dass er dem höchsten Punkt der Schnur möglichst nah kommt. Zur Sicherheit sollte das ein paarmal wiederholt werden.
4. Eine der Aufnahmen wählen wir für unser Experiment aus.
5. Zeit, unsere Daten aufzuschreiben! Beim Durchsehen der Aufnahmen können wir jetzt eine Liste schreiben, die Zeit und Höhe miteinander in Beziehung setzt. Ich habe die Höhen ab dem Punkt gemessen, an dem mein Assistent den Ball losgelassen hat. Als Zeitmarker hatte ich allerdings nur die Zeitanzeige des Videos – ich musste also ein bisschen googeln, wie man diese Zeitlupenaufnahme in Echtzeit umrechnet.
6. Dann ging es ans Millimeterpapier – ich habe zwei Achsen eingezeichnet. Die horizontale Achse ist für die Zeit (oder auch Video-Frames) und die vertikale für die Höhe des Balls, gemessen an der Schnur. Na gut, das ist nicht ganz richtig – ich

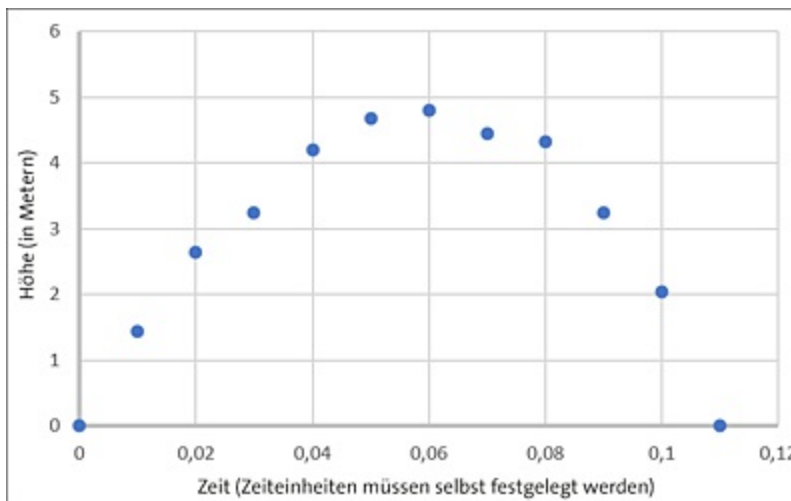
habe die Dokumentation in einer Tabelle gemacht, nicht auf dem Millimeterpapier:

<b>Zeit</b>	<b>Höhe</b>
0	0
0,01	1,4
0,02	2,6
0,03	3,2
0,04	4,2
0,05	4,7
0,06	4,8
0,07	4,4
0,08	4,3
0,09	3,2
0,10	2,0
0,11	0

**Tabelle 5.1**    *Zeit- und Höhenmessung des geworfenen Balls*



7. Wir übertragen die Punkte auf das Millimeterpapier oder in die Tabelle. Das Ergebnis sollte ungefähr so aussehen wie der Graph in [Abbildung 5.3](#). Vielleicht ist es etwas knifflig, die richtigen Zeitpunkte erfasst zu bekommen, aber es ist wirklich nicht schlimm, wenn die Ergebnisse nicht super genau sind. In diesem Buch geht es um die Mathematik hinter der KI, nicht um Videobearbeitung.



**Abbildung 5.3** Mit den Zeit- und Höhendaten, die wir aus unserem Video ermittelt haben, können wir einen solchen Graphen zeichnen.

8. Jetzt geht es an den technischen Teil. Wir müssen zu den eingetragenen Punkten die bestmöglich passende Parabel bestimmen.

Wenn es um die naheliegendste Gerade gehen würde, wäre der Fall denkbar einfach: Du würdest ein Lineal zur Hand nehmen und einfach die Linie an der Stelle einzeichnen, wo sie am besten aussieht. Vielleicht mit etwa der Hälfte der Punkte über und unter der Linie.

Aber die Kurve, die hier gesucht ist, ist die einer quadratischen Funktion – also eine, die mit der folgenden Gleichung beschrieben werden kann:

$$\text{Höhe} = at^2 + bt + c$$

$t$  steht dabei für die Zeit. Unser Ziel ist es, die drei Parameter  $a$ ,  $b$  und  $c$  zu bestimmen, die für die ideale (also niedrigstmögliche) Abweichung von den Daten sorgen. Aber bevor wir damit beginnen, kommt hier noch eine kurze Erläuterung.

Unser Beispiel hier ist eher ungewöhnlich in der Welt des Machine Learnings. Es gibt da nämlich eine Besonderheit, mit der man nicht oft in Berührung kommt: Wir haben jetzt also ein paar Punkte auf Millimeterpapier, die unsere Daten darstellen. Wir wollen eine Kurve an diese Punkte anpassen, das ist unser geplantes mathematisches Modell.

Welche Form haben diese Punkte? Na ja, ich würde sagen, sie sehen ziemlich parabolisch aus. Also verläuft unsere anzupassende Kurve in der Form einer Parabel. Bei anderen mathematischen Modellen können wir an dieser Stelle maximal eine recht brauchbar angepasste Linie zeichnen. Wenn die Gerade nicht zum Modell passt, müssten wir sogar noch eine andere Form ausprobieren: kubisch, quartisch, vielleicht sogar noch komplexer. Das ist nicht mal unwahrscheinlich!

Aber unsere Daten in diesem Fall stammen aus einem sehr spezifischen Versuch: einem Ball, der unter Einfluss der Schwerkraft aufsteigt und fällt. Und laut Sir Isaac Newton sollte die Kurve, die der

Ball dadurch beschreibt, *genau* eine Parabel beschreiben! Mit anderen Worten: Wir haben hier eine Funktion für die Regression vorliegen, von der uns die Wissenschaft ganz genau sagt, wie sie auszusehen hat. Glaub mir, so ein glücklicher Zufall wird dir sonst nur in Lehrbüchern und konstruierten Prüfungsaufgaben begegnen!

Laut Newton kann die Beziehung zwischen Höhe und Zeit in dieser Parabel mit  $a = -\frac{1}{2} \cdot g$  beschrieben werden.

$g$  ist dabei die Beschleunigung durch die Schwerkraft. Es hat den Wert  $9,81 \text{ m/s}^2$ .

Nun aber zurück zu unseren Daten aus dem Experiment und der Suche nach der besten Parabelform. Da wir Zeit und Höhe ab 0 messen, können wir schon einmal den Parameter  $c$  auf 0 setzen, sodass nur noch  $a$  und  $b$  bestimmt werden müssen.

Dafür benötigen wir eine *Kostenfunktion*, die uns angibt, wie weit das Modell (in diesem Fall eine Parabel) von den Daten (den Punkten) entfernt liegt. Wie erwähnt, wird die Kostenfunktion, die oft für die Erstellung einer Regression verwendet wird, berechnet, indem alle Abweichungswerte der Datenpunkte von der Kurve quadriert und summiert werden. Man kann sie für dieses Beispiel auch wie folgt schreiben:

$$\text{Kostenfunktion} = \sum_{i=1}^N (\text{gemesseneHöhe}_i - \text{Kurvenhöhe}_i)^2.$$

$N$  ist dabei die Anzahl der Datenpunkte – in diesem Beispiel gilt also  $N = 11$ , den Punkt  $(0,0)$  schließe ich dabei aus. Die *gemesseneHöhe* stammt aus den Daten, und die *Kurvenhöhe* berechnen wir aus der

Formel für die Parabel, die uns Newton freundlicherweise zur Verfügung gestellt hat. Für den ersten Punkt rechnen wir dann wie folgt:

$$\text{gemesseneHöhe}_j = 1,4$$

$$\text{Kurvenhöhe}_j = a \cdot 0,012 + b \cdot 0,01$$

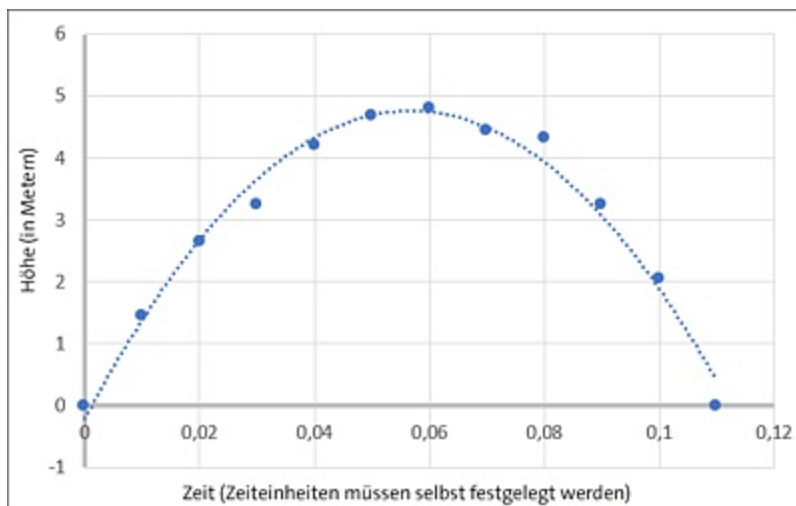
Wenn du alle Terme in der Kostenfunktion addierst, erhältst du einen Ausdruck mit unbekanntem  $a$  und  $b$ . Dann musst du für diese Parameter nur noch die Werte finden, die die Kostenfunktion kleinstmöglich halten. Dieses Vorgehen wird als *Methode der kleinsten Quadrate* bezeichnet (kurz *MKQ*).

Du brauchst dir keine Gedanken darüber zu machen, wie dieser kleinstmögliche Wert gefunden wird. Du kannst ihn einfach mithilfe der Tabelle in [Abbildung 5.6](#) am Ende des Kapitels ablesen. Alternativ gibt es auch noch eine Formel, um ihn zu berechnen – oder du probierst einfach mit den Werten herum, bis du die kleinste Abweichungssumme gefunden hast. Für das folgende Projekt ist es aber nicht wichtig, dass du genau weißt, wie man diese Werte bestimmt – es reicht völlig, wenn du weißt, dass es sich dabei um ein ziemlich unkompliziertes mathematisches Problem handelt, das schnell gelöst ist, wenn du dich ein bisschen mit ihm beschäftigst.

Bestimmt fragst du dich jetzt, wie nah mein Ergebnis am richtigen Wert war. Nein? Ich sag's dir trotzdem: Abgeleitet von meinen Daten und der bestmöglichen Parabelanpassung daran habe ich eine Beschleunigung durch die Schwerkraft von  $9,6 \text{ m/s}^2$  errechnet. Ganz

schön gut, finde ich. (Die »Zeit« in meinen Daten musste ich ja wegen der Zeitlupenfunktion im Video erst einmal umrechnen. Wenn du das Experiment selbst durchführen möchtest, musst du das ebenfalls tun. Dafür musst du allerdings erst einmal feststellen, wie viele Bilder pro Sekunde deine Foto-App aufnimmt.)

In [Abbildung 5.4](#) siehst du meine ursprünglichen Daten noch einmal. Dieses Mal habe ich die bestmöglich angepasste Parabel mit eingefügt.



**Abbildung 5.4** Im Diagramm ist neben den erhobenen Daten auch der Graph eingezeichnet, der den korrekten Kurvenverlauf der Beschleunigung des Balls anzeigt.

## 5.4 Dein Projekt

Hier nehmen wir uns mal was ganz Einfaches vor: Gibt es einen Zusammenhang zwischen dem Gewicht eines Hundes und der Menge an Futter, die er frisst? (Und nein, es geht natürlich nicht darum, im Anschluss die Futtermenge zu variieren, um den Hund auf Diät zu setzen oder zu mästen. Es geht hier nur um das Verhältnis von »Tier« zu »üblicher Futtermenge«.)

Wobei, warum sollten wir dieses Projekt auf Hunde beschränken? Wir können einfach alle Haustiere mit einbeziehen. Los geht's!

1. Als Erstes wiegst du all deine Tiere. Das ist jetzt natürlich einfacher gesagt als getan. Wenn du eine Katze oder einen kleinen Hund hast, sollte das ziemlich einfach sein: Erst wiegst du nur dich selbst, und dann wiegst du dich, während du dein Haustier auf dem Arm hast. Die Differenz zwischen den Messungen ist das Gewicht deines Haustiers. Dein Hund ist zu schwer? Dein Tierarzt könnte den Wert in seiner Akte stehen haben – oder er steht in deinem Haustierpass. Wenn du einen reinrassigen Hund hast, weiß Google vielleicht sogar die Antwort. Du hast ein Kaninchen oder ein Meerschweinchen? Versuch's mal mit der Küchenwaage bzw. mit einer Schüssel auf der Küchenwaage. Du hast eine Maus oder eine Ratte, die auch für eine Küchenwaage ein bisschen zu klein ist, und dazu noch etwas

zappelig? Hier kannst du es wieder mit Googeln versuchen. Okay, aber wie sieht es mit einem Wellensittich aus? Ehrlich, google das bitte direkt – einem Wellensittich brauchst du mit der Küchenwaage gar nicht erst zu kommen.

2. Jetzt wiegst du das Futter, das du deinem Haustier an einem normalen Tag gibst. Hier solltest du eigentlich keine Probleme bekommen, wenn du ein unkompliziertes Haustier hast. Na gut, es gibt Ausnahmen. Mehrere Hunde, die aus einer Schüssel fressen zum Beispiel. Oder sehr große Tiere. Hast du eine Schlange? Ich denke, die können wir hier vernachlässigen, weil sie nicht besonders oft fressen. Spinnen, Echsen oder Stabschrecken? Überspringen wir auch. Tropenfische? Das ist definitiv ein Kandidat für ein anderes Mal.
3. Nun stellst du den Futterverbrauch dem Gewicht des Tieres gegenüber, ein Punkt pro Haustier. Ich würde vorschlagen, für jede Tierart ein anderes Diagramm zu erstellen.
4. Danach zeichnest du eine Gerade durch diese Punkte – eine Gerade für jedes Tier. Das ist deine Regression für das Beispiel.

### **Einige Hinweise**

- Du brauchst nicht viele Tiere pro Diagramm. Es sind weder Hunderte noch Dutzende nötig. Aber du solltest auf jeden Fall mehr als zwei oder drei haben. Wenn du nämlich für deine lineare Regression nur zwei Punkte

verwendest, wird die Anpassung der Geraden absolut perfekt sein. Dein Abweichungswert wird also gleich 0 sein – dabei könntest du einen massiven Denkfehler in deinem Modell haben. Je mehr Punkte du im Diagramm verwendest, desto besser bekommst du ein Gefühl dafür, ob die Gerade eine gute Umsetzungsform für dein mathematisches Modell ist oder nicht.

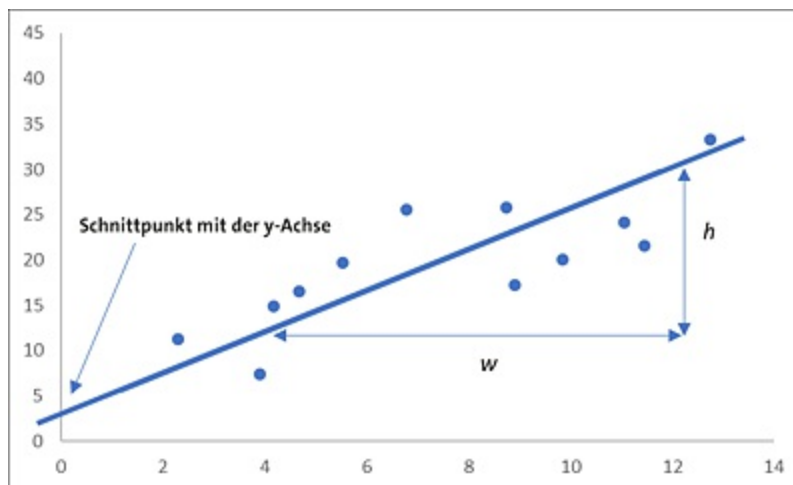
- Da du für dieses Beispiel keine Abstände zwischen den Merkmalsvektoren misst, musst du hier auch keine Neuskalierung durchführen.
- Du musst die Abweichung nicht mathematisch berechnen. Das Prinzip einer Kostenfunktion und deren Minimierung habe ich ja erklärt – ein geschätzter Wert ist für dieses Projekt absolut ausreichend.

5. Wenn du für jedes Haustier ein Diagramm erstellt hast, kannst du jetzt zwei Werte auf der eingezeichneten Linie abmessen: einmal ihren Schnittpunkt mit der  $y$ -Achse und einmal die Steigung der Geraden.
6. Der Schnittpunkt ist der Punkt, an dem deine Gerade die  $y$ -Achse kreuzt. Der Punkt gibt an, wie viel Futter du deinem Haustier geben würdest, wenn es nichts wiegen würde. Offensichtlich ist



das ein etwas fragwürdiges Konzept, aber so kannst du sehen, ob deine Gerade auch für niedrige Gewichte die richtige Form hat.

7. Die Steigung berechnest du, indem du, wie in [Abbildung 5.5](#) dargestellt, ein Dreieck einzeichnest. Dann berechnest du das Verhältnis der Höhe des Dreiecks ( $h$ ) zu seiner Breite ( $w$ ). Damit das Ergebnis möglichst genau ist, solltest du das Dreieck so groß wie möglich einzeichnen. Schau dir vor allem an, wie die Steigung von Tier zu Tier variiert.



**Abbildung 5.5** Die Steigung des ausgewählten Graphen kann mithilfe von  $h$  und  $w$  berechnet werden.

## Übrigens: Verwendung einer Tabelle für deine Regression

Wenn du mal versuchen möchtest, eine Kostenfunktion mit Excel zu minimieren, kannst du die folgenden Schritte ausprobieren:

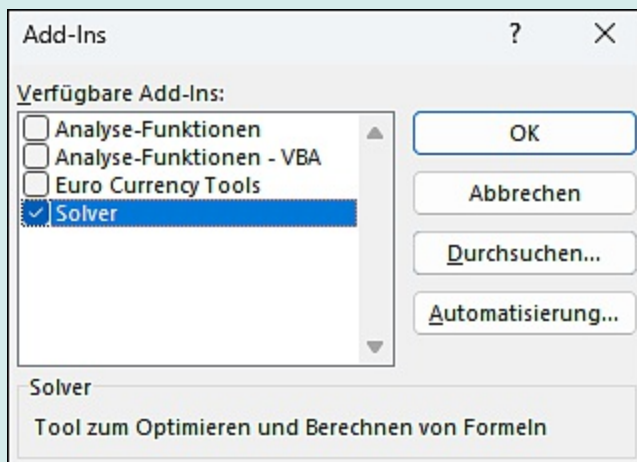
1. Gib deine Daten wie in eine Tabelle unterhalb von x und y ein (siehe [Abbildung 5.6](#)).
2. Gib ungefähre Werte, die dir passend vorkommen, für die Parameter a und b in die gelben Zellen ein.
3. Berechne mithilfe der abgebildeten Formel die Datenabweichung zu deiner Linie.
4. Nimm die Ergebnisse zum Quadrat und addiere sie dann. (Das Ergebnis dieser Rechnung siehst du im blauen Kasten.)
5. Jetzt kannst du diesen Wert mithilfe des integrierten Solvers von Excel optimieren.

	A	B	C	D	E	F	G
1			a	1,54465			
2			b	6,15445			
3							
4		x	y	Abweichung			
5		2,3	11,3	-1,59286	=B5*\$D\$1+\$D\$2-C5		
6		3,9	7,4	4,77857			
7		4,7	16,6	-3,18571			
8				35,5207	=SUMSQ(D5:D7)		
9							
10							

**Abbildung 5.6** In Excel lässt sich die Kostenfunktion auch schnell und simpel durch Ausprobieren ermitteln.

Die Formeln, die du brauchst, siehst du in der Abbildung. Um sie nutzen zu können, musst du den Solver von Excel aktivieren.

Dazu klickst du in Excel auf **Start** → **Optionen** → **Add-Ins** → **Verwalten: Excel-Add-Ins: Los ...**. Danach öffnet sich das Fenster, das du in [Abbildung 5.7](#) siehst, und du kannst den Solver einfach aktivieren.



**Abbildung 5.7** Nach einem Klick auf »Add-Ins« in den Optionen von Excel kannst du den Solver mit einem Mausklick aktivieren.

Aber denk daran: Es lohnt sich an dieser Stelle absolut nicht, an dieser Berechnung zu verzweifeln – setz dich wirklich nur damit auseinander, wenn du auf der Suche nach dem Tüpfelchen auf dem *i* für dein Projekt bist.

## Kapitel 6

# Mehr zu: Clustering

Kürzlich habe ich ein wunderschönes Herrenhaus besucht, ein National-Trust-Anwesen an der Südküste Englands. An den Wänden hingen die üblichen Familienporträts der ehemaligen Bewohner\*innen, unter der Decke gab es die klassischen alten Eichenbalken, wie man das eben so kennt. Es gab Räume, die ausschließlich der Käse- und Bierherstellung gewidmet waren – und dann war da natürlich die Bibliothek. Ich habe schon einige ganz ähnliche Räume in ganz ähnlichen Anwesen gesehen, aber bei diesem Anblick musste ich wirklich den Reiseleiter kurz fragen, nach welchem System die früheren Bewohner\*innen des Anwesens wohl ihre Bücherregale organisiert haben. Die Antwort, die ich erwartet habe, wäre zum Beispiel »Fiktion steht hier drüben, Sachbücher an der anderen Wand« oder »Da vorne stehen die Klassiker, moderne Literatur findet sich weiter hinten« gewesen. Stattdessen bekam ich folgende Antwort: »Vor einigen Jahrzehnten wurden alle Bücher der Bibliothek bei einem Feuer zerstört – also ging der Besitzer in einen Antiquitätenladen in London und kaufte einige Meter alter Bücher. Sie

stehen noch so in diesen Regalen, wie sie damals aus den Transportkisten geholt wurden.«

Das ist doch mal eine ungewöhnliche Art und Weise, seine Regale zu sortieren! (Ich habe mich allerdings gefragt, wie die Familienporträts den Brand überstanden haben ...)

Genau um dieses Prinzip geht es beim Clustering. Stell dir die ganzen Kategorien vor, nach denen du deine Bücher einteilen könntest: fiktionale oder Sachliteratur, Alter, Farbe, Größe, persönliche Bewertung, Wert und so weiter. Wenn du diese Einteilung selber vornehmen würdest, müsstest du wahrscheinlich jedes Buch einzeln beschriften – das wäre dann eine Klassifikation mit *überwachtem* Lernen.

Aber beim Clustering übernimmt ja der Algorithmus diese Klassifizierung für dich. Die Daten wären dabei die Gruppierungen, die ich grade aufgezählt habe: fiktionale Titel oder Sachliteratur, das Alter usw. Nach dieser Vorgabe der Klassen wird es dann aber dem Algorithmus überlassen, welche Elemente er in welche Gruppen einteilt. Das macht Clustering zu einer *unüberwachten* Lernmethode. Und das macht Clustering für uns ganz besonders spannend.

## **6.1 Wofür Clustering eingesetzt wird**

- **Empfehlungssysteme:** Clustering kann verwendet werden, um Benutzer\*innen einer App oder auch Artikel basierend auf ihrer Ähnlichkeit oder hinsichtlich ihrer Vorlieben zu gruppieren und dann Produkte oder Dienstleistungen zu empfehlen, die sie wahrscheinlich interessieren.
- **Suchmaschinen:** Mithilfe von Clustering erhältst du Ergebnisse, die deiner ursprünglichen Anfrage entsprechen oder ähnlich zu ihr sind.
- **Klassifizierung von Dokumenten:** Diese Art von Algorithmen können verwendet werden, um Dokumente auf der Grundlage der von dir vorgegebenen Themen, Schlüsselwörter oder Stimmungen in verschiedene Kategorien einzuteilen.
- **Medizinische Diagnosen:** In medizinischen Daten können durch Clustering sowohl Muster als auch Anomalien gefunden werden. So können Blutdruck, Herzfrequenz oder Symptome darüber Aufschluss geben, ob eine medizinische Diagnose notwendig sein könnte, oder die Clustering-Algorithmen helfen dabei, bei vorliegenden Erkrankungen den Verlauf zu beobachten.
- **Marktsegmentierung:** Clustering-Algorithmen können Kund\*innen auf der Grundlage ihrer demografischen Zugehörigkeit, ihres Verhaltens oder ihrer Bedürfnisse in verschiedene Segmente unterteilen, damit dann Marketingstrategien für jedes Segment entworfen werden können.

- **Sportwissenschaft:** Hier können mittels Clustering Spieler oder Spielerinnen identifiziert werden, die sich hinsichtlich ihrer körperlichen Merkmale oder ihres Spielstils ähneln.
- **Bildsegmentierung:** Clustering-Algorithmen können verwendet werden, um ein Bild in verschiedene Regionen zu unterteilen. Das geschieht auf der Grundlage ihrer Farbe, Textur oder Intensität. Im Anschluss können sie dann Aufgaben wie Objekterkennung, Gesichtserkennung oder Kompression durchführen.

## 6.2 Im Detail

Genau wie der Nearest-Neighbours-Algorithmus in [Kapitel 4](#) verwenden Clustering-Methoden Abstände zwischen Merkmalsvektoren, um ihre Ähnlichkeit zueinander zu messen. Je kleiner die Distanz zwischen den Vektoren ist, desto ähnlicher sind sich die Daten.

Aber Clustering-Methoden (es gibt davon nämlich mehrere) unterscheiden sich von den K-Nearest-Neighbours dadurch, dass sie nach unbeaufsichtigten Methoden lernen, sodass die einzelnen Datenpunkte nicht bereits gelabelt oder klassifiziert sind.

Es gibt mehrere Möglichkeiten, wie Vektoren mittels Clustering gruppiert werden. Im Folgenden stelle ich drei von ihnen vor.

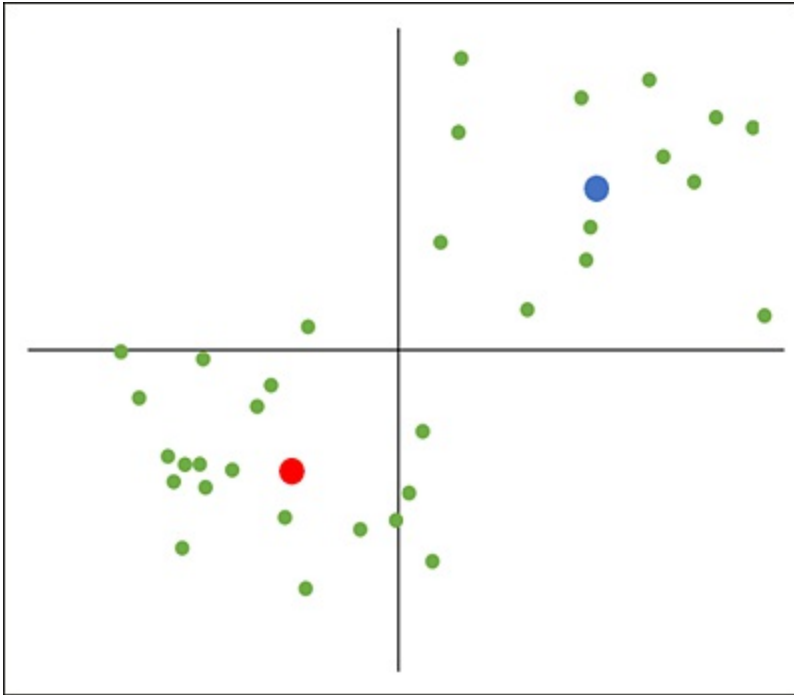
- **k-Means-Algorithmus:** Der *k*-Means-Algorithmus, auch *k*-Means-Clustering genannt, ist eine unüberwachte Lernmethode. Dafür liegt eine Vielzahl an Daten in Vektorform vor – z. B. als Punkte, die im Raum verteilt sind, wie wir es von unseren Graphen aus den vorangegangenen Beispielen kennen. Nun gibt es sogenannte *k*-Punkte in diesem Raum, also Cluster-Schwerpunkte. Jeder Datenpunkt wird seinem nächstgelegenen Schwerpunkt zugeordnet, wodurch definiert wird, welcher Kategorie er angehört. Anschließend ermitteln wir die beste Position für diese *k*-Punkte,



damit die  $k$  Klassen, in die wir unsere Daten einteilen wollen, bestmöglich aufgeteilt sind. Die Schwerpunkte sollten also so platziert sein, dass sie ihr Cluster (die Summe der Datenpunkte, die ihnen zugeteilt wurde) bestmöglich mit ihrer Position repräsentieren.

Wenn du dir [Abbildung 6.1](#) weiter unten anschaust, siehst du die kleinen Punkte, die unsere Daten repräsentieren. Die Merkmalsvektoren sind für dieses Beispiel zweidimensional, daher gibt es nur die zwei Achsen. Es wird auf den ersten Blick klar, dass es zwei Gruppierungen der Daten gibt: eine oben rechts und eine links unten. Die beiden größeren Punkte sind die Cluster-Schwerpunkte der beiden Gruppen. Die Punkte, die näher am oberen Schwerpunkt sind, werden diesem zugeteilt, und die Punkte, die näher am unteren Schwerpunkt sind, werden diesem zugeteilt. Diese Schwerpunkte werden durch Iteration gefunden.

Dass es nur zwei Gruppen gibt, ist für dieses Beispiel offensichtlich. In der Praxis wird das nicht immer so sein, besonders wenn die Merkmalsvektoren mehr als nur zwei Dimensionen haben!



**Abbildung 6.1** In einem zweidimensionalen Diagramm werden Datenpunkte sowie zwei Cluster-Schwerpunkte dargestellt.

- **Dichtebasiertes Clustering:** Bei dieser Methode werden die Datenpunkte nach der Dichte der Punkte zueinander gruppiert. Die Cluster werden durch Bereiche getrennt, in denen die Dichte geringer ist. Beim dichtebasierten Clustering sind die Grenzen zwischen den Gruppen sehr flexibel.
- **Selbstorganisierende Karten:** Eine selbstorganisierende Karte ist eine unüberwachte Lernmethode, die Daten mit vielen Dimensionen in anschauliche, typischerweise zweidimensionale Bilder umwandelt, um die Beziehungen zwischen den Datenpunkten zu visualisieren. Stell dir ein Schachbrett vor, auf dessen Felder du Datenpunkte mit ähnlichen Eigenschaften legst. Du definierst diese Eigenschaften nicht, sie sind vielmehr Teil des Algorithmus und

organisieren sich selbst. Sehen wir uns ein Beispiel für diese Methode an!

## 6.3 Aus dem Leben gegriffen: Ein Beispiel

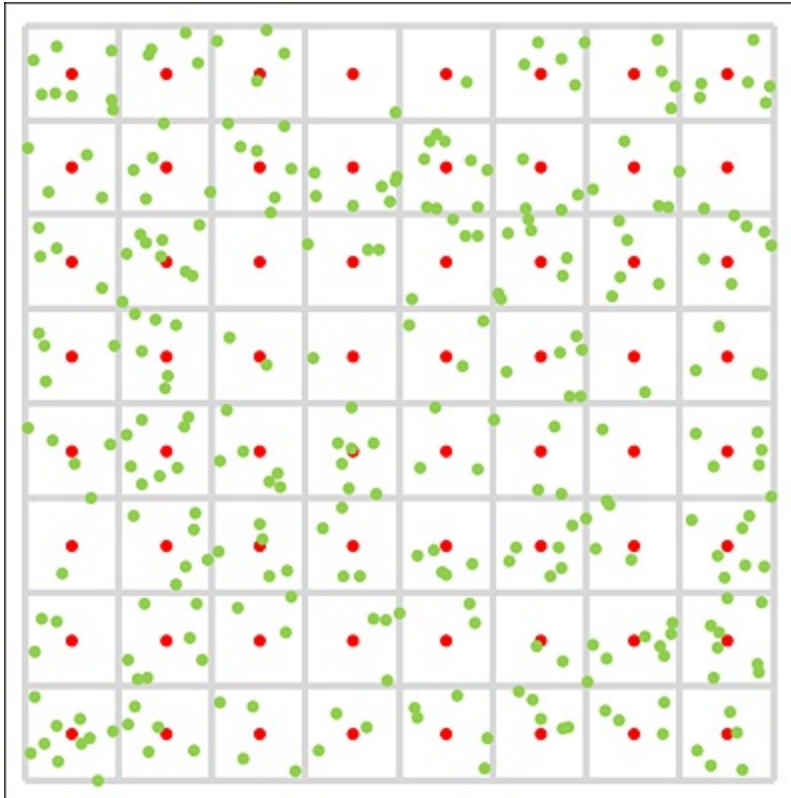
Schauen wir uns die selbstorganisierenden Karten mal im Rahmen eines Beispiels genauer an. Dazu widmen wir uns den Abstimmungsprotokollen des britischen Unterhauses (*House of Commons*). Aber bevor wir uns die Ergebnisse irgendwelcher Abstimmungen ansehen, möchte ich erklären, warum *selbstorganisierende Karten* (SOM, nach dem englischen Begriff *Self-Organizing Map*) überhaupt so besonders nützlich sind.

In allen bisherigen Beispielen und Projekten habe ich mich auf Fälle konzentriert, für die wir anschauliche Bilder zeichnen können. Dafür musste ich mich auf Merkmalsvektoren mit maximal zwei Dimensionen beschränken. Solche unterkomplexen Beispiele sind sehr selten im Machine Learning. Merkmalsvektoren können nämlich absolut jede Länge haben – ob Hunderte oder sogar Tausende von Zeilen pro Datenpunkt. Ich habe keine Ahnung, wie viele Daten Facebook so durchschnittlich von einem einzelnen User oder einer Userin sammelt, aber ich bin ziemlich sicher, dass ein davon abgeleiteter Merkmalsvektor eine enorme Größe hätte.

Nehmen wir mal an, wir analysieren einen Fall mit 100 Merkmalen. Wir können jetzt offensichtlich kein 100-dimensionales Diagramm erstellen. Klar, wir können uns irgendwelche Analyseergebnisse ausgeben lassen, oder die Abstände zwischen den Vektoren. Wir

können unsere Daten auch in Gruppen einteilen, aber wir können all diese Vorgänge nicht einfach so *visualisieren*.

An dieser Stelle kommen die selbstorganisierenden Karten ins Spiel.



**Abbildung 6.2** In einem zweidimensionalen Diagramm werden Datenpunkte und ihre zugehörigen repräsentativen Vektoren dargestellt.

[Abbildung 6.2](#) veranschaulicht, wie SOM funktionieren. Eine kurze Erinnerung: Ein Merkmalsvektor enthält beliebig viele Einträge, es müssen nicht zwingend zwei Dimensionen sein. Lass dich also nicht von der zweidimensionalen Struktur der Darstellung täuschen!

Die grünen Punkte entsprechen den *Datenpunkten*. Die roten Punkte sind *repräsentative Vektoren*. Die grünen Datenpunkte werden immer

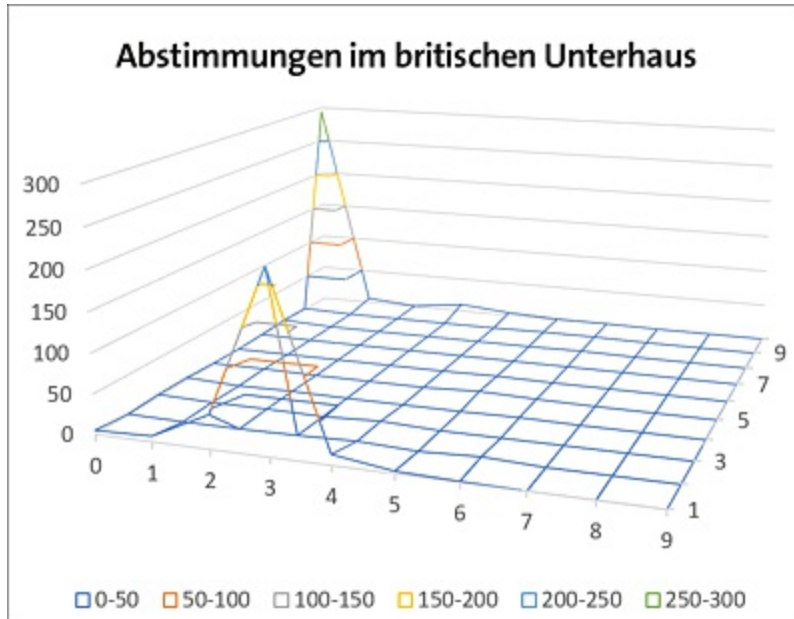
mit demjenigen repräsentativen Vektor in ein Kästchen gelegt, dem sie am nächsten sind. Nun enthalten einige Kästchen sehr viele grüne Punkte und andere sehr wenige. Dadurch wird leicht ersichtlich, welche Datensätze sich ähnlich sind (sie liegen im Raster nahe beieinander) und welche sehr unterschiedlich sind (sie liegen weit auseinander).

Aber woher kommen diese repräsentativen Vektoren? Sie werden durch eine iterative Technik gefunden, deren Erklärung den Rahmen dieses Buches sprengen würde. Dieser Vorgang ist das »selbst« im Begriff »selbstorganisierende Karte«.

Jetzt aber zum richtigen Beispiel!

In [Abschnitt A.2](#), »Datensammlung«, steht noch ausführlicher, wo die Informationen über die Abstimmungsergebnisse im *House of Commons* zu finden sind. Die Daten zeigen, wie jedes Mitglied des Parlaments (MP) bei jeder Abstimmung abgestimmt hat. Du kannst dir also jedes MP als einen Merkmalsvektor mit ungefähr 100 Einträgen vorstellen. Der oberste Eintrag ist dabei die erste Abstimmung an einem bestimmten Datum, der zweite die nächste Abstimmung und so weiter. Wenn mit »Aye« (Ja) gestimmt wurde, ist der Eintrag im Vektor +1, wenn mit »No« (Nein) gestimmt wurde, ist er -1. Bei einer Enthaltung oder Nicht-Abstimmung ist der Eintrag 0. Für jedes MP repräsentieren die Einträge demnach genau die gleichen Abstimmungsanlässe. Es gibt insgesamt 650 Abgeordnete. All diese und ihre 100-dimensionalen Merkmalsvektoren möchten wir in einem

zweidimensionalen Diagramm darstellen, um zu sehen, wie ähnlich oder abweichend voneinander sie abgestimmt haben.



**Abbildung 6.3** Die Abstimmungen im britischen Unterhaus, in einem zweidimensionalen Diagramm dargestellt

Na ja, wie du in [Abbildung 6.3](#) siehst, sind die Ergebnisse ziemlich vorhersehbar! Die Höhe der Oberfläche des Diagramms wird durch die Anzahl der Abgeordneten in jedem Kästchen des Rasters bestimmt. Je höher die Oberfläche also angelegt wird, desto mehr MPs haben auf diese Art abgestimmt. Es gibt also wirklich fast nur zwei Fraktionen von Abgeordneten – diese entsprechen den beiden größten Parteien. Wenn man ganz genau hinschaut, kann man im Diagramm erkennen, dass es ein paar wagemutige Abgeordnete gibt, die nicht zu 100 % mit ihren Parteikolleg\*innen übereinstimmen, aber bei diesen handelt es sich nur um eine Minderheit.

Im folgenden Projekt wird du nicht mit so etwas Komplexem wie SOM konfrontiert, aber du wirst ganz schön nahe dran kommen.



## 6.4 Dein Projekt

Für dieses Projekt sammeln wir die Art von Daten, die auch Netflix von seinen Kunden und Kundinnen erhebt. Netflix verwendet sie anschließend, um zu entscheiden, welche Filme oder Serien ihnen als nächste auf der Startseite oder als Empfehlung angezeigt werden. Du brauchst für dieses Projekt eine Gruppe von Menschen, sagen wir 20 Leute, die jeweils für eine gewisse Anzahl an Filmen angeben, ob sie sie mochten oder nicht. Eine weitere Person wird die Rolle des Moderators oder der Moderatorin übernehmen.

1. Der Moderator bzw. die Moderatorin schreibt 10 Filme auf, von denen er oder sie denkt, dass die Gruppe sie kennt.
2. Diese Filme sollten verschiedene Genres umfassen: Thriller, Animationsfilm, Liebeskomödie, Musical, Historiendrama usw.
3. Jetzt wird für jedes Gruppenmitglied ein Bewertungsbogen ausgedruckt. Er sollte in etwa so aussehen wie das folgende Beispiel:

Film	-1 / 0 / +1
Matilda	
Die Eiskönigin	

E.T.	
Mary Poppins	
...	

4. Jetzt bewerten die einzelnen Personen jede\*r für sich, ob sie den jeweiligen Film mochten (+1), ob sie ihn nicht mochten (−1) oder ob sie keine Einstellung zum Film haben bzw. den Film nicht gesehen haben (0).
5. Unter der ausgedruckten Liste ergänzen dann alle Teilnehmenden noch drei weitere Filme, die nicht in der Liste aufgeführt sind, die sie aber besonders toll fanden.
6. Der Moderator bzw. die Moderatorin füllt die Liste ebenfalls aus.
7. Er oder sie füllt darüber hinaus noch drei weitere Zettel aus. Sie sollen repräsentativ für unterschiedliche Filmgeschmäcker stehen. Beispielsweise könnte auf einer Liste +1 für alle animierten Filme vergeben werden und −1 für den Rest, oder es gibt +1 für alle Musicals und −1 für Filme ohne Filmmusik. Ein Actionfilm-Fan würde vielleicht +1 für alle Actionfilme vergeben und −1 für alle anderen. Diese drei Zettel werden zusammen mit dem ersten vom Moderator oder der Moderatorin ausgefüllten Zettel die repräsentativen Vektoren bilden, an denen wir die Abstände aller Vektoren der Gruppe messen werden.

8. Bis zu diesem Zeitpunkt sind die Ergebnisse auf den Zetteln, die der Rest der Gruppe ausgefüllt hat, immer noch nicht bekannt und sie werden auch weiterhin nicht gezeigt.
9. Der/Die Moderierende zeigt nun die vier repräsentativen Vektoren (ohne zu verraten, welcher sein oder ihr eigener ist). Dies sind die A-, B-, C- und D-Cluster. Anschließend wird jede Ecke des Raumes, in dem der Test durchgeführt wird, mit A, B, C oder D beschriftet.
10. Jedes Gruppenmitglied misst nun die Entfernung seines eigenen Vektors zu jedem der vier repräsentativen Vektoren. Dafür kann einfach die Manhattan-Distanz verwendet werden, die wir weiter vorne im Buch schon kennengelernt haben, da sie am einfachsten zu berechnen ist. Bei 10 Filmen auf der Liste können die Ergebnisse der Abweichung von 0 (die eigene Liste stimmt genau mit den Ergebnissen der repräsentativen Vektoren überein) bis 20 (die eigene Liste bewertet die Filme genau gegenteilig) reichen.
11. Jedes Gruppenmitglied weiß also nun, welcher repräsentative Vektor am nächsten an seinem eigenen liegt. Es ist einfach derjenige mit der kürzesten Entfernung (dem niedrigsten Wert, der mithilfe der Manhattan-Distanz berechnet wurde).
12. Nun begeben sich alle in die Ecken des Raumes, die nach dem repräsentativen Vektor benannt sind, dem ihre Daten am nächsten kommen.

Jetzt wird es Zeit, die Ergebnisse zu diskutieren! Vergleiche deinen

Vektor mit denen der anderen Teilnehmenden in deinem Cluster. Haben befreundete Personen ähnliche Ergebnisse? Was halten die Actionfilm-Fans von Musicals? Sind einige Ecken wesentlich besser gefüllt als andere?

Jetzt könnte auch der Moderator oder die Moderatorin preisgeben, welches sein oder ihr persönlicher Vektor ist (der ja auch durch eine Ecke des Raumes repräsentiert wird).

Raum für Diskussion bieten nun auch die drei Filme, die alle am Ende ihrer Liste notieren sollten. Gibt es hier Ähnlichkeiten unter den Menschen, die in derselben Ecke stehen?

Zum Schluss möchte ich dir auch noch eine kleine Hausaufgabe mit auf den Weg geben: Schau dir doch einfach mal einen der Filme an, die jemand aus deiner Ecke am Ende seiner Liste empfohlen hat und die du noch nicht kennst.

## Kapitel 7

# Mehr zu: Entscheidungsbäumen

Entscheidungsbäume sind eine überwachte Lernmethode. Sie funktionieren ähnlich wie ein Flowchart oder eine Partie »Wer bin ich?«.

Bei »Wer bin ich?« geht es darum, in 20 Fragen oder weniger die richtigen Ja/Nein-Fragen zu stellen, um zur korrekten Antwort zu gelangen. Machine Learning verfolgt ein ähnliches Ziel. Es gibt einen klassifizierten Trainingsdatensatz, der verwendet wird, um die bestmögliche Baumstruktur von Fragen und Antworten für das Modell zu erstellen. So kann, sobald ein neuer Datensatz klassifiziert werden soll, diese Klassifizierung schnell und genau mithilfe der Struktur erfolgen.

Der Unterschied zu »Wer bin ich?« ist dabei: Die Fragen in einem Entscheidungsbaum müssen nicht binär sein (wie im Falle von *Ja* oder *Nein*) und die Antworten können numerisch sein.

Bevor wir richtig ins Thema einsteigen, müssen wir zuerst ein bisschen über Fachbegriffe und Grundvoraussetzungen sprechen: Der Entscheidungsbaum wird auf den Kopf gestellt dargestellt, die *Wurzel* (also die erste Frage) ist dabei oben. Jede Frage auf dem Weg nach unten ist eine Eingrenzung, die Merkmale oder Attribute in kleinere Gruppen aufteilt. Von jedem Knoten aus gibt es *Äste*, die die möglichen Antworten auf die Frage darstellen. Wenn du am Ende eines Pfades durch den Baum angelangt bist, sodass es keine weiteren Entscheidungen mehr gibt, dann bist du an einem *Blatt* angekommen.

Wie ein Entscheidungsbaum wächst bzw. aufgebaut ist, wird durch die Reihenfolge bestimmt, in der die Merkmale deiner Daten abgefragt und eingegrenzt werden.

## 7.1 Wofür Entscheidungsbäume eingesetzt werden

- **Data Mining:** Entscheidungsbäume können verwendet werden, um Muster in umfangreichen und komplexen Datensätzen zu entdecken. So kann man sie beispielsweise zur Kundensegmentierung, zur Warenkorbanalyse oder zur Betrugserkennung einsetzen.

- **Entscheidungsfindung:** Mithilfe von Entscheidungsbäumen können Szenarien visualisiert und analysiert werden. Die Grundlage dafür bildet eine Reihe von bereits getroffenen Entscheidungen, von denen dann weitere Entscheidungen abgeleitet werden. Das macht sie zu einem wertvollen Werkzeug für medizinische Diagnosen, die Investitionsplanung oder die Erstellung einer Spielstrategie.
- **Merkmalsauswahl:** Aus einer Reihe von Merkmalen, die ein Datensatz hat, können die wichtigsten selektiert und irrelevante oder redundante eliminiert werden. So tragen Entscheidungsbäume zu Vorhersagen über die Beschaffenheit der Zielvariablen bei – ihre Merkmale können vorab bewertet und eingegrenzt werden.
- **Erklärbarkeit:** Weil Entscheidungsbäume visuell leicht zu erschließen sind und verständlich dargestellt werden können, können mit ihrer Hilfe transparente Übersichten für die vom Modell gemachten Vorhersagen erstellt werden.
- **Mustererkennung:** Entscheidungsbäume können verwendet werden, um Bilder, Texte oder Sprache auf der Grundlage ihrer Merkmale zu klassifizieren. So können sie zum Beispiel für die Gesichtserkennung, zur Stimmungsanalyse oder zur Spracherkennung eingesetzt werden.
- **Sport:** Mithilfe von Entscheidungsbäumen kann der Ausgang von Spielen vorhergesagt werden.
- **Bildung:** Auch die Leistung von Schülern und Schülerinnen kann mit der Unterstützung von Entscheidungsbäumen eingeschätzt

werden.



## 7.2 Im Detail

Ich habe online eine Liste der beliebtesten Schauspieler\*innen aller Zeiten gefunden. Sie wurde auf der Website von YouGov veröffentlicht, einem Umfrageunternehmen. Die URL zur Seite findest du in [Abschnitt A.2](#) in der Datensammlung am Ende dieses Buches. Ich wähle im Folgenden eine Person aus den Top 100 aus und du hast, wie bei einer Partie »Wer bin ich?«, 20 Fragen, um herauszufinden, an wen ich denke. Wie gehst du vor? Fragst du zuerst, ob die Person noch lebt? Oder ist es besser, zu fragen, ob sie weiblich oder männlich ist? Fragst du vielleicht sogar direkt einzelne Namen ab, die du für wahrscheinlich hältst?

Ich gebe dir hier mal ein paar Informationen, damit du besser entscheiden kannst, welche Frage du zuerst stellen solltest – welche Frage also an der Wurzel deines Entscheidungsbaums stehen sollte.

Von den 100 Schauspieler\*innen sind 70 männlich und 30 weiblich. (Hey, ich habe diese Liste nicht zusammengestellt!) Von den 100 sind außerdem noch 78 am Leben, 22 sind schon tot. (*Hätte* ich die Liste erstellt, hätte ich auf jeden Fall mehr Personen aus dem Golden Age von Hollywood ausgewählt!)

Welche ist jetzt also die beste Frage, die man zuerst stellen kann?

Wenn du dich für die Frage nach lebendig/tot entscheidest, könntest du Glück haben, wenn die Antwort *tot* lautet. In dem Fall hast du nämlich sofort 78 % der Möglichkeiten ausschließen können. Andererseits wirst du halt mit einer Wahrscheinlichkeit von 78 % nur 22 % der Möglichkeiten ausschließen. Hmm.

Bei der Frage nach männlich/weiblich liegen die Wahrscheinlichkeiten bei 70 % und 30 %. Also schließt du zu 70 % auch nur 30 % der Möglichkeiten aus.

Bei dieser Art von Problem kann es helfen, besonders extreme Beispiele zu verwenden, um das Prinzip hinter der Fragestellung besser zu verstehen. (Das ist eine sehr gängige Technik in der Mathematik.) Angenommen, du fragst mich direkt: »Ist es Audrey Hepburn?« Dann wirst du fast sicher falschliegen und hast damit eine Frage verschwendet, ohne Fortschritte auf dem Weg zur Lösung gemacht zu haben. In 99 % der Fälle schließt du nur 1 % der möglichen Antworten aus. Wenn du jetzt weiterhin einzelne Namen rätst, wirst du ziemlich wahrscheinlich all deine 20 Fragen aufbrauchen und trotzdem nicht erfahren, an wen ich gedacht habe.

Wenn wir all die Informationen in Betracht ziehen, die dir bisher vorliegen (es geht weiter darum, ob die Frage lauten sollte »Ist die Person weiblich/männlich?« bzw. »Ist die Person tot/lebendig?«), dann solltest du fragen: »Ist die Person männlich?« In der Antwort auf diese Frage schwingen nämlich noch weitere Informationen mit. Der Begriff *Information* hat eine technisch-mathematische Bedeutung, aber darauf einzugehen, ginge an dieser Stelle wirklich zu weit. Es

gibt dafür ein ganzes Fach namens »Informationstheorie«, das ursprünglich vom Jongleur, Einradfahrer und – ja, auch das – Mathematiker, Claude Shannon entwickelt wurde.

Es gibt tatsächlich eine noch bessere Frage, aber das kannst du mit den vorliegenden Informationen nun wirklich nicht wissen. Wenn du nämlich fragst: »Enthält der Name der Person den Buchstaben L?«, dann gibt es genau 50 Namen, die ein »L« enthalten und 50, die es nicht tun. Das wäre eine sehr nützliche Information gewesen, denn so hättest du in 100 % der Fälle ganze 50 % der Möglichkeiten ausschließen können. Wenn du mit jeder einzelnen Frage die Hälfte der möglichen Personen ausschließen könntest, kannst du das kleine Rätsel hier innerhalb kürzester Zeit lösen. Na ja gut, eigentlich in sieben oder acht Fragen. Die mathematischen Hintergründe dazu präsentiere ich dir gleich noch.

Ich weiß natürlich, dass es total um die Ecke gedacht ist, ausgerechnet nach dem Buchstaben »L« im Namen der Person zu fragen. Aber aus pädagogischen Gründen musste ich zumindest *eine* Eigenschaft in der Liste der Schauspieler\*innen finden, die mir eine 50/50-Aufteilung ermöglichen würde. (Ich hatte Glück, dass ich überhaupt eine gefunden habe. Mein Dank geht an dieser Stelle wie so oft an Microsoft Excel.)

Es steckt viel Mathematik hinter dem Prinzip, nach dem Entscheidungsbäume funktionieren. Eines fernen Tages, wenn du dieses Fach tatsächlich an der Uni studierst, wirst du jede Menge

Details über Logarithmen, Entropie und Informationstheorie wissen müssen. Aber nicht heute.

Ach ja – ich habe übrigens wirklich an Audrey Hepburn gedacht.

## 7.3 Aus dem Leben gegriffen: Ein Beispiel

Hier kommt jetzt also endlich noch der mathematische Hintergrund zu unserem Beispiel mit den 20 Fragen, das wir im Kapitel zu Clustering kennengelernt haben.

Ich denke an eine ganze Zahl von 1 bis 32. Wie findest du jetzt am besten heraus, welche Zahl gemeint ist? Du kannst mir jede beliebige Frage stellen:

»Ist es 3?« – »Nein.«

»Ist es 27?« – »Nein.«

»14?« – »Nein.«

Das könnte jetzt noch ein Weilchen dauern ...

Du könntest Glück haben und direkt beim ersten Mal die richtige Zahl treffen. Du könntest aber auch Pech haben und erst beim zweiunddreißigsten Mal richtig liegen. Wie viele Fragen würdest du im Durchschnitt wohl brauchen?

Bei solchen Fragen kann es hilfreich sein, mit kleineren Zahlen zu beginnen.

Angenommen, ich denke an eine 1 oder eine 2. Es gibt also eine 50-prozentige Chance ( $\frac{1}{2}$ ), dass du beim ersten Mal richtig rätst. Ebenso

hoch ist die Chance, zwei Fragen zu benötigen, um das richtige Ergebnis zu erraten. Das ergibt im Durchschnitt:  $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1 \frac{1}{2}$ . Jetzt versuchen wir das Ganze mal für die Zahlen 1, 2 und 3. Im Durchschnitt wirst du zwei Fragen brauchen, wenn du bei jedem Versuch eine konkrete Zahl erfragst. Das ist also hier immer noch sinnvoll. Bei vier Zahlen sind es  $2 \frac{1}{2}$  Versuche. Bei  $n$  Zahlen brauchst du im Durchschnitt  $\frac{1}{2} (n+1)$  Fragen. Bei 32 Zahlen ergibt das also eine durchschnittliche Fragen-Anzahl von  $16 \frac{1}{2}$ .

Geht das noch besser?

Nehmen wir mal an, wir schauen uns die Zahlen 1 bis 4 an. Wenn wir fragen, ob die Zahl 2 oder kleiner ist, und wir die Antwort »Nein« bekommen, dann haben wir die korrekte Antwort auf die Zahlen 3 und 4 eingegrenzt. An diesem Punkt raten wir einfach mal, dass es 3 ist. Wenn wir richtig liegen, dann haben wir es erraten, wenn wir falsch liegen, sagen wir jetzt »4« und sind fertig. Das ist jetzt also ein Durchschnitt von  $2 \frac{1}{2}$ . Das ist genau der gleiche Wert, den wir erreicht haben, als wir die Zahlen zufällig ausgewählt haben. Aber anstatt zwischen einer und vier Fragen zu benötigen, brauchen wir jetzt mindestens zwei und höchstens drei.

Wenn du dieses Prinzip mit den Zahlen 1 bis 32 wiederholst, erreichen wir mit unserer Aufteilungsmethode einen Durchschnitt von  $5 \frac{1}{2}$ , mit im besten Fall fünf und im schlechtesten Fall sechs Fragen nach dem Ergebnis. Das steht im krassen Gegensatz zum einzelnen Auswählen der Zahlen, bei dem wir einen stolzen Durchschnitt von 16

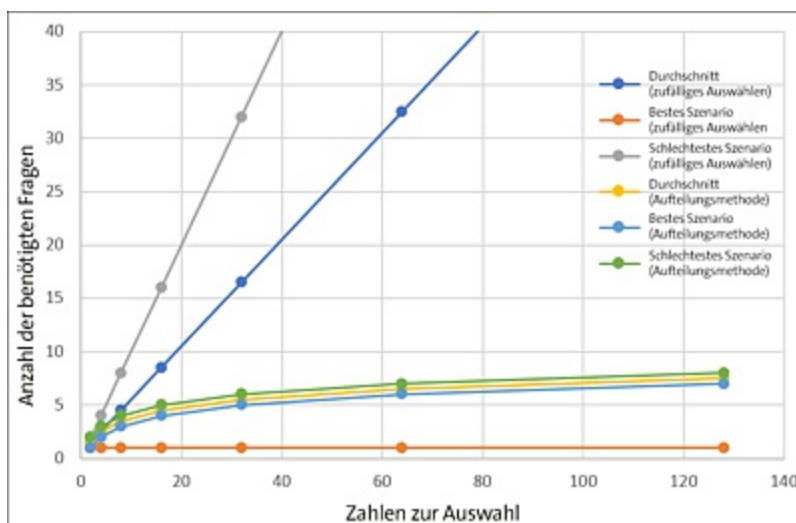
$\frac{1}{2}$  mit im besten Fall einer und im schlechtesten Fall 32 Fragen erzielt haben.

Ein Teil des Entscheidungsbaums für diesen Algorithmus ist in [Abbildung 7.1](#) dargestellt.



**Abbildung 7.1** Der Entscheidungsbaum für die Suche nach einer zufälligen ganzen Zahl zwischen 1 und 32

Die folgende [Abbildung 7.2](#) zeigt, wie der errechnete Durchschnitt an Fragen sowie der beste und der schlechteste Fall von der Spanne der Zahlen abhängen, die zur Auswahl stehen. Es ist klar, dass das zufällige Auswählen ganzer Zahlen nicht der ideale Weg ist – es sei denn, du weißt von dir, dass du bei so etwas einfach unfassbares Glück hast.



**Abbildung 7.2** *Die Anzahl an Fragen (auf der y-Achse), die zum Erraten einer Zahl (x-Achse) benötigt werden, bei steigender Menge möglicher Zahlen*

Zum Abschluss kommt hier noch eine Frage für dich: Wenn wir nur 20 Fragen stellen dürfen – wie viele Zahlen dürfen dann maximal auswählbar sein, um mit dieser Aufteilungsmethode überhaupt noch zu einem Ergebnis zu kommen?

Ganz einfach:  $2^{20} = 1,048,576$ . Wow!



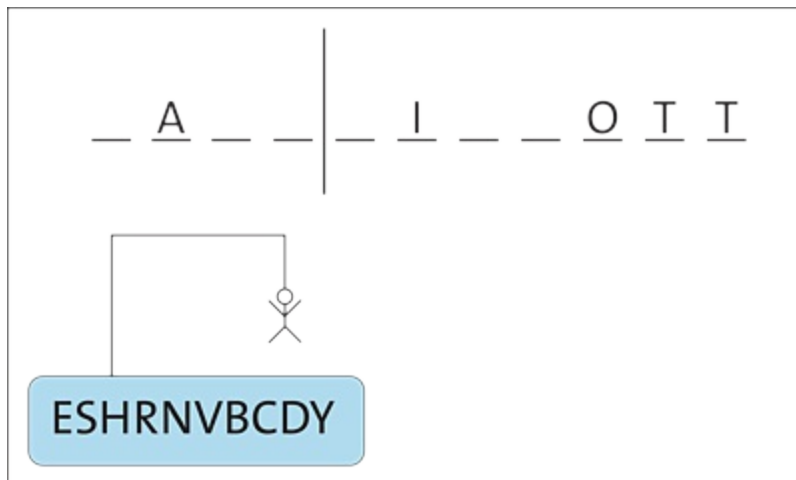
## 7.4 Dein Projekt

Nach all deiner getanen, harten Arbeit darfst du dir jetzt endlich eine Entspannungspause gönnen – und ein paar Spiele spielen!

Na gut, die Spiele sind natürlich wieder mit der Entscheidungsbaum-Methodik verbunden, aber hey! Entspannung ist Entspannung.

### 7.4.1 Galgenmännchen

Das erste Spiel, um das es hier gehen soll, ist *Galgenmännchen*. Du wählst nacheinander Buchstaben aus, um ein Wort zu vervollständigen. Wenn die ausgewählten Buchstaben im Wort enthalten sind, werden sie an die richtige Stelle geschrieben. Wenn nicht, wird eine Skizze einer gehängten Person erstellt – Strich für Strich. Damit du dieses Spiel gewinnen kannst, brauchst du eine ungefähre Vorstellung davon, welche Buchstaben des Alphabets wie oft in deutschen Texten vorkommen.



**Abbildung 7.3** Finaler Stand einer Partie »Galgenmännchen« – das Galgenmännchen wurde vervollständigt, bevor die Lösung erraten werden konnte.

In seiner Kurzgeschichte von 1843, *Der Goldkäfer*, schrieb Edgar Allan Poe: »Im Englischen ist der meistverwendete Buchstabe das E. Danach folgen in absteigender Reihenfolge: a o i d h n r s t u y c f g l m w b k p q x z. Das E ist in seiner Häufigkeit allerdings so vorherrschend, dass es kaum einen Satz von beliebiger Länge gibt, in dem es nicht der meistverwendete Buchstabe ist.«

Der Protagonist in dieser Geschichte benutzte die Häufigkeitsliste, um einen kodierte Brief zu übersetzen.

Diese Reihenfolge ist allerdings nicht mehr ganz aktuell. Nach neueren Analysen ist die Reihenfolge im Englischen: e t a o i n s h r d l c u m w f g y p b v k j x q z. Für die deutsche Textsprache ist es: e n i s r a t d h u l c g m o b w f k z p v ß j y x q. Solche Informationen sind beim Spielen von Galgenmännchen natürlich sehr hilfreich, da sie zumindest einige gute Wahlmöglichkeiten für erste Vorschläge mit sich bringen. Man beginnt besser nicht mit z oder q, sondern eher mit

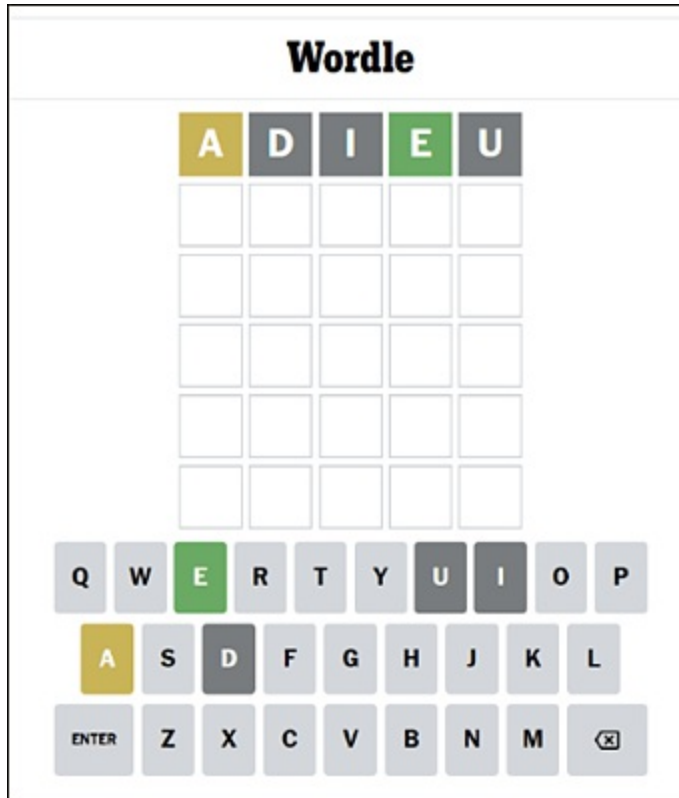
*e*, *n* und *i*. Danach kann man dann immer noch Vermutungen über die Häufigkeit gewisser Buchstabenpaare, die Platzierung von Vokalen usw. anstellen. Für das Spiel Galgenmännchen kann man sich leicht einen Entscheidungsbaum vorstellen (auch wenn er nicht besonders repräsentativ für Entscheidungsbäume allgemein wäre). Beim Spielen hilft es auch, zu wissen, wie schlau dein Gegner oder deine Gegnerin ist. Weiß er oder sie auch über die Buchstabenhäufigkeiten Bescheid und wird er bzw. sie dieses Wissen ausnutzen, um ein möglichst kompliziertes Wort zu präsentieren?

Wie könnte nun der Entscheidungsbaum für Galgenmännchen aussehen? Welcher wäre der erstgenannte Buchstabe, der ganz oben steht? Und wie groß wäre der gesamte Baum?

## 7.4.2 Wordle

Auf eine ähnliche Weise funktioniert das Spiel *Wordle* der *New York Times*. Dabei musst du ein englisches Wort mit fünf Buchstaben finden. (Es gibt mittlerweile auch eine deutsche Version unter dem Namen *Wördle*.) Du hast sechs Versuche, das Wort zu erraten, und für jeden Versuch wird ein Feedback in Form farbiger Kacheln gegeben, die anzeigen, ob ein verwendeter Buchstabe im Wort vorkommt und ob er die richtige Position einnimmt. Jeder Versuch muss ein tatsächlich existierendes Wort sein, du kannst nicht einfach zufällige Buchstaben einsetzen wie bei Galgenmännchen. Es gibt

insgesamt 12.972 Wörter, die du zum Raten einsetzen könntest, aber nur 2.315 Lösungswörter.



**Abbildung 7.4** Ein Wordle-Spiel nach dem Eingeben des ersten geratenen Wortes – der Buchstabe A kommt vor, steht aber noch nicht an der richtigen Stelle im Wort.

Auch für dieses Spiel ist die Lösung über einen Entscheidungsbaum möglich. Gibt es ein bestes Wort für den ersten Eintrag? Eines, das die nützlichsten Informationen für den Anfang liefert? Laut *Scientific American* sind beliebte erste Wahlmöglichkeiten AUDIO und ADIEU, weil sie viele Vokale enthalten. Online wirst du viele Forumsdiskussionen dazu finden – und in dem Zusammenhang auch den Bezug auf Informationstheorie und Entropie. Ein Vorschlag für die erste Zeile ist dort das Wort SALET – das ist ein rundlicher Ritterhelm,

der bis über den Hinterkopf reicht. Es gibt allerdings immer wieder auch das Gegenargument, dass man sich mit dem Wissen darum das Spiel selbst verdirbt. Für uns ist die Entscheidung für das erste Wort ohnehin nicht so spannend – denn wir bräuchten zum Lösen des Rätsels auch noch den Entscheidungsbaum, der uns dann sagt, welches das beste *zweite* Wort zum Raten ist.

### 7.4.3 Mastermind

Das letzte Spiel, über das wir heute reden sollten, ist *Mastermind*, ein Codeknacker-Spiel aus den 1970er-Jahren für zwei Personen. Das Mastermind-Brett besteht aus zehn Reihen mit je vier Löchern (siehe [Abbildung 7.5](#)). Eine der beiden mitspielenden Personen gibt den Code der Partie vor, indem sie in die vier Löcher am Ende des Bretts vier farbige Stifte steckt. Diese sind aber durch einen kleinen Sichtschutz vor dem anderen Spieler bzw. der anderen Spielerin verborgen. Diese\*r hat nun die Aufgabe, den Code zu knacken. Das Originalspiel hat sechs verschiedenfarbige Stifte – je nach gewünschtem Schwierigkeitsgrad kann vorgegeben werden, dass alle Stifte, die im Code verwendet wurden, unterschiedliche Farben haben müssen. Genau wie bei Wordle gibt es Hinweise darauf, welche Stifte sich hinter dem Sichtschutz befinden.

Beim Codeknacken äußerst du eine Vermutung, indem du farbige Stifte in die erste Reihe steckst. Die andere Person gibt dann an, wie

viele Stifte die richtige Farbe haben und in der richtigen Position stecken – und wie viele die richtige Farbe haben, aber noch nicht an der richtigen Position stecken. Das passiert über eine Extrareihe auf dem Spielbrett und schwarze und weiße Stifte. Schwarz wird für die richtige Farbe an korrekter Position verwendet, Weiß für die richtige Farbe an falscher Position.



**Abbildung 7.5** Ein möglicher Spielstand einer Partie »Mastermind«

Natürlich steckt auch hinter diesem einfachen Spiel eine Menge Mathematik.

Mit vier Stiften, sechs Farben und unter der Prämisse, dass man Muster auch wiederholt ausspielen kann, gibt es  $6 \cdot 6 \cdot 6 \cdot 6 = 1.296$

mögliche Muster. Spiel dieses Spiel doch einfach ein paarmal, damit du ein Gefühl dafür bekommst, wie viele Versuche du brauchen könntest, um den Code zu knacken.

Laut Donald Knuth kann das Spiel in maximal fünf Zügen gelöst werden. Aber dafür müsstest du genau der von ihm entwickelten Strategie folgen. Der erste Versuch muss zwei Farben jeweils doppelt enthalten. Wenn also beispielsweise Blau und Gelb zwei der insgesamt sechs Farben sind, solltest du zunächst das Muster *Blau, Blau, Gelb, Gelb* mit den Stiften stecken. Du kannst zwei beliebige Farben verwenden – aber sie müssen im ersten Versuch je zweimal vorkommen. Danach wird es kompliziert!

Donald Knuths Strategie ist ein Entscheidungsbaum, der auf dem *Minimax-Prinzip* beruht. Das ist eine Entscheidungsregel, die sich am schlechtestmöglichen Szenario orientiert, das aber *sicher* zu erreichen ist. Es stellt sich also nicht die Frage: »Wie erreiche ich das bestmögliche Ergebnis?« Stattdessen wird eine pessimistische Grundhaltung angenommen: »Wie erreiche ich mit dem niedrigsten Risiko ein Ergebnis, das den Mindestanforderungen genügt?« Mehr Informationen zu dieser Methode findest du für den Anfang auch auf Wikipedia: <https://de.wikipedia.org/wiki/Minimax-Regel>.

## Kapitel 8

# Mehr zu: Neuronalen Netzen

Ein *neuronales Netz* ist eine Art von Machine Learning, die nachahmen soll, wie das Gehirn funktioniert. Mathematisch gesprochen werden numerische Signale von »Neuronen« empfangen, die sie mathematisch verarbeiten und dann an weitere Neuronen weitergeben – das Ganze passiert dann über mehrere Instanzen. Das Input-Signal läuft potenziell durch mehrere Neuronen-Schichten, bevor es als Klassifikation oder Regression in den Output gegeben wird. Keine Sorge, wenn du das jetzt noch nicht alles nachvollziehen konntest: Ich zeige dir gleich auf einigen Abbildungen, wie das Prinzip genau funktioniert.

Neuronale Netzwerke können sowohl für überwachtes als auch für unüberwachtes Lernen verwendet werden.

## 8.1 Wofür neuronale Netze eingesetzt werden

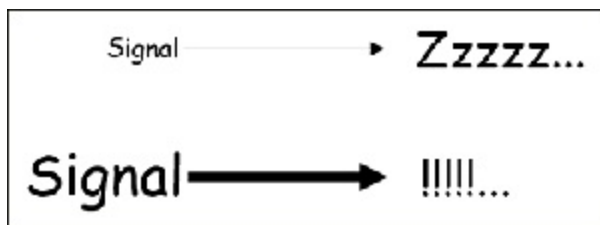


- **Bilderkennung:** Neuronale Netze können dabei helfen, Objekte in digitalen Bildern zu identifizieren. Das findet wiederum Anwendung in einer Vielzahl von Bereichen: Sicherheit, Suchmaschinen, medizinische Diagnosevorgänge, Gesichtserkennung oder auch selbstfahrende Autos.
- **Spracherkennung:** Mithilfe neuronaler Netze können Sprachsignale in Text oder zu Handlungsanweisungen für Maschinen umgewandelt werden. So werden natürliche Sprachverarbeitung, Sprachassistenten, Übersetzungsbots oder die Sprachsynthese unterstützt.
- **Natürliche Sprachverarbeitung:** Natürliche Sprachtexte können analysiert und generiert werden, was essenziell für die Erfüllung von Aufgaben wie Gefühlsanalyse (Sentiment-Analyse), Textzusammenfassungen, maschinelle Übersetzungen, Chatbots oder Textgenerierung ist.
- **Data Mining:** Neuronale Netze können verwendet werden, um Muster in umfangreichen und komplexen Datensätzen aufzudecken. Das ermöglicht die schnelle Erkennung von Anomalien, Betrugserkennung, Kundensegmentierung oder das Entwickeln von Empfehlungssystemen.
- **Prognose:** Basierend auf historischen Trends und anderen Faktoren können neuronale Netze die Werte einer Variablen voraussagen und so zum Beispiel bei der Aktienmarktprognose, bei Wettervorhersagen, Umfrageprognosen oder Verkaufsprognosen unterstützen.

- **Mustererkennung:** Neuronale Netze können verwendet werden, um Daten basierend auf ihren Merkmalen und Labels zu klassifizieren. So werden die Erkennung von Spam, Kreditrisikobewertungen oder die Erkennung von Handschriften ermöglicht.
- **Merkmalsextraktion:** Mithilfe neuronaler Netze können die wichtigsten Merkmale aus einer Gruppe von Rohdaten extrahiert werden, was bei der Erfüllung von Aufgaben wie der Dimensionsreduktion, der Merkmalsauswahl für ein besonderes Projekt oder bei der Entwicklung neuer Merkmale hilfreich sein kann.

## 8.2 Im Detail

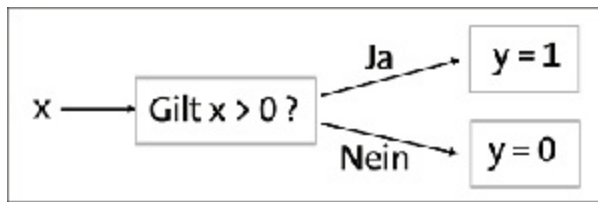
Neuronen im Gehirn kommunizieren mithilfe elektrischer und chemischer Signale miteinander. Wenn ein Neuron Impulse von einem oder mehreren anderen Neuronen erhält, gibt es dieses Signal nur dann an andere Neuronen weiter, wenn das empfangene Signal stark genug ist. Wenn es nicht ausreichend stark ist, wird das empfangene Signal ignoriert.



**Abbildung 8.1** Die Funktionsweise von Neuronen im Gehirn, vereinfacht dargestellt

Das ist, was in einem einzelnen Neuron vor sich geht. Allerdings hat das menschliche Gehirn über 80 Milliarden von ihnen! Und jedes Neuron ist potenziell wiederum mit 10.000 anderen Neuronen verbunden!

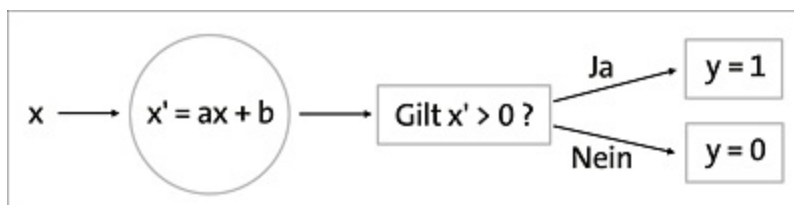
Diese Reaktion eines Neurons auf ein Signal kann auch durch ein schematisches Diagramm dargestellt werden, das die mathematischen Hintergründe des Prozesses deutlich macht.



**Abbildung 8.2** Schematische Darstellung der Funktionsweise von Neuronen im Gehirn

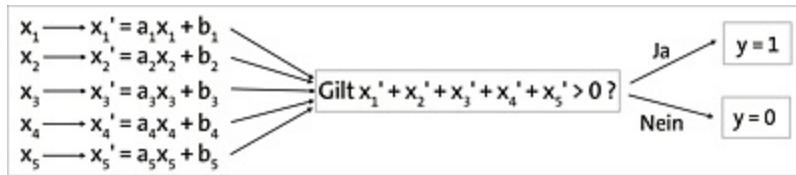
Das eingehende Signal, also unser Input, ist  $x$ , und das ausgehende Signal, der Output, ist  $y$ . Ein positiver Wert für  $x$  zählt als starkes Signal, ein negativer Wert für  $x$  als schwaches.

In einem *künstlichen* neuronalen Netz, also einem, das im Machine Learning eingesetzt wird, ist natürlich noch ein bisschen mehr Mathematik im Spiel. Zunächst wird der Input  $x$  ein wenig modifiziert, indem er mit einem konstanten Parameter  $a$  multipliziert wird und indem dann ein weiterer Parameter  $b$  addiert wird, sodass er zu  $x'$  wird.



**Abbildung 8.3** Schematische Darstellung eines künstlichen neuronalen Netzes mit modifiziertem Input

Allerdings könnte es ja auch gleich einen ganzen Vektor an Inputs geben, also  $x_1, x_2, x_3, \dots$



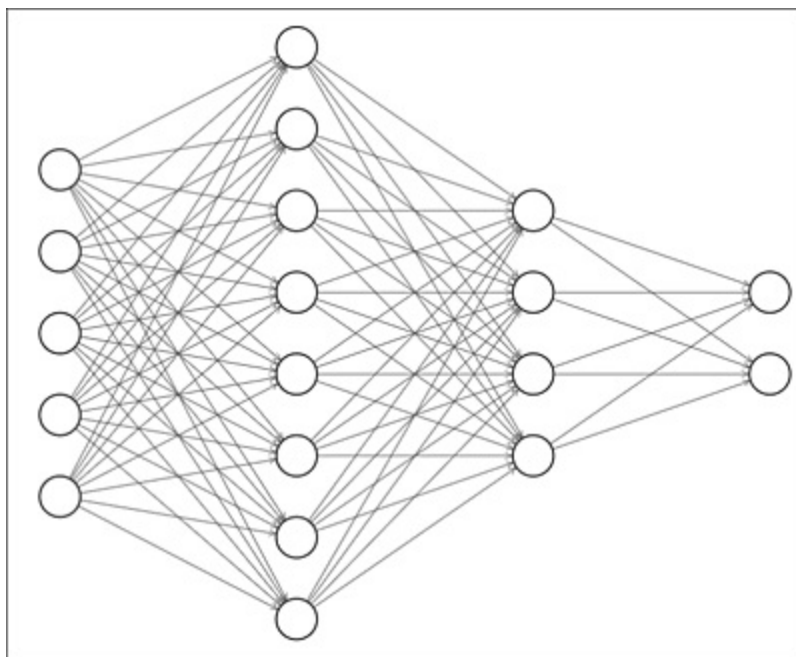
**Abbildung 8.4** Ein neuronales Netz, das einen Vektor als Input verarbeitet

Und wir sind immer noch nicht fertig!

Der Output  $y$  könnte eine andere Funktion durchlaufen haben als nur die 0/1-Selektion aus unserem Beispiel.

Und es könnte mehr als einen Output geben. Der Output könnte beispielsweise auch ein Vektor sein:  $y_1, y_2, \dots$

Und dann könnte es auch noch mehr als eine Ebene von Berechnungen geben ...



**Abbildung 8.5** Schematische Darstellung eines neuronalen Netzes mit mehreren Berechnungsebenen

Das schematische Diagramm in [Abbildung 8.5](#) zeigt ein typischeres neuronales Netzwerk. Dieses ist aber im Vergleich zu in der Praxis verwendeten Netzen immer noch sehr simpel in seiner Struktur. Es hat fünf Inputs auf der linken Seite, es wurde also ein fünfdimensionaler Merkmalsvektor eingegeben. Dem ersten Pfeil folgend, wird nun mit einem Parameter multipliziert und dann wird ein weiterer addiert. Wenn du bei einem der nächsten Kreise angelangst, werden alle Zahlen addiert, die durch die Pfeile von links bei diesem spezifischen Kreis ankommen (je fünf Werte pro Kreis). Anschließend wird dieser Wert verwendet, um eine Funktion auszuführen – zum Beispiel die 0/1-Selektion, die wir oben betrachtet haben. Ist der errechnete Wert eine 1? Dann wird das Signal weitergegeben. Ist er eine 0? Dann wird das Signal nicht weitergegeben. Hier sind aber auch ganz andere Funktionen denkbar.

Nach dieser Ausführung folgen unsere Ergebnisse weiteren Pfeilen, über die der beschriebene Prozess wiederholt wird. Nachdem die Werte alle »verborgenen Schichten«, in unserem Falle zwei, durchlaufen haben, gibt es zwei Outputs. Für unser Beispiel haben wir also einen fünfdimensionalen Merkmalsvektor als Input, zwei verborgene Schichten mit einmal 8 und einmal 4 Knoten sowie zwei Outputs.

Ganz schön anstrengend! Aber mach dir keine Sorgen, du musst das jetzt nicht beim ersten Lesen schon alles lückenlos verstanden haben. Und ganz bestimmt musst du diese ganzen Berechnungen beim nächsten Projekt nicht selbst durchführen!

Weiter oben habe ich ein paarmal das Wort *Parameter* erwähnt. Bei der Netzwerkarchitektur (also der Anzahl der einzelnen Zellen, der verborgenen Schichten usw.) des neuronalen Netzes [Abbildung 8.5](#) würden etwa 200 Parameter verwendet. Diese Parameter würde man normalerweise alle durch die Minimierung einer Kostenfunktion bestimmen. Sobald noch mehr Knoten, mehr verborgene Schichten und mehr Output ins Spiel kommen, ist man schnell bei neuronalen Netzen mit Millionen oder Milliarden von Parametern. Zum Vergleich: Das menschliche Gehirn hat 60 Billionen Parameter – das sind 60.000.000.000.000 Verbindungen.

## 8.3 Aus dem Leben gegriffen: Ein Beispiel

Für das folgende Beispiel setze ich ein neuronales Netzwerk ein, um handschriftlich notierte Zahlen von 0 bis 9 zu erkennen. Dieses Beispiel findest du häufig in Lehrbüchern, weil diese Daten zugänglich und relativ einfach zu implementieren sind – wenn du schon ein Programmierer bzw. eine Programmiererin bist. Den Beispielcode findest du überall im Internet. Leider ist die Durchführung des Ganzen immer noch viel zu fortgeschritten für dieses Buch.

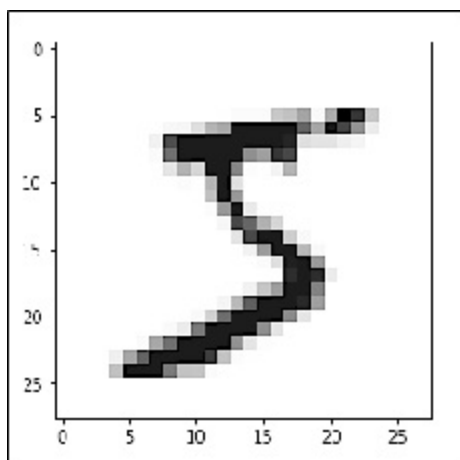
Wie gebe ich also die Handschrift in den Computer ein? Erst einmal müssen die Bilder digitalisiert werden. Dazu kann man zum Beispiel ein Schwarz-Weiß-Bild in ein  $28 \times 28$ -Pixel-Quadrat umwandeln. Jedes Pixel erhält eine Zahl von 0 bis 255 – 0 steht dabei für reines Weiß, 255 für das schwärzeste Schwarz. Jede handgeschriebene Ziffer wird dann durch einen Vektor der Dimension  $28 \times 28 = 784$  dargestellt. Das ist unser Input-Vektor, viel größer als alles, womit wir bisher gearbeitet haben.

Um unseren Algorithmus zu trainieren und zu optimieren, um also die Zehntausende von Parametern für unser neuronales Netz zu finden, benötigen wir jetzt Daten. Glücklicherweise gibt es einen bekannten Datensatz, die *Modified National Institute of Standards and Technology*-Datenbank (MNIST) mit 60.000 Trainingsbildern und 10.000 Testbildern. Das alles sind Handschriftproben, die von

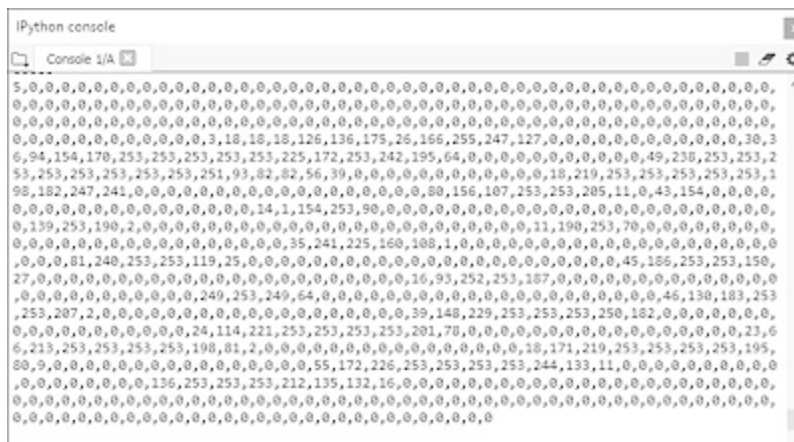


Mitarbeitenden des amerikanischen Volkszählungsbüros und amerikanischen Highschool-Schüler\*innen stammen. In [Abschnitt A.2](#) am Ende des Buches zeige ich dir, wo genau diese Daten zu finden sind.

In [Abbildung 8.6](#) und [Abbildung 8.7](#) siehst du ein Beispiel für ein solches digitalisiertes Bild und seine Rohdaten.



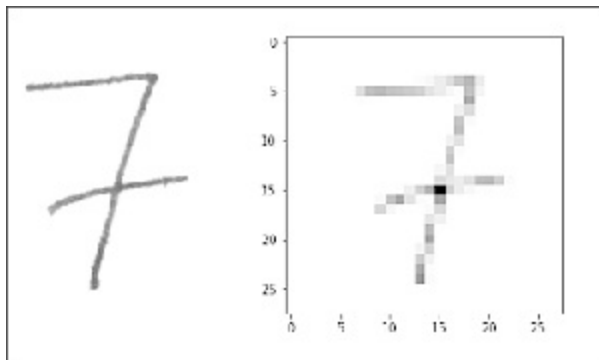
**Abbildung 8.6** Die digitalisierte Abbildung einer handschriftlichen Ziffer



**Abbildung 8.7** Die Rohdaten der digitalisierten Ziffer aus [Abbildung 8.6](#)

Das Netzwerk, das ich verwendet habe, hatte also 784 Eingaben, eine verborgene Schicht mit 50 Knoten und 10 Outputs. Warum 10 Outputs und nicht nur einen? Wir versuchen nun einmal, eine Zahl vorherzusagen. Und da wäre es nicht hilfreich, herauszufinden, dass beispielsweise eine 7 auf halbem Weg zwischen einer 6 und einer 8 liegt. Das ist ein *Klassifizierungsproblem*.

Nachdem ich den Algorithmus trainiert habe, habe ich ihm schließlich einige meiner handschriftlichen Ziffern gezeigt. Ich habe ihm meine 3 gezeigt, die du als [Abbildung 4.5](#) in diesem Buch. Das neuronale Netz hat sie richtig erkannt: Es wusste, dass es sich um eine 3 handelte.



**Abbildung 8.8** *Eine von mir handschriftlich verfasste 7 mit ihrer digitalisierten Entsprechung*

Anschließend habe ich ihm meine 7 gezeigt. Diese hat es allerdings falsch erkannt – vielleicht, weil ich sie im europäischen Stil mit einem Querbalken in der Mitte schreibe.

Da der Output des neuronalen Netzes ein zehndimensionaler Vektor ist, können wir uns auch genauer anschauen, was der Algorithmus genau von meiner 7 hielt. Er gab an, dass es sich mit 84,2 %

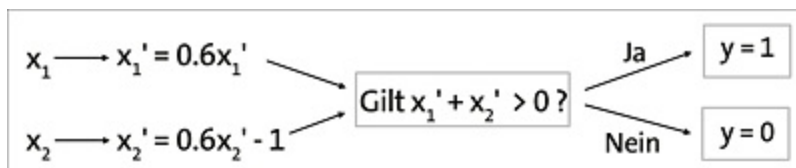
Wahrscheinlichkeit um eine 2 handele – mit 6,8 % Wahrscheinlichkeit habe ich eine 7 geschrieben und mit 5,7 % eine 4. Diese Abweichung lag aber wirklich an meiner Handschrift. Bei der Verwendung von Testdaten hatte der Algorithmus eine Erfolgsquote von 95 %.

## 8.4 Dein Projekt

Ein eigenes neuronales Netz zu erstellen, ist gar nicht so einfach. Deshalb gebe ich dir als eigenes Projekt drei schon fertige neuronale Netze und du überlegst dir, was genau sie darstellen könnten. Keine Sorge: Ich gebe dir natürlich ein paar Hinweise.

Alle neuronalen Netze haben zweidimensionale Input-Vektoren und einen einzigen Output.

Der erste Algorithmus ist super einfach aufgebaut. Er hat nicht einmal verborgene Schichten. So sieht er aus:



**Abbildung 8.9** Der erste Beispielalgorithmus ist von einer bekannten Logikfunktion abgeleitet.

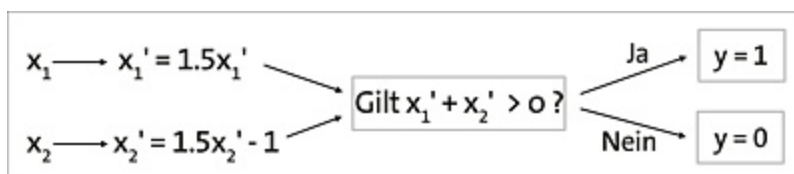
Für den Anfang kannst du einmal nur Nullen und Einsen für  $x_1$  und  $x_2$  eingeben, dann hast du vier verschiedene Kombinationen. Die Ergebnisse kannst du in einer Tabelle wie unten festhalten.

**Ein Hinweis:** Es handelt sich bei dem Netz um eine sehr spezielle Funktion in der Welt der Logik.

--	--	--

$x_1$	$x_2$	$y$
0	0	
1	0	
0	1	
1	1	

Der zweite Algorithmus ist dem ersten Beispiel ziemlich ähnlich:



**Abbildung 8.10** Ein zweiter Beispiyalgorithmus, um die Funktionsweise neuronaler Netze besser kennenzulernen

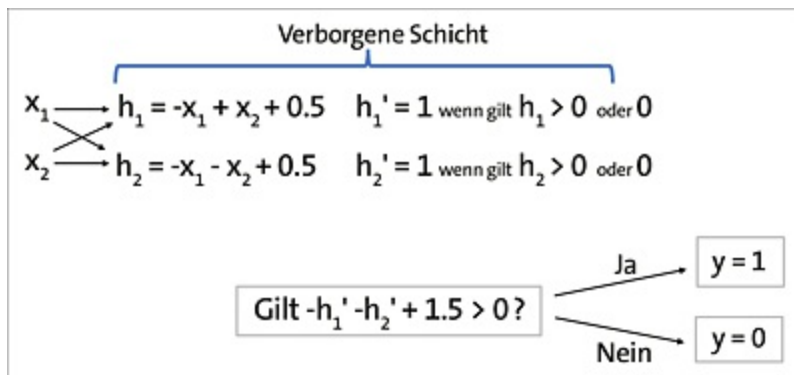
Gib wieder wie oben Nullen und Einsen ein und notiere dann die Ergebnisse in einer ebensolchen Tabelle.

**Ein Hinweis:** Hier handelt es sich um eine weitere spezielle Logikfunktion.

$x_1$	$x_2$	$y$
0	0	
1	0	

0	1	
1	1	

Der dritte Algorithmus ist etwas kniffliger. Er ist zwar eine weitere Logikfunktion, kann allerdings nicht durch ein neuronales Netz ohne verborgene Schicht dargestellt werden. Das ist total normal: In der Praxis haben fast alle neuronalen Netze verborgene Schichten, manchmal auch sehr viele davon.



**Abbildung 8.11** Ein drittes, etwas komplexeres Beispiel für ein neuronales Netz

Für dieses Beispiel musst du allerdings eine größere Tabelle ausfüllen:

$x_1$	$x_2$	$h_1$	$h_2$	$h_1'$	$h_2'$	$h_1' + h_2'$	$y$
0	0						
1	0						

0	1						
1	1						

Was glaubst du, welche Logikfunktion hier dargestellt wird?

Was ist denn nun anders am dritten Beispiel, warum ist es unmöglich, die Funktion ohne eine verborgene Schicht darzustellen? Um das zu verstehen, könnte es helfen, ein Diagramm mit einer  $x_1$ - und einer  $x_2$ -Achse zu zeichnen. Zeichne dann einen Punkt für jede der vier Kombinationen: Rot für  $y = 0$  und Blau für  $y = 1$ .

## Kapitel 9

# Mehr zu: Reinforcement Learning

Ich habe die wahrscheinlich spannendste Machine-Learning-Methode für den Schluss aufgehoben.

*Reinforcement Learning* ist eine der wichtigsten Arten des Machine Learnings. Mit einem System aus Belohnungen und Bestrafungen lernt ein Algorithmus, wie er handeln oder sich verhalten soll. Dabei kann es beispielsweise um ein Spiel gehen – oder auch darum, wie sich ein Roboter in einer bestimmten Umgebung bewegt. Dass dabei das Spiel oder die Umgebung nicht im Code festgelegt wird, ist der Faktor, der das Reinforcement Learning ausmacht. Der Algorithmus muss die Konsequenzen seines Handelns aus diesen Handlungen selbst lernen – er lernt also durch Versuch und Irrtum (*Trial and Error*).

Während es beim überwachten Lernen darum geht, zu entscheiden, was etwas ist bzw. als was es klassifiziert werden soll, und das unüberwachte Lernen sich damit befasst, wie Beziehungen zwischen Daten gefunden werden können, geht es beim Reinforcement Learning darum, der Maschine beizubringen, etwas Bestimmtes zu



tun. Dieser Vorgang ist von der Verhaltenspsychologie inspiriert. Wir wollen erreichen, dass ein Algorithmus lernt, welche Aktionen er durchführen kann, um die größtmögliche Belohnung zu erhalten (oder eben die kleinstmögliche Bestrafung). Da es darum geht, der Maschine beizubringen, ein bestimmtes Ziel zu erreichen, ist es sehr naheliegend, dafür den Vorgang des Spielens als Vorbild zu verwenden. Spiele haben klare Regeln und ein Ziel. Unsere Beispiele in diesem Abschnitt werden auch mehr oder weniger mit Spielen zusammenhängen. Der Trick beim Reinforcement Learning besteht darin, dass der Maschine nicht explizit mitgeteilt wird, was das Ziel des Spiels ist – oder welche Regeln gelten. Sie lernt das Ganze nur durch Trial and Error, während sie mit ihrer Umgebung interagiert.

Du kannst etwas ganz Ähnliches beobachten, wenn du einem zweijährigen Kind die Fernbedienung für den Fernseher gibst. Es hat wahrscheinlich schon einmal gesehen, wie du die Fernbedienung benutzt hast, um den Fernseher einzuschalten und die Kanäle zu wechseln, daher kann es eine vage Verbindung herstellen. Aber wenn es zum ersten Mal tatsächlich die Fernbedienung bekommt, drückt es vermutlich zufällig auf Tasten, bis etwas passiert. Es wird eine Phase geben, in der sich das Kind auf die wichtigsten Tasten konzentriert – und nach einer Weile lernt es, welche Taste den Fernseher einschaltet, wie man die Kanäle wechselt oder eine DVD startet. Und danach verwendet es absolut zuverlässig eine Teilmenge von Tasten.

Komischerweise scheint es nie zu lernen, wie man den Fernseher wieder ausschaltet – zumindest nicht, bis es Mitte vierzig ist.

Eine andere Art, Trial and Error in die Programmierung eines Algorithmus einzubeziehen, besteht im Ausnutzen und Erkunden (*Exploiting and Exploring*): Einerseits möchten wir ja, dass die Maschine alles, was sie gelernt hat, voll ausnutzt, um das Spiel zu gewinnen. Auf der anderen Seite wird sie aber auf diese Weise nichts lernen – es sei denn, sie hat zuvor viel Erkundungsarbeit betrieben. Ohne diese Erkundung kann die Maschine in einer festen Routine stecken bleiben und Aktionen ausführen, die nicht optimal sind. Ein Ausgleich zwischen Exploiting und Exploring wird für die Lernalgorithmen in diesem Kapitel wichtig sein.

## 9.1 Wofür Reinforcement Learning eingesetzt wird

- **Spiele spielen:** Reinforcement Learning ermöglicht das Spielen komplexer Spiele wie Go, Schach oder auch von Spielen auf dem Atari. Dazu wird die Spielumgebung Stück für Stück erkundet und die Belohnungen durch sie werden maximiert.
- **Robotik:** Mit Reinforcement Learning können Roboter für die industrielle Automatisierung gesteuert werden – zum Beispiel für das Zusammenbauen von Teilen, das Sortieren von Objekten oder das Navigieren durch Hindernisse.

- **Geschäftsstrategie-Entwicklung:** Reinforcement Learning kann verwendet werden, um geschäftliche Entscheidungen zu optimieren – zum Beispiel die Preisgestaltung von Produkten, das Bestandsmanagement, Marketingkampagnen oder auch den Kundenservice.
- **Bildung:** Algorithmen können lernen, personalisierte Lernunterlagen zu erstellen. Sie können individuelle Aufgabenstellungen und Materialien entsprechend den Bedürfnissen und Vorlieben Lernender bereitstellen.
- **Kühlung von Datenzentren:** Mit Reinforcement Learning kann das Kühlsystem von Datenzentren gesteuert werden, indem der zukünftige Energieverbrauch vorhergesagt und der Stromverbrauch minimiert wird, während automatisiert Sicherheitsstandards eingehalten werden.
- **Selbstfahrende Autos:** Algorithmen können sogar das autonome Fahren lernen, indem sie Straßenverkehrsregeln, Verkehrssignale, Passant\*innen, Menschen auf Fahrrädern sowie andere Fahrzeuge wahrnehmen und entsprechende Maßnahmen ergreifen, die Sicherheit und einen reibungslosen Verkehrsablauf gewährleisten.
- **Finanzen:** Mithilfe von Reinforcement Learning können Algorithmen effizient mit Aktien, Währungen oder Kryptowährungen handeln. Sie analysieren Markttrends, Risiken und Chancen und maximieren Gewinne.

## 9.2 Im Detail

Was Reinforcement Learning ist, lässt sich sehr gut über die fachsprachlichen Ausdrücke erklären, die für es verwendet werden. Ich habe diese Fachbegriffe nicht in den Abschnitt zum Fachjargon aufgenommen, weil sie nicht für das Machine Learning allgemein, sondern speziell für das Reinforcement Learning verwendet werden. Während ich im Folgenden die grundlegenden Begriffe erläutere, werde ich (wie ich oben schon angedeutet habe) auf einige bekannte Spiele verweisen.

- **Aktion:** Welche Entscheidungen sollte man beim jeweils vorliegenden mathematischen Modell treffen? Welchen einarmigen Banditen wählt man am besten in einem Casino? (Diese Frage klären wir demnächst auch noch im Detail.) Wo sollte das nächste Kreuz im Tic-Tac-Toe-Raster gesetzt werden? (Das hatten wir im Abschnitt zu Michie und den Streichholzschachteln.) Welche meiner Handkarten sollte ich bei einer Partie *Bungo* als nächste spielen? All diese Entscheidungen und ihre Durchführung sind *Aktionen*.
- **Belohnung/Strafe:** Du führst eine Aktion durch und wirst (vielleicht) belohnt. Du drückst in *Doom* einen Knopf an irgendeiner Wand und erhältst dafür die BFG. Du schlägst beim *Schach* eine Figur deines Gegners oder deiner Gegnerin – und bei diesem Schach-Set ist die Figur aus (weißer) Schokolade und du darfst sie essen, wenn du sie

geschlagen hast. Das sind alles Beispiele für sofortige *Belohnungen*. Aber es wäre auch denkbar, dass es bis zum Ende des Spiels erst einmal keine Belohnung gibt: Zum Beispiel erhält der Schachspieler bzw. die Schachspielerin die 1.000-Dollar-Gewinnprämie erst am Ende des Turniers. Und natürlich gibt es nicht nur Belohnungen! Du könntest beispielsweise ein Rätsel lösen müssen, um einen Preis zu gewinnen. Jede Sekunde, die du dafür brauchst, ist deine *Strafe* im Spiel.

- **Zustand:** Der *Zustand* ist eine Beschreibung eines aktuellen Spielstandes. Stell dir beispielsweise einen Schnappschuss vom Spielbrett vor: Du siehst darauf die aktuellen Positionen von Kreisen und Kreuzen bei deiner *Tic-Tac-Toe*-Partie oder die Positionen der Steine bei einem *Kōnane*-Spiel. Der Spielzustand an sich ist dann aber auch wieder ein spannendes Konzept. Die Frage ist: Welche Informationen brauche ich, um einen Zustand darzustellen? Bei Tic Tac Toe brauchte Donald Michie 304 Streichholzschachteln.

Beim *Blackjack*-Spielen musst du die Anzahl der Karten in deiner Hand kennen, die Anzahl der Asse und du musst wissen, was die offen gespielte Karte des Kartengebers (Croupiers) ist. Damit du in einem Casino eine Gewinnchance hast, solltest du auch wissen, welche Karten aus dem Deck schon verwendet bzw. ausgeteilt wurden. Aber manchmal ist die Menge an Informationen, die zur Beschreibung eines Zustandes nötig sind, unabzählbar groß. Bei Go gibt es zum Beispiel typischerweise 361 Punkte, die jeweils leer oder von einem weißen oder schwarzen Stein besetzt sein könnten. Man

könnte nun vermuten, dass es  $3^{361}$  mögliche Zustände gibt. Aber weil nicht alle Zustände spielregelkonform sind, ist die korrekte Anzahl lediglich

208.168.199.381.979.984.699.478.633.344.862.770.286.522.453.884.!

## 9.3 Aus dem Leben gegriffen: Ein Beispiel

Für den Fall, dass du bisher noch nie ein Casino besucht hast – oder die Spielhalle am Ende des Brighton Piers oder (weniger glamourös) auch eine Raststätte auf der Autobahn M4: Ich erkläre dir im Folgenden den Schrecken des einarmigen Banditen bzw. des Spielautomaten.

Wie funktionieren diese Maschinen? Nun, als Erstes steckst du mal dein hart verdientes Gehalt (oder dein Taschengeld) in den Schlitz am Automaten. Dann drückst du einen großen Knopf oder ziehst an einem Hebel (daher kommt das »einarmig« in »einarmiger Bandit«; warum er »Bandit« heißt, werden wir auch gleich noch thematisieren). Vor deinen Augen drehen sich bunte Räder und drei oder mehr Symbole erscheinen in einer Reihe nebeneinander – klassischerweise werden Bilder von Früchten verwendet. Wenn die Bilder sich zu speziellen Kombinationen zusammensetzen (ich meine mich zu erinnern, dass Kirschen besonders hohe Gewinne erzielten), kommt Geld aus einem Schacht an der Maschine. Du hast gewonnen! Indem sie hohe Jackpots verkünden, locken Casinos immer wieder Menschen in dieses Narrenspiel.



**Abbildung 9.1** *Blick in ein Casino mit diversen Spielautomaten*

Leider stehen die Chancen immer gegen die Spielenden. Du kannst dich glücklich schätzen, wenn du für jeden Dollar, den du in den Schlitz steckst, 95 Cent, manchmal auch nur 70, zurückbekommst. Du wirst dein Geld wahnsinnig schnell verlieren. Deswegen habe ich oben »Schrecken« geschrieben und deswegen heißt die Maschine auch »Bandit«. Wirf einfach mal einen Blick auf die Menschen, die an diesen Automaten spielen – willst du werden wie sie? [Anmerkung der Redaktion: Genug mit der Moralpredigt jetzt, Wilmott.]

Okay, schauen wir uns doch einfach mal an, warum das Ganze ein gutes Beispiel für Reinforcement Learning ist.

Stell dir vor, du sitzt vor einer Reihe von 10 Spielautomaten in einem Casino in Las Vegas. Es ist 7 Uhr morgens an einem Mittwoch im Februar. (Es sollte also nicht viel los sein.) Du darfst jetzt mal kostenlos mit den Banditen spielen! Such dir einen Automaten aus, zieh am Arm und warte auf das Ergebnis. Anschließend kannst du



einen anderen Banditen ausprobieren oder beim gleichen bleiben.  
Dein Ziel ist es, so viel wie möglich zu gewinnen.

Es gibt allerdings ein Geheimnis, das ich dir beim Spielen nicht verraten werde. Also schau mal kurz weg und lies den nächsten Absatz hier nicht, okay?

Einige dieser Automaten haben bessere Gewinnchancen als andere!  
In [Tabelle 9.1](#) findest du eine Liste aller zehn Banditen mit ihren Gewinnchancen. Offensichtlich sind Nummer 1 und 5 ziemlich mies, wohingegen Nummer 4 der beste Automat für dich wäre.

<b>Bandit 1</b>	<b>10 %</b>
<b>Bandit 2</b>	<b>50 %</b>
<b>Bandit 3</b>	60 %
<b>Bandit 4</b>	80 %
<b>Bandit 5</b>	10 %
<b>Bandit 6</b>	25 %
<b>Bandit 7</b>	60 %
<b>Bandit 8</b>	45 %
<b>Bandit 9</b>	75 %

**Tabelle 9.1** Gewinnchancen unterschiedlicher einarmiger Banditen in einem fiktiven Casino

Leider kennst du die Wahrscheinlichkeiten natürlich nicht – du hast absolut keine Ahnung, welcher der für dich beste Bandit ist. Du weißt nicht einmal, dass sie unterschiedliche Gewinnchancen haben. Das ist die Idee hinter dem Reinforcement Learning: Du kennst die Regeln des Spiels nicht und du weißt nicht, welche Aktion zum besten Ergebnis führt. Du muss einfach das Spiel spielen und aus den Ergebnissen lernen.

Okay, jetzt kannst du wieder weiterlesen.

Lass uns spielen!

Du wählst Bandit 2, ziehst den Hebel und ... verlierst. Sorry!

Versuch es noch mal.

Weil du mit Bandit 2 kein Glück hattest, versuchst du es jetzt mit Bandit 3. Du gewinnst! Glückwunsch.

Hast du das Gefühl, dass du jetzt nur noch gewinnen kannst? Okay, du versuchst es mit Bandit 3 noch einmal ... und verlierst.

Was nun?

Um ehrlich zu sein, hast du nicht viele Informationen, auf die du dich hier stützen könntest. Drei Versuche sind statistisch gesehen nicht gerade aussagekräftig.

Okay, du versuchst es jetzt einfach mal mit Bandit 10. Du gewinnst.

Du machst also mit Bandit 10 weiter. Du gewinnst wieder!

Du hast das Gefühl, dass du auf einem guten Weg bist, also versuchst du es noch einmal mit Bandit 10. Du verlierst. Verdammt!

Du versuchst es weiterhin mit Bandit 10 und erzielst gute Ergebnisse. Aber du fragst dich ständig, ob andere Automaten vielleicht noch besser sind. Also experimentierst du herum und spielst mit verschiedenen Banditen.

Dadurch machst du *Erfahrungen* hinsichtlich der Prozentsätze.

Du *lernst* etwas über die Gewinnchancen.

Und du wirst *belohnt*, in bar.

Dieses Beispiel zeigt dir die Grundlagen des Reinforcement Learnings und wie es für das Machine Learning programmiert werden könnte. Es ist auch ein besonders einfaches Beispiel, weil es keine *Zustände* gibt, die man analysieren und getrennt voneinander betrachten müsste. Aktionen? Ja: das Ziehen am jeweiligen Banditen. Aber Zustände gibt es nicht.

Ein wichtiger Teil des Algorithmus ist es, jedem Banditen einen Wert zuzuweisen bzw. der Aktion »den Banditen verwenden« einen Wert zuzuweisen. Für dieses Beispiel müssen wir dafür das Verhältnis von »den Banditen verwenden« zu »gewinnen« beobachten. Das ist dann der Wert des jeweiligen Banditen.

Zuerst wählst du also einen Banditen aus und spielst. Du gewinnst oder verlierst. Jetzt kannst du den Wert des Banditen berechnen. Nach jedem Zug musst du einen neuen Banditen zum Spielen aussuchen. Dazu wählst du die *Aktion* (den Automaten), die zu diesem Zeitpunkt den höchsten errechneten *Wert* hat.

Von Zeit zu Zeit, sagen wir mal bei 10 % der Aktionen, wählst du aber auch mal einen Banditen bzw. eine Aktion rein zufällig aus, wobei alle Banditen gleich wahrscheinlich sind.

In [Tabelle 9.2](#) findest du eine Abfolge von Aktionen, die in dem Casino stattfinden könnten. Dabei ist natürlich aber nur ein Auszug abgebildet. Denk daran, dass es in unserem Beispiel nur zwei Begründungen gibt, sich für einen Automaten zu entscheiden:

1. Du wählst einen Banditen, der zu diesem Zeitpunkt den höchsten Wert hat – wobei der Wert das Verhältnis von Gewinnen zur Anzahl der Aktionen am Automaten ist.
2. Du wählst rein zufällig einen Banditen aus. Sagen wir mal, dieser Fall tritt bei 10 % deiner Aktionen ein und kann auch bedeuten, dass du einen Spielautomat erwischst, der keine besonders guten Gewinnchancen hat.

Du wirst sehr viele Versuche brauchen, aber schließlich wird dieser fiktive Spielautomaten-Algorithmus Bandit Nummer 4 begegnen, der im Endeffekt den höchsten Wert erzielen und fast immer ausgewählt werden wird.

Ich schreibe »fast immer«, weil da ja immer noch diese rein zufällige Auswahl geschieht. Der Grund, warum diese überhaupt stattfindet, ist, dass wir nicht in einer guten, aber eben nicht ganz optimalen Routine stecken bleiben wollen.

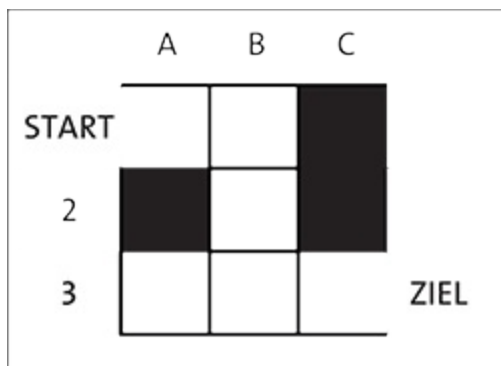
<b>Anmerkungen</b>	<b>Bandit</b>	<b>Gewonnen (g) / verloren (v)</b>	<b>Anzahl Aktionen an diesem Banditen</b>	<b>Anzahl Gewinne an diesem Banditen</b>	<b>Wert c Bandit</b>
Zufallsauswahl Nr. 2	2	v	1	0	0
Zufallsauswahl Nr. 3	3	g	1	1	1
Bei 3 bleiben, höchster Wert	3	v	2	1	0,5
Bei 3 bleiben, höchster Wert	3	v	3	1	0,333
Zufallsauswahl Nr. 10	10	g	1	1	1
Bei 10 bleiben, höchster Wert	10	g	2	2	1

Bei 10 bleiben, höchster Wert	10	v	3	2	0,667
Bei 10 bleiben, höchster Wert	10	v	4	2	0,5
Bei 10 bleiben, höchster Wert	10	v	5	2	0,4
Zufallsauswahl Nr. 5	5	v	1	0	0
Zurück zu 10, höchster Wert	10	g	6	3	0,5
Bei 10 bleiben, höchster Wert	10	g	7	4	0,571
Bei 10 bleiben, höchster Wert	10	g	8	5	0,625
...					
Zufallsauswahl Nr. 4	4	g	1	1	1
Bei 4 bleiben, höchster Wert	4	g	2	2	1

**Tabelle 9.2** Das beispielhafte Auswählen einarmiger Banditen im Verlauf. Positive Aktionen bringen Belohnung, das Verlieren senkt den Wert der Automaten.

## 9.4 Dein Projekt

Finde den Ausgang aus dem Labyrinth! Zugegeben, es ist nicht das herausforderndste Labyrinth aller Zeiten – schau es dir auf [Abbildung 9.2](#) einmal genauer an.



**Abbildung 9.2** Ein Labyrinth, das aus 3×3 Feldern besteht und ein Start- und ein Zielfeld aufweist

Du startest in Zelle A1 und bewegst dich dann von Zelle zu Zelle. Wenn du C3 erreichst, hast du es geschafft. Wie viele Schritte wirst du dafür brauchen?

(Es ist wirklich nicht so einfach, wie es aussieht. Stell dir vor, du befindest dich in einem Kellerraum ohne Licht. Es gibt Türen, durch die du gehen kannst, aber du hast keine Ahnung, wie viele Räume es insgesamt gibt. Das ist Reinforcement Learning: Die Regeln des Spiels sind nicht explizit programmiert, du musst einfach das Spiel spielen und sehen, wie du zurechtkommst!)

Du fängst also an, dich zufällig durch das Labyrinth zu bewegen. Nimm einen sechsseitigen Würfel zur Hand und würfle, um zu bestimmen, welchen Weg du als Erstes einschlägst.

1. **A1:** Es gibt nur eine Möglichkeit, sich von diesem Feld aus zu bewegen, daher ist kein Wurf erforderlich.
2. **B1:** Es gibt nun zwei Möglichkeiten: zurück zu A1 (wenn eine 1, 2 oder 3 gewürfelt wird) oder runter zu B2 (wenn eine 4, 5 oder 6 gewürfelt wird).
3. **B2:** Würfele eine 1, 2 oder 3, um wieder zu B1 zu gehen, eine 4, 5 oder 6, um weiter zu B3 zu gehen.
4. **B3:** Von hier aus kannst du drei verschiedene Zellen ansteuern. Würfele eine 1 oder 2 und gehe zu B2; mit einer 3 oder 4 gehst du zu A3; mit einer 5 oder 6 gehst du zu C3 – in diesem Fall hast du es aus dem Labyrinth geschafft!
5. **A3:** Es ist kein Wurf erforderlich, denn du kannst von hier aus nur zur Zelle B3 gehen.

Du wirst ziemlich schnell feststellen, dass du eine Menge Züge brauchst, um von A1 zu C3 zu gelangen. Wenn meine Berechnungen stimmen, sind es wegen viel Hin- und Herzieherei etwa 18 Züge. Selbst um von B3, das nur ein Feld vom Ausgang entfernt ist, rauszukommen, werden durchschnittlich neun zufällige Würfe benötigt. Das ist wirklich weit entfernt von »optimal«.

Um das Szenario zu optimieren, müssen wir eine weitere Wertfunktion



aufstellen. Aber im Gegensatz zum Beispiel mit dem einarmigen Banditen müssen wir hier nicht nur die Aktion bewerten, sondern auch einen Zustand. Mit dem *Zustand* ist in diesem Fall die Zelle gemeint, in der wir uns befinden, und mit *Aktion* die Richtung, in die wir uns von diesem Zustand aus bewegen. In [Abbildung 9.3](#) habe ich skizziert, wie du den Verlauf der Bewegungen dokumentieren könntest.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
A1	x		x																	
B1		x		x																
B2					x															
B3						x		x		x										
A3							x		x											
C3											x									

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
A1	x				x															
B1		x		x		x		x												
B2			x				x		x		x									
B3										x		x		x						
A3													x							
C3															x					

**Abbildung 9.3** Zwei Versuchsläufe, um aus unserem Labyrinth zu entkommen

Hier siehst du zwei Versuche, aus dem Labyrinth herauszukommen. Um zu sehen, wie es funktioniert, musst du jetzt Dutzende dieser Versuche aufschreiben (bei komplexeren Labyrinthen müssten es noch viel mehr sein). Die Tabelle auf der linken Seite zeigt den Verlauf des Weges, der auf der Grundlage von Würfelwürfen genommen wurde. Die obere Route war:

A1 → B1 → A1 → B1 → B2 → B3 → A3 → B3 → A3 → B3 → C3

Das sind 10 Züge.

Der zweite Versuch dauerte insgesamt 14 Züge.

Die Tabellen auf der rechten Seite zeigen, wie lange es dauerte, aus dem Labyrinth zu kommen bzw. von welcher Zelle aus du dich zu welcher Zelle bewegt hast. Das ist die *Wertfunktion*, die ich vorhin schon erwähnt habe. Die schwarzen Zellen sind Aktionen, die nicht möglich sind. Zum Beispiel kannst du nicht von B2 nach A1 gehen. Für die grauen Zellen gibt es keine Auswahl. Von B1 und B2 aus gibt es zwei Optionen, von B3 aus gibt es drei.

Jetzt fügst du die Zahlen in die Tabelle ein. Siehst du in der oberen linken Tabelle die beiden Kreuzchen in der A1-Zeile? Diese wurden eingetragen, weil der Weg durch das Labyrinth zeitweise zurück in Richtung Eingang verlief – A1 wurde dabei zweimal angesteuert. Einer der Wege von A1 aus hat 10 Züge bis zum Ausgang gedauert, der zweite Weg von A1 aus (nachdem wir zwischenzeitlich wieder dort angekommen waren) dauerte 8. Welcher war schneller? Ganz offensichtlich der mit nur 8 Zügen. Also kommt in die graue Zelle A1 (Zeile) / B1(Spalte) eine 8. Ähnlich war es in Zelle B3, von der aus wir in 5 (über A3) oder 3 Zügen den Ausgang erreicht haben. Ergo kommt in die Zelle B3(Zeile) / A3(Spalte) eine 3.

In diesem Würfelszenario gab es keine Bewegung von B2 zu B1, also lassen wir dieses Feld weiß. (Wenn du diesen Algorithmus wirklich mal programmieren solltest, solltest du in diese Felder allerdings besser eine sehr große Zahl schreiben, damit das Programm nicht auf die Idee kommt, von diesem Feld aus bräuchte es null Züge bis zum Ausgang.)

Nachdem du diese Routen ein paarmal durchgespielt hast, siehst du ziemlich schnell, welche Züge am besten geeignet sind, um den Ausgang aus dem Labyrinth in kürzester Zeit zu schaffen.

Für den letzten Teil des Projekts brauchst du eine weitere Person, die dir hilft. Eine\*r von euch entwirft ein Labyrinth, die andere Person versucht, daraus zu entkommen. Diese Person bekommt dabei aber den Aufbau des Labyrinths nicht zu sehen. Alles, was sie bekommt, sind Anweisungen wie:

- *Vom Apfel aus kannst du zur Banane oder zur Mandarine gehen.*
- *Von der Banane aus kannst du zum Apfel, zur Mandarine, zur Dattel oder zur Holunderbeere gehen.*
- *Von der Mandarine aus kannst du zu ... und so weiter.*

Ich habe die ganzen Buchstaben durch Obstsorten ersetzt, damit es nicht so leicht ist, sich das Raster des Labyrinths vor Augen zu rufen und zu merken. Fangt erst einmal mit einem 5×5-Labyrinth an, das ist für den Beginn Herausforderung genug. Der Mensch, der sich durch das Labyrinth bewegt, kann sich zufällig bewegen, einen Würfel werfen oder auch einfach raten. Je größer das Labyrinth ist, desto mehr Versuche müsst ihr durchführen, bevor die Person im Labyrinth anfangen kann, die Route zu optimieren.

Bei noch komplexeren Labyrinth und anderen Fragestellungen hinsichtlich Reinforcement Learning musst du wahrscheinlich etwas differenzierter vorgehen, um herauszufinden, welche Aktion optimal

ist. Es müsste aber immer noch eine *Wertfunktion* erstellt werden, die die Belohnungen für bestimmte Aktionen in den einzelnen Zuständen determiniert, und wie gehabt müssten wieder rein zufällige Aktionen eingefügt werden, damit der Optimalfall nicht schlicht übersehen wird.

Eigentlich ist das Vorgehen nicht anders, als wenn man den kürzesten Weg nach Hause suchen würde: Manchmal denkt man, man hat die beste Route, aber von Zeit zu Zeit geht man dann doch einen anderen Weg. Beim Partner\*innen-Projekt könnte zum Beispiel die Person, die das Labyrinth erstellt hat, eine Route mit besonders ungewöhnlichen Bedingungen präsentieren, die man gar nicht finden *kann*, wenn man den üblichen Pfaden folgt.



**Abbildung 9.4** Ein besonders ungewöhnliches Labyrinth mit komplexer Wegführung

In [Abbildung 9.4](#) siehst du ein Labyrinth mit einem sehr schnellen Weg zum Ausgang. Allerdings führt dieser über ein Wurmloch, das sich nur alle sechs Züge öffnet.

# Anhang

## Nachwort

Es hat mir großen Spaß gemacht, dieses Buch zu schreiben – und ich hoffe, du hattest ebenso viel Spaß dabei, es zu lesen. Vielleicht fandest du es ja sogar lehrreich. In der englischen Ausgabe des Buches habe ich sogar geschummelt und für die Abschnitte mit dem Titel »Wofür ... eingesetzt wird« die Hilfe generativer KI in Anspruch genommen. Abgesehen davon sind aber natürlich alle Worte in diesem Buch meine eigenen und ich übernehme die volle Verantwortung für sie.

Bitte lass mich gern wissen, wie ich dieses Buch noch verbessern kann – und wie du mit den Projekten zurechtgekommen bist. (Darüber hinaus bin ich auch gerne bereit, die hochoffiziellen wilmottschen Spielregeln für das Spiel *Bungo* zu teilen.) Ich bin erreichbar über [paul@wilmott.com](mailto:paul@wilmott.com) – und überraschend gut zugänglich.

(Ich habe weder Twitter noch Facebook, Instagram oder sonstige Social-Media-Konten – man kann nie wissen, welche Daten sie dort von einem sammeln.)

# Datensammlung

Für ein sinnvolles Training vieler Algorithmen brauchst du große Datenmengen. Natürlich kannst du immer selbst Umfragen zu den Größen, Gewichten, Haustieren usw. der Menschen in deinem Umfeld durchführen, aber sobald es dann um Immobilienwerte, Verkehrsstatistiken und Ähnliches geht, wirst du sehr schnell an deine Grenzen stoßen. Du brauchst Daten aus dem Internet. Gute Adressen, um nach solchen Daten zu suchen, sind zum Beispiel <https://toolbox.google.com/datasetsearch> und <https://www.kaggle.com>. Letztere schreiben sogar Wettbewerbe aus – allerdings musst du dich dort registrieren.

Im Folgenden findest du einige Datensätze, die du für den Start oder einfach zum Ausprobieren herunterladen könntest.

- **Titanic:** Ein sehr beliebtes Datenset für Machine Learning ist das der Passagiere auf der Titanic, insbesondere Informationen über die Überlebensraten basierend auf Geschlecht, Ticketklasse usw. Du findest die Daten über <https://www.kaggle.com/c/titanic>.
- **Iris:** Es gibt ein sehr bekanntes Datenset für Iris-Sorten. Gemeint ist die Blume, nicht die Regenbogenhaut im Auge. Dieses Datenset ist an vielen Stellen zu finden, zum Beispiel unter <https://www.kaggle.com/uciml/iris>.

- **Größen und Gewichte:** Zehntausende Größen und Gewichte von Erwachsenen findest du unter <https://www.kaggle.com/mustafaali96/weight-height>.
- **Handschrift:** Siebzigtausend handgeschriebene Bilder von Ziffern findest du unter <http://yann.lecun.com/exdb/mnist/>. Es handelt sich um die Handschriften von Mitarbeitenden des amerikanischen Volkszählungsbüros sowie amerikanischen Highschool-Schüler\*innen im CSV-Format.
- **Abstimmungen im House of Commons:** Die Website <https://www.publicwhip.org.uk> sammelt Daten zur Aktivität in den britischen Parlamentskammern *House of Commons* und *House of Lords*. Um an die Rohdaten zu gelangen, kannst du dich bis zu <https://www.publicwhip.org.uk/project/data.php> und <https://www.publicwhip.org.uk/data/> durchklicken.
- **Pilze:** Du möchtest vorhersagen, ob ein Pilz essbar oder giftig ist? Dann schau doch mal auf <https://www.kaggle.com/uciml/mushroom-classification> vorbei. Dort gibt es Daten für über 8.000 Pilze – inklusive Informationen über ihre Geometrie, Farben usw. Lieber du als ich. (Du erinnerst dich daran, was weiter vorne im Buch steht? »Der Verlag und der Autor dieses Buches haften nicht für zufällige oder Folgeschäden im Zusammenhang mit oder infolge der Bezugnahme auf Informationen in diesem Buch oder der Verwendung der in diesem Buch enthaltenen Vorschläge.«)
- **Finanzen:** Unter <https://finance.yahoo.com> findest du viele ältere Aktien-, Index- und Wechselkursdaten. Für spezifische Aktien gehst

du zu

[https://finance.yahoo.com/quote/\[HierdeinTickersymboleinfügen\]/history](https://finance.yahoo.com/quote/[HierdeinTickersymboleinfügen]/history)

Die Bank of England hat darüber hinaus auch

Wirtschaftsindikatoren, Zinssätze usw. unter

<https://www.bankofengland.co.uk/statistics/> veröffentlicht.

- **Banknoten:** Daten zu den Eigenschaften von Banknoten findest du unter <http://archive.ics.uci.edu/ml/datasets/banknote+authentication>.
- **Tiere:** Daten zu den Eigenschaften verschiedener Tiere findest du unter <http://archive.ics.uci.edu/ml/datasets/zoo>.
- **Schauspieler:** Beliebte Schauspieler und Schauspielerinnen findest du unter <https://today.yougov.com/ratings/entertainment/popularity/all-time-actors-actresses/all>.



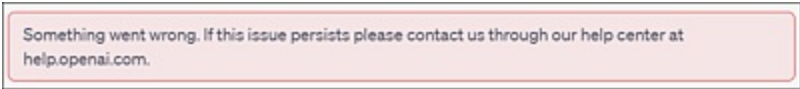
# Danksagung

Ich möchte meiner Lieblings-*Middleschool*-Schülerin Genevieve Wilmott und dem Mathematiklehrer Heerpal Sahota an der *Latymer Upper School* für ihre Vorschläge ganz herzlich danken.

Ich danke auch den Nachlassverwaltern von Donald Michie dafür, dass ich zwei Fotografien von ihm verwenden durfte.

Danke auch an Bing für seine Fotogenerierungsfunktion. Das Cover der englischen Originalausgabe wurde vom talentierten Liam Larkin mithilfe von Bildern von *Freepik.com* gestaltet.

Schließlich habe ich noch ChatGPT gebeten, eine Danksagung an sich selbst zu schreiben. Das kam dabei heraus:

A screenshot of a ChatGPT error message displayed in a light pink box with a thin border. The text inside the box reads: "Something went wrong. If this issue persists please contact us through our help center at help.openai.com."

Something went wrong. If this issue persists please contact us through our help center at [help.openai.com](https://help.openai.com).

**Abbildung A.1** Auf meinen Prompt hin gibt ChatGPT eine Fehlermeldung aus: »Etwas ist schiefgelaufen. Wenn dieses Problem weiterhin besteht, kontaktieren Sie uns bitte über unser Hilfe-Center auf [help.openai.com](https://help.openai.com).«

Und ich schätze, das ist ein hervorragendes Schlusswort.

## Über Paul Wilmott

Paul ist angewandter Mathematiker. Er forscht in der Mathematik, lehrt Mathematik und schreibt über Mathematik.

Seine Liebe zur Mathematik begann an der *Stanton Road County*-Grundschule in Bebington, Merseyside. Er denkt oft liebevoll an seine erste Hausaufgabe zurück, Seiten um Seiten von Bruchmanipulation. Darauf folgte die *Wirral Grammar School* für Jungen, wo er so viele mathematische Fächer wie möglich belegte. Anschließend studierte er Mathematik am *St Catherine's College* in Oxford, wo er auch seinen Dokortitel erhielt.

Paul war zwischenzeitlich professioneller Jongleur bei den *Dab Hands* und verdeckter Ermittler für *Channel 4*. Er hat dreimal die Auszeichnung »Half Blue« im Gesellschaftstanz an der *Oxford University* erhalten. Und er spielt die Ukulele.



# Stichwortverzeichnis

↓ **A** ↓ **B** ↓ **C** ↓ **D** ↓ **E** ↓ **G** ↓ **H** ↓ **I** ↓ **J** ↓ **K** ↓ **L** ↓ **M** ↓ **N** ↓ **O** ↓ **P** ↓ **R** ↓ **S** ↓ **T**  
↓ **U** ↓ **V** ↓ **W** ↓ **Z**

**A** ↑

---

Abhängige Variable [→ 3.1 Fachjargon und mathematische Konzepte]

Aktion [→ 9.2 Im Detail]

Alan Turing [→ 2.2 Der Turing-Test und die Geburtsstunde der KI]

Albert Einstein [→ 3.1 Fachjargon und mathematische Konzepte]

Algorithmus [→ 3.1 Fachjargon und mathematische Konzepte]

AlphaGo [→ 2.5 DeepMind]

Aufteilungsmethode [→ 7.3 Aus dem Leben gegriffen: Ein Beispiel]

## B ↑

---

Belohnung [→ 9.2 Im Detail]

BERT [→ 2.7 Schlechte Nachrichten ...]

Bilderkennung [→ 8.1 Wofür neuronale Netze eingesetzt werden]

Bildsegmentierung [→ 6.1 Wofür Clustering eingesetzt wird]

Blackjack [→ 9.2 Im Detail]

## C ↑

---

Charles Babbage [→ Vorwort]

ChatGPT [→ 1.3 ChatGPT] [→ 1.3 ChatGPT]

Claude Shannon [→ 7.2 Im Detail]

Clustering [→ 3.2 Technische Grundlagen] [→ 6.1 Wofür Clustering eingesetzt wird]

*dichtebasiertes* [→ 6.2 Im Detail]

## D ↑

---

Data Mining [→ 4.1 Wofür K-Nearest-Neighbours eingesetzt werden] [→ 5.1 Wofür Regression eingesetzt wird] [→ 7.1 Wofür

Entscheidungsbäume eingesetzt werden] [→ 8.1 Wofür neuronale Netze eingesetzt werden]

Daten [→ 3.1 Fachjargon und mathematische Konzepte]

Deep Blue [→ 2.4 Deep Blue]

DeepFake [→ Vorwort]

DeepMind [→ 2.5 DeepMind]

Dimension [→ 3.1 Fachjargon und mathematische Konzepte]

Dokumentklassifizierung [→ 6.1 Wofür Clustering eingesetzt wird]

Donald Knuth [→ 7.4 Dein Projekt]

Donald Michie [→ 2.3 Michie und die Streichholzschachteln]

Drei Gesetze der Robotik [→ 2.8 Die »Drei Gesetze der Robotik«] [→ 2.8 Die »Drei Gesetze der Robotik«]

## E ↑

---

Edgar Allan Poe [→ 7.4 Dein Projekt]

Einarmiger Bandit [→ 9.3 Aus dem Leben gegriffen: Ein Beispiel]

Empfehlungssysteme [→ 6.1 Wofür Clustering eingesetzt wird]

Entscheidungsbaum [→ 3.2 Technische Grundlagen] [→ 7.1 Wofür Entscheidungsbäume eingesetzt werden] [→ 7.3 Aus dem Leben gegriffen: Ein Beispiel]

Erklärbarkeit [→ 7.1 Wofür Entscheidungsbäume eingesetzt werden]

Euklidischer Abstand [→ 3.1 Fachjargon und mathematische Konzepte]

## G ↑

---

Galgenmännchen [→ 7.4 Dein Projekt]

Gehirn [→ 3.2 Technische Grundlagen]

Generative KI [→ 2.7 Schlechte Nachrichten ...]

Generative Pre-trained Transformer (GPT) [→ 1.3 ChatGPT]

## H ↑

---

Halluzinieren [→ 2.7 Schlechte Nachrichten ...]

House of Commons [→ 6.3 Aus dem Leben gegriffen: Ein Beispiel] [→ Datensammlung]

## I ↑

---

Input [→ 3.1 Fachjargon und mathematische Konzepte]

Intrusion Detection [[→ 4.1 Wofür K-Nearest-Neighbours eingesetzt werden](#)]

Isaac Asimov [[→ 2.1 Psychohistorie](#)]

Isaac Newton [[→ 5.3 Aus dem Leben gegriffen: Ein Beispiel](#)]

## J

---

James Cameron [[→ Vorwort](#)]

## K

---

Klasse [[→ 3.1 Fachjargon und mathematische Konzepte](#)] [[→ 3.2 Technische Grundlagen](#)] [[→ 5.1 Wofür Regression eingesetzt wird](#)]

Klassifikation [[→ 3.1 Fachjargon und mathematische Konzepte](#)] [[→ 4.4 Dein Projekt](#)] [[→ 6.1 Wofür Clustering eingesetzt wird](#)]

k-Means-Algorithmus [[→ 6.2 Im Detail](#)] [[→ 6.2 Im Detail](#)]

k-Means-Clustering → siehe [k-Means-Algorithmus]

K-Nearest-Neighbours [[→ 3.2 Technische Grundlagen](#)] [[→ 4.1 Wofür K-Nearest-Neighbours eingesetzt werden](#)] [[→ 4.1 Wofür K-Nearest-Neighbours eingesetzt werden](#)] [[→ 4.1 Wofür K-Nearest-Neighbours eingesetzt werden](#)] [[→ 4.2 Im Detail](#)] [[→ 4.3 Aus dem Leben gegriffen: Ein Beispiel](#)]



KNN → siehe [K-Nearest-Neighbours]

Kostenfunktion [→ 3.1 Fachjargon und mathematische Konzepte] [→ 3.1 Fachjargon und mathematische Konzepte] [→ 5.2 Im Detail] [→ 5.3 Aus dem Leben gegriffen: Ein Beispiel]

## L ↑

---

Lee Sedol [→ 2.5 DeepMind]

## M ↑

---

Machine Educable Noughts And Crosses Engine [→ 2.3 Michie und die Streichholzschachteln] [→ 2.3 Michie und die Streichholzschachteln] [→ 3.2 Technische Grundlagen]

Machine Learning [→ 1.1 Definitionssache] [→ 1.1 Definitionssache] [→ 2.1 Psychohistorie] [→ 3.1 Fachjargon und mathematische Konzepte]

Manhattan-Distanz [→ 3.1 Fachjargon und mathematische Konzepte]

Marktsegmentierung [→ 6.1 Wofür Clustering eingesetzt wird]

Maschinelles Lernen → siehe [Machine Learning]

Mastermind [→ 7.4 Dein Projekt]

Mathematische Modellierung [→ 3.1 Fachjargon und mathematische Konzepte]

Mathematisches Modell [→ 3.1 Fachjargon und mathematische Konzepte] [→ 3.1 Fachjargon und mathematische Konzepte]

*qualitatives* [→ 3.1 Fachjargon und mathematische Konzepte]

*quantitatives* [→ 3.1 Fachjargon und mathematische Konzepte]

MENACE → siehe [Machine Educable Noughts And Crosses Engine]

Merkmale [→ 3.1 Fachjargon und mathematische Konzepte]

Merkmalsauswahl [→ 7.1 Wofür Entscheidungsbäume eingesetzt werden]

Merkmalsextraktion [→ 8.1 Wofür neuronale Netze eingesetzt werden]

Methode der kleinsten Quadrate (MKQ) [→ 5.3 Aus dem Leben gegriffen: Ein Beispiel]

Minimax-Prinzip [→ 7.4 Dein Projekt]

Mustererkennung [→ 4.1 Wofür K-Nearest-Neighbours eingesetzt werden] [→ 5.1 Wofür Regression eingesetzt wird] [→ 7.1 Wofür Entscheidungsbäume eingesetzt werden] [→ 8.1 Wofür neuronale Netze eingesetzt werden]

## N ↑

---

Nächste Nachbarn → siehe [K-Nearest-Neighbours]

Neuronales Netz [→ 3.2 Technische Grundlagen] [→ 8.1 Wofür neuronale Netze eingesetzt werden]

Nick Cave [→ 2.7 Schlechte Nachrichten ...]

## O ↑

---

Optimierung [→ 3.1 Fachjargon und mathematische Konzepte]  
[→ 3.1 Fachjargon und mathematische Konzepte]

Output [→ 3.1 Fachjargon und mathematische Konzepte]

## P ↑

---

Parameter [→ 3.1 Fachjargon und mathematische Konzepte]

Psychohistorie [→ 2.1 Psychohistorie]

## R ↑

---

Regression [→ 3.2 Technische Grundlagen] [→ 5.1 Wofür Regression eingesetzt wird] [→ 5.2 Im Detail] [→ 5.3 Aus dem Leben gegriffen: Ein Beispiel]

Reinforcement Learning [→ 3.2 Technische Grundlagen] [→ 9.1 Wofür Reinforcement Learning eingesetzt wird] [→ 9.2 Im Detail]

Repräsentative Vektoren [→ 6.3 Aus dem Leben gegriffen: Ein Beispiel]

Robotik [→ 9.1 Wofür Reinforcement Learning eingesetzt wird]

## S ↑

---

Satz des Pythagoras [→ 3.1 Fachjargon und mathematische Konzepte]

Schmetterlingseffekt [→ 3.1 Fachjargon und mathematische Konzepte]

Schwerkraft [→ 5.3 Aus dem Leben gegriffen: Ein Beispiel]

Selbstorganisierende Karten [→ 6.2 Im Detail] [→ 6.3 Aus dem Leben gegriffen: Ein Beispiel]

Skalierung [→ 4.2 Im Detail] [→ 5.4 Dein Projekt]

Skalierung von Daten [→ 3.1 Fachjargon und mathematische Konzepte] [→ 3.1 Fachjargon und mathematische Konzepte]

Spracherkennung [→ 8.1 Wofür neuronale Netze eingesetzt werden]

Sprachverarbeitung [→ 8.1 Wofür neuronale Netze eingesetzt werden]

## T ↑

---

Tic Tac Toe [→ 2.3 Michie und die Streichholzschachteln]

[→ 2.3 Michie und die Streichholzschachteln]

Training [→ 3.1 Fachjargon und mathematische Konzepte]

Trial and Error [→ 9.1 Wofür Reinforcement Learning eingesetzt wird]

Turing-Test [→ 2.2 Der Turing-Test und die Geburtsstunde der KI]

## U ↑

---

Überanpassung [→ 3.1 Fachjargon und mathematische Konzepte]

Überwachtes Lernen [→ 3.2 Technische Grundlagen] [→ 9.1 Wofür Reinforcement Learning eingesetzt wird]

Ukulele [→ 3.1 Fachjargon und mathematische Konzepte]

Unabhängige Variable [→ 3.1 Fachjargon und mathematische Konzepte]

Unteranpassung [→ 3.1 Fachjargon und mathematische Konzepte]

Unüberwachtes Lernen [→ 3.2 Technische Grundlagen] [→ 9.1 Wofür Reinforcement Learning eingesetzt wird]

## V ↑

---

Vektor [[→ 3.1 Fachjargon und mathematische Konzepte](#)] [[→ 3.1 Fachjargon und mathematische Konzepte](#)]

Verborgene Schichten [[→ 8.2 Im Detail](#)] [[→ 8.4 Dein Projekt](#)]

Verhaltenspsychologie [[→ 9.1 Wofür Reinforcement Learning eingesetzt wird](#)]

Verlustfunktion → siehe [[Kostenfunktion](#)]

## W ↑

---

Wordle [[→ 7.4 Dein Projekt](#)]

## Z ↑

---

Zustand [[→ 9.2 Im Detail](#)]

# Rechtliche Hinweise

Das vorliegende Werk ist in all seinen Teilen urheberrechtlich geschützt. Weitere Hinweise dazu finden Sie in den Allgemeinen Geschäftsbedingungen des Anbieters, bei dem Sie das Werk erworben haben.

Die automatisierte Analyse des Werkes, um daraus Informationen insbesondere über Muster, Trends und Korrelationen gemäß § 44b UrhG (»Text und Data Mining«) zu gewinnen, ist untersagt.

## Markenschutz

Die in diesem Werk wiedergegebenen Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. können auch ohne besondere Kennzeichnung Marken sein und als solche den gesetzlichen Bestimmungen unterliegen.

## Haftungsausschluss

Ungeachtet der Sorgfalt, die auf die Erstellung von Text, Abbildungen und Programmen verwendet wurde, können weder Verlag noch Autor\*innen, Herausgeber\*innen, Übersetzer\*innen oder Anbieter für mögliche Fehler und deren Folgen eine juristische Verantwortung oder irgendeine Haftung übernehmen.



# Über den Autor



**Paul Wilmott** vermittelt angewandte Mathematik – mit Kultstatus. Seine unverwechselbaren Einführungen bringen seit Jahrzehnten Licht in finanzmathematische Modelle, Derivatehandel, Risikobewertung und Co. Wilmott forscht, lehrt und schreibt nicht nur, er war auch schon als Profi-Jongleur und Under-Cover-Ermittler (Channel 4) im Einsatz, arbeitet für das CQF-Institut (Quantitative Finance) und für die Bildungsinitiative "Math on Toast" (Mathematik für Familien).

# **Dokumentenarchiv**

Das Dokumentenarchiv umfasst alle Abbildungen und ggf. Tabellen und Fußnoten dieses E-Books im Überblick.

**MAELZEL'S EXHIBITION,**  
*No. 29, St. James's Street.*

---

**The**

**Automaton**



**Chess**

**Player**

Being returned from *Edinburgh* and *Liverpool*, where (giving the Pawn and Move) it baffled all Competition, in upwards of 200 Games, although opposed by ALL THE BEST PLAYERS.

**Has opened its Second Campaign,**  
 WITH THE ADDITION OF THE  
**AUTOMATON TRUMPETER,**  
 AND THE  
***Conflagration of Moscow,***

In which Mr. M. has endeavoured to combine the ARTS of DESIGN, MECHANISM, and MUSIC, so as produce, by a novel Imitation of Nature, a perfect Fac Simile of the real Scene. The View is from an elevated Station on the Fortress of the *Kremlin*, at the Moment when the Inhabitants are evacuating the Capital of the Czars, and the Head of the French Columns commences its Entry. The gradual Progress of the Fire, the hurrying Bustle of the Fugitives, the Eagerness of the Invaders, and the Din of warlike Sounds, will tend to impress the Spectator with a true Idea of a Scene which baffles all Powers of Description.

---

The MORNING EXHIBITIONS begin at 1 and 3 o'Clock, and the EVENING EXHIBITION at 8 precisely, when GAMES will be played AGAINST ANY OPPONENT, to whom the double Advantage of A PAWN AND THE MOVE WILL BE GIVEN.

*Admission 2s.6d. Children 1s.6d. each.*

Each Exhibition lasts One Hour. Should a Game not be finished in that Time, the Party will be at Liberty to take it down with a View to its being resumed at another Opportunity.

*Mr. M. begs leave to announce that the ORCHESTRION, the AUTOMATON TRUMPETER, the CONFLAGRATION OF Moscow, and the Patent for the METRONOMES, are to be disposed of.*

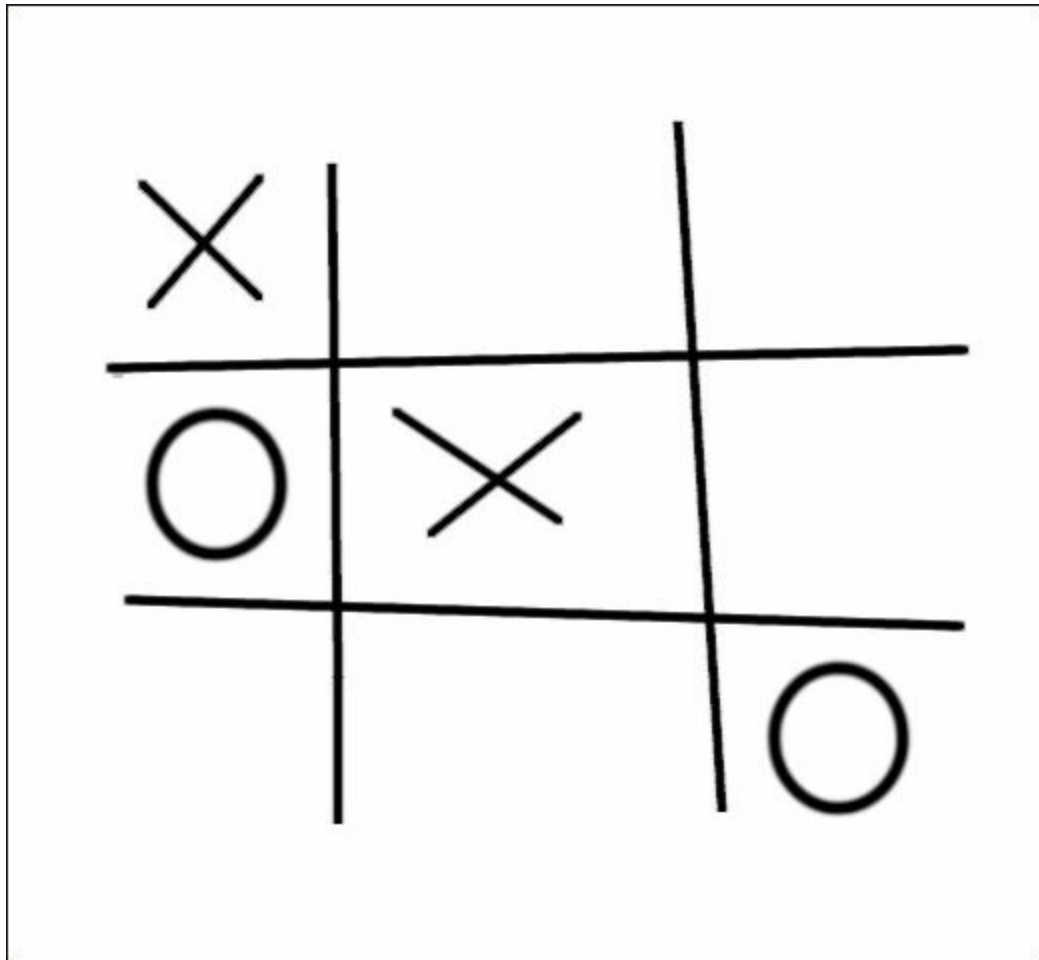
**Abbildung 1** Ein Ausstellungsflyer für den »Mechanical Turk«, einen Schachautomaten aus dem Jahr 1770



**Abbildung 2**    *Ein Filmausschnitt aus dem Film »Metropolis«*



**Abbildung 2.1** *Donald Michie, der »Vater der künstlichen Intelligenz in Großbritannien«*

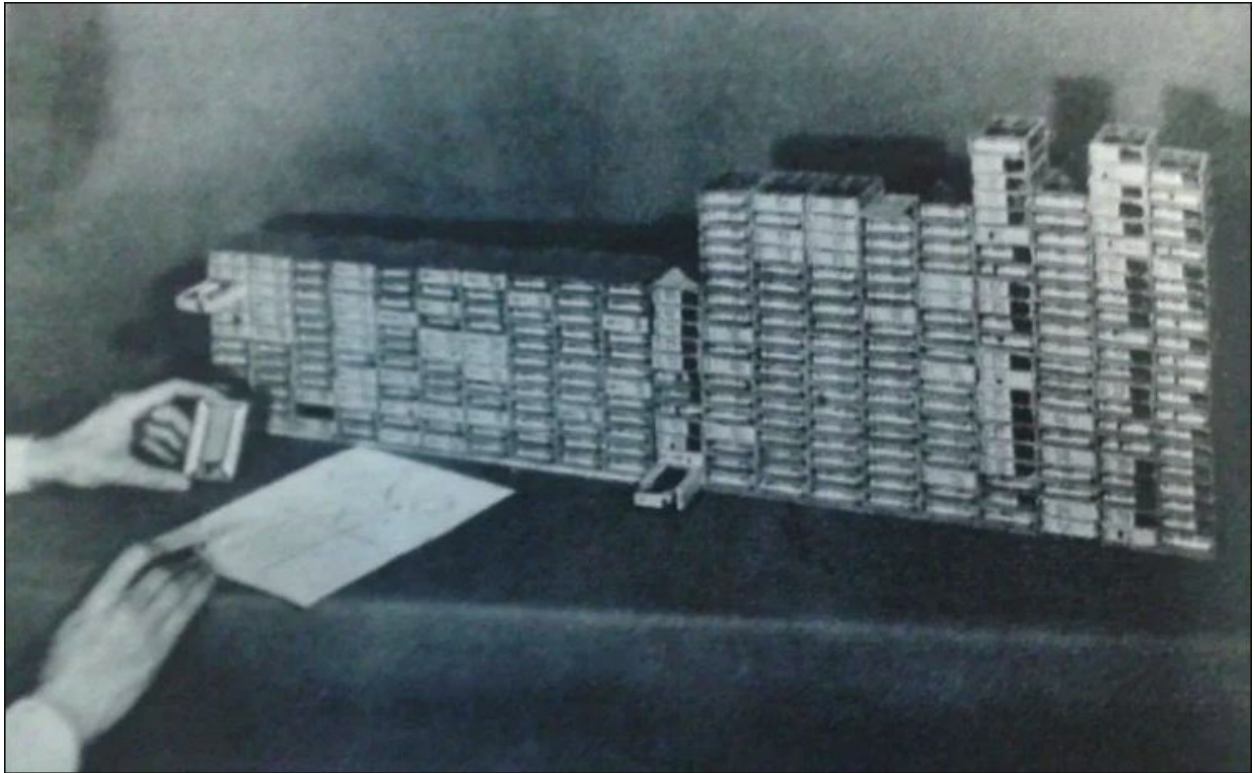


**Abbildung 2.2** *Ein Tic-Tac-Toe-Spielstand, wie ihn Michie mit seinen Streichholzschachteln abgebildet hat*



**Abbildung 2.3** *Eine Darstellung von Michies perlengefüllten Streichholzschachteln*

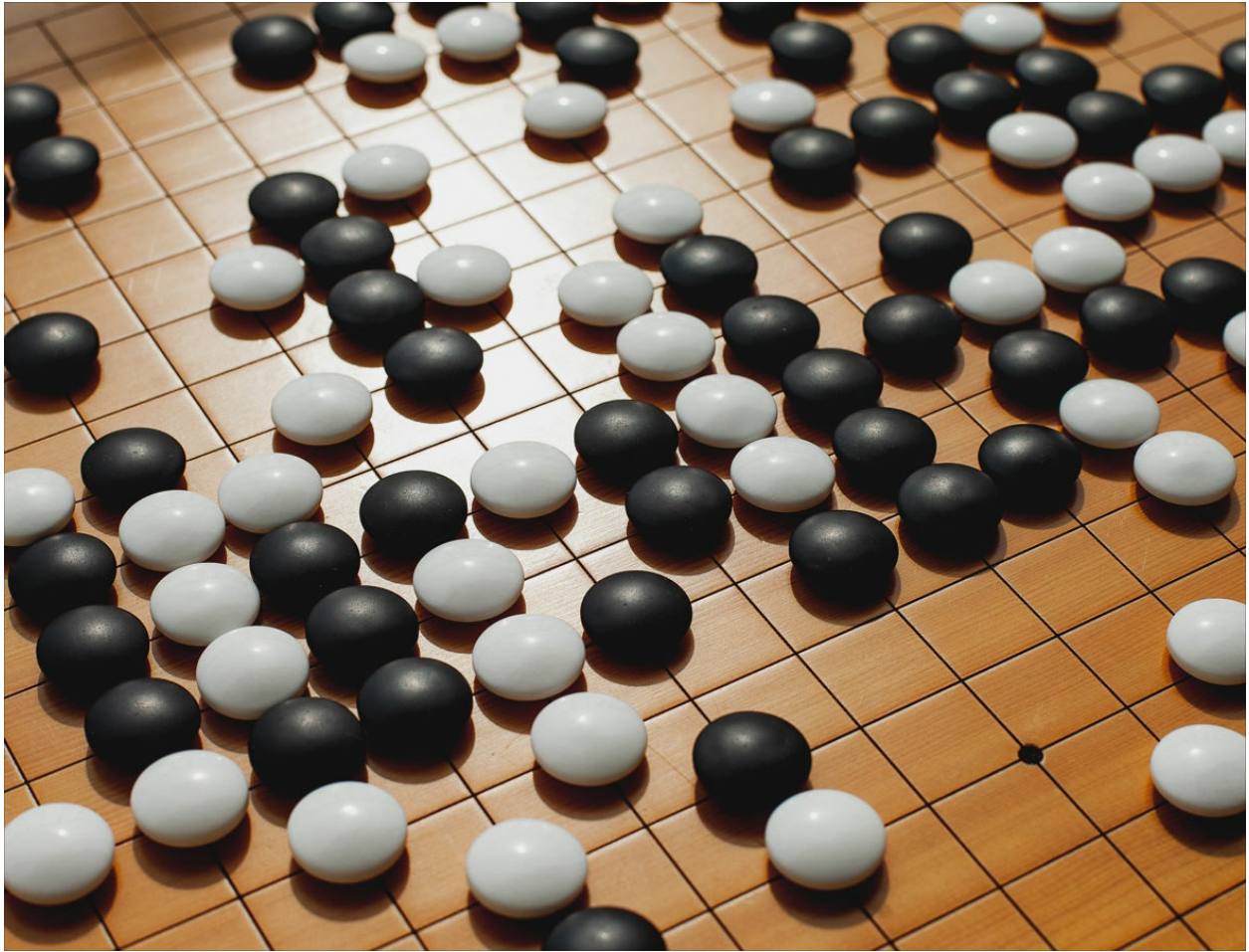




**Abbildung 2.4**    *Michies Streichholzschachteln auf einer Fotografie*







**Abbildung 2.5**    *Nahaufnahme von einem Go-Spielbrett*





**Abbildung 2.6**    *Die (KI-generierte) Zeichnung eines Blauwals*





**Abbildung 2.7**    *Das KI-generierte Bild eines gealterten Elvis'*







**Abbildung 3.1**    *Mathematische Modellierung für die Schwingbewegung einer Schaukel*





**Abbildung 3.2** *Die Fotografie eines Hundes*

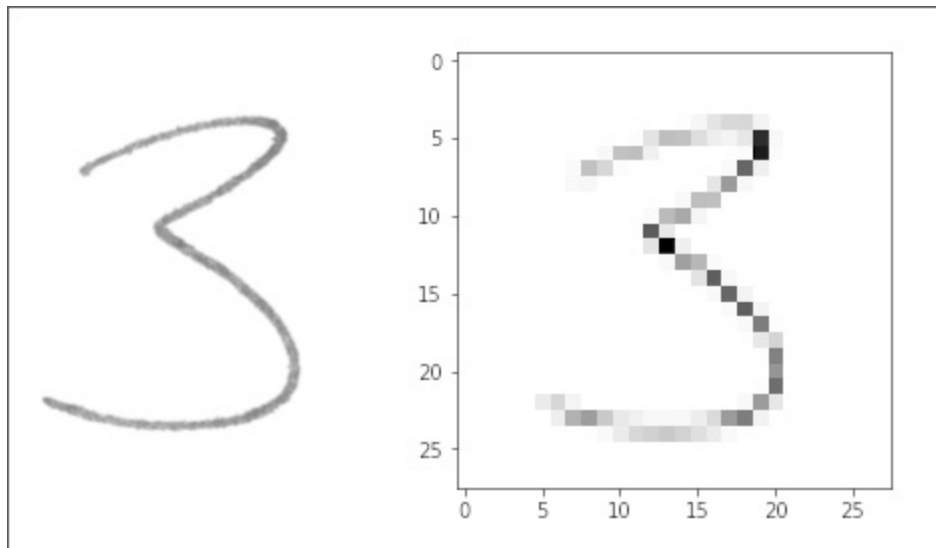
	A	B	C	D	E	F	G	H	I	J	K	
1	Hound											
2	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]
3	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]	[255,255,255]
4	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]	[253,255,252]	[253,255,252]	[254,255,253]	[254,255,253]	[254,255,253]	[254,255,253]
5	[253,255,252]	[253,255,250]	[253,255,252]	[254,255,251]	[254,255,253]	[254,255,251]	[253,255,252]	[253,255,250]	[254,255,253]	[254,255,251]	[253,255,252]	[253,255,252]
6	[248,253,247]	[249,254,247]	[250,255,249]	[251,255,249]	[251,255,250]	[251,255,249]	[251,255,250]	[251,255,249]	[250,255,249]	[250,255,248]	[249,254,248]	[248,253,247]
7	[246,251,244]	[247,253,243]	[248,253,246]	[250,255,246]	[250,255,248]	[250,255,246]	[250,255,248]	[250,255,246]	[247,252,245]	[247,253,243]	[247,252,245]	[246,251,244]
8	[244,252,241]	[245,253,240]	[246,254,243]	[248,255,243]	[248,255,245]	[248,255,243]	[248,255,245]	[247,255,242]	[245,253,242]	[245,253,240]	[245,253,242]	[244,252,241]
9	[244,252,239]	[245,253,240]	[246,254,241]	[247,255,242]	[248,255,243]	[247,255,242]	[246,254,241]	[246,254,241]	[246,254,241]	[246,254,241]	[245,253,240]	[244,252,239]
10	[243,254,238]	[243,254,238]	[243,254,238]	[243,254,238]	[243,254,238]	[242,253,237]	[242,253,237]	[242,253,237]	[242,253,237]	[243,254,238]	[244,255,239]	[243,254,238]
11	[243,254,238]	[243,254,238]	[243,254,238]	[243,254,238]	[243,254,238]	[242,253,237]	[242,253,237]	[242,253,237]	[240,251,235]	[241,252,236]	[242,253,237]	[243,254,238]
12	[242,255,238]	[242,255,238]	[242,255,238]	[242,255,238]	[242,255,238]	[241,254,237]	[241,254,237]	[241,254,237]	[238,251,234]	[238,251,234]	[239,252,235]	[242,253,237]
13	[242,255,238]	[242,255,238]	[242,255,238]	[242,255,238]	[242,255,238]	[241,254,237]	[241,254,237]	[240,253,236]	[238,251,234]	[238,251,234]	[238,251,234]	[239,252,235]
14	[243,254,240]	[243,254,240]	[244,255,241]	[244,255,241]	[243,254,240]	[242,253,239]	[241,252,238]	[241,252,238]	[240,251,237]	[240,251,237]	[240,251,237]	[243,254,240]
15	[243,254,240]	[243,254,240]	[244,255,241]	[244,255,241]	[244,255,241]	[242,253,239]	[241,252,238]	[240,251,237]	[242,253,239]	[241,252,238]	[241,252,238]	[243,254,240]
16	[243,253,242]	[243,253,242]	[244,254,243]	[244,254,243]	[244,254,243]	[242,252,241]	[241,251,240]	[240,250,239]	[242,252,241]	[241,251,240]	[241,251,240]	[243,253,242]
17	[243,253,242]	[243,253,242]	[244,254,243]	[244,254,243]	[244,254,243]	[242,252,241]	[241,251,240]	[240,250,239]	[241,251,240]	[240,250,239]	[240,250,239]	[243,253,242]
18	[244,255,241]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[242,253,239]	[242,253,239]	[242,253,239]	[241,252,238]	[244,255,241]
19	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[242,253,239]	[242,253,239]	[242,253,239]	[242,253,239]	[241,252,238]	[243,254,240]
20	[244,255,241]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[242,253,239]	[242,253,239]	[242,253,239]	[242,253,239]	[241,252,238]	[244,255,241]
21	[244,255,241]	[244,255,241]	[244,255,241]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[242,253,239]	[244,255,241]
22	[244,255,241]	[244,255,241]	[244,255,241]	[244,255,241]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[242,253,239]	[244,255,241]
23	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[243,254,240]	[242,253,239]	[242,253,239]	[242,253,239]	[242,253,239]	[241,252,238]	[241,252,238]	[243,254,240]
24	[242,253,239]	[242,253,239]	[242,253,239]	[241,252,238]	[241,252,238]	[241,252,238]	[241,252,238]	[241,252,238]	[239,250,236]	[240,251,237]	[240,251,237]	[242,253,239]
25	[241,252,238]	[240,251,237]	[240,251,237]	[240,251,237]	[240,251,237]	[240,251,237]	[239,250,236]	[239,250,236]	[237,248,234]	[238,249,235]	[239,250,236]	[241,252,238]
26	[239,250,236]	[239,250,236]	[239,250,236]	[239,250,236]	[239,250,236]	[238,249,235]	[238,249,235]	[238,249,235]	[239,250,236]	[239,250,236]	[239,250,236]	[239,250,236]
27	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[238,249,235]	[237,248,234]	[239,250,236]	[238,249,235]	[238,249,235]	[238,249,235]
28	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[239,250,236]	[238,249,235]	[237,248,234]	[237,248,234]
29	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[236,247,233]	[236,247,233]	[239,250,236]	[238,249,235]	[237,248,234]	[237,248,234]
30	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[236,247,233]	[236,247,233]	[238,249,235]	[237,248,234]	[237,248,234]	[237,248,234]
31	[237,248,234]	[237,248,234]	[237,248,234]	[237,248,234]	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[237,248,234]	[236,247,233]	[236,247,233]	[237,248,234]
32	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[236,247,233]	[235,246,232]	[235,246,232]	[235,246,232]	[235,246,232]	[235,246,232]	[235,246,232]	[236,247,233]

**Abbildung 3.3** Die digitalisierte Version des Hundebildes. Die Tabellenzellen stehen jeweils für einen Farbpixel.





**Abbildung 3.4** *Bunt gemusterte Socken hängen an der Wäscheleine – keine gleicht der anderen.*



**Abbildung 3.5** *Eine handgeschriebene 3 und ihre digitalisierte Version*



**Abbildung 3.6** *Ein herkömmliches Puzzle, wie es auch ein Kind lösen könnte*

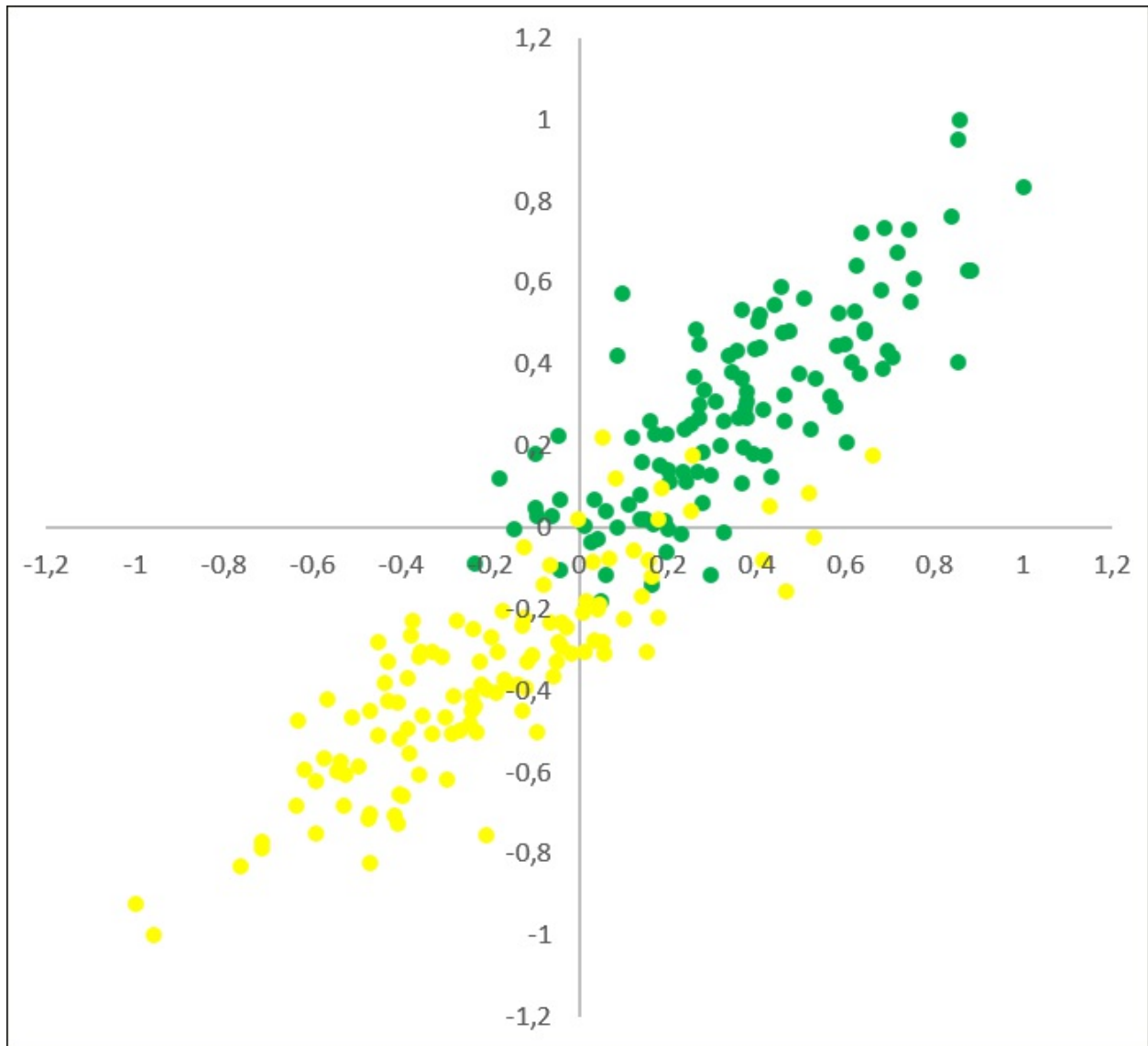




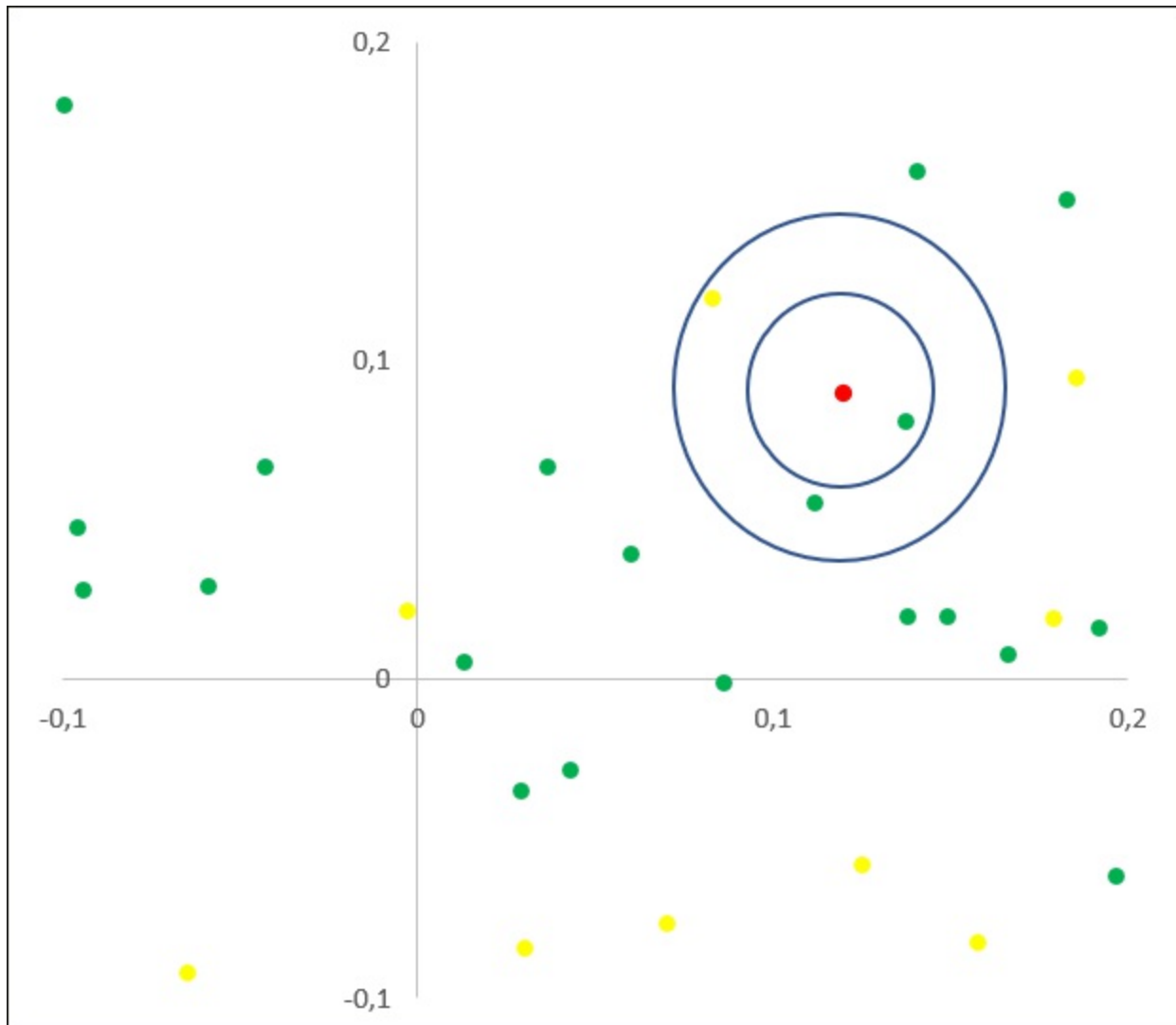
	A	B	C	D	E	F	G	H	
1		<b>Gender</b>	<b>Height</b>	<b>Weight</b>		<b>Gender</b>	<b>Height</b>	<b>Weight</b>	
2		Male	73,84702	241,8936		Female	58,91073	102,0883	
3		Male	68,7819	162,3105		Female	65,23001	141,3058	
4		Male	74,11011	212,7409		Female	63,369	131,0414	
5		Male	71,73098	220,0425		Female	64,48	128,1715	
6		Male	69,8818	206,3498		Female	61,7931	129,7814	
7		Male	67,25302	152,2122		Female	65,96802	156,8021	
8		Male	68,78508	183,9279		Female	62,85038	114,969	
9		Male	68,34852	167,9711		Female	65,65216	165,083	
10		Male	67,01895	175,9294		Female	61,89023	111,6762	
11		Male	63,45649	156,3997		Female	63,67787	104,1516	
12		Male	71,19538	186,6049		Female	68,10117	166,5757	
13		Male	71,64081	213,7412		Female	61,79888	106,2337	
14		Male	64,76633	167,1275		Female	63,37146	128,1182	
15		Male	69,28307	189,4462		Female	58,89589	101,6826	
16		Male	69,24373	186,4342		Female	58,43825	98,19262	
17		Male	67,64562	172,1869		Female	60,8098	126,9155	
18		Male	72,41832	196,0285		Female	70,12865	151,2543	
19		Male	63,97433	172,8835		Female	62,25743	115,7974	
20		Male	69,64006	185,984		Female	61,73509	107,8669	
21		Male	67,936	182,4266		Female	63,05956	145,5899	
22		Male	67,91505	174,1159		Female	62,28684	139,5227	
23		Male	69,43944	197,7314		Female	61,82748	122,7662	
24		Male	66,14913	149,1736		Female	66,34754	157,381	
25		Male	75,20597	228,7618		Female	65,32063	145,0374	
26		Male	67,8932	162,0067		Female	66,10387	148,6452	
27		Male	68,14403	192,344		Female	64,52718	132,6809	
28		Male	69,08963	184,4352		Female	56,54797	84,87212	
29		Male	72,80084	206,8282		Female	62,73928	138,5304	
30		Male	67,42124	175,2139		Female	61,5852	137,4253	
31		Male	68,49642	154,3426		Female	62,02442	124,6039	
32		Male	68,61811	187,5068		Female	65,31133	140,2667	
33		Male	74,03381	212,9102		Female	61,20496	108,2901	
34		Male	71,52822	195,0322		Female	67,02509	150,2192	
35		Male	69,18016	205,1836		Female	60,21117	119,0375	

**Abbildung 4.1** *Ein Auszug aus einer Tabelle, in der sich Größen- und Gewichtsdaten erwachsener Menschen befinden*



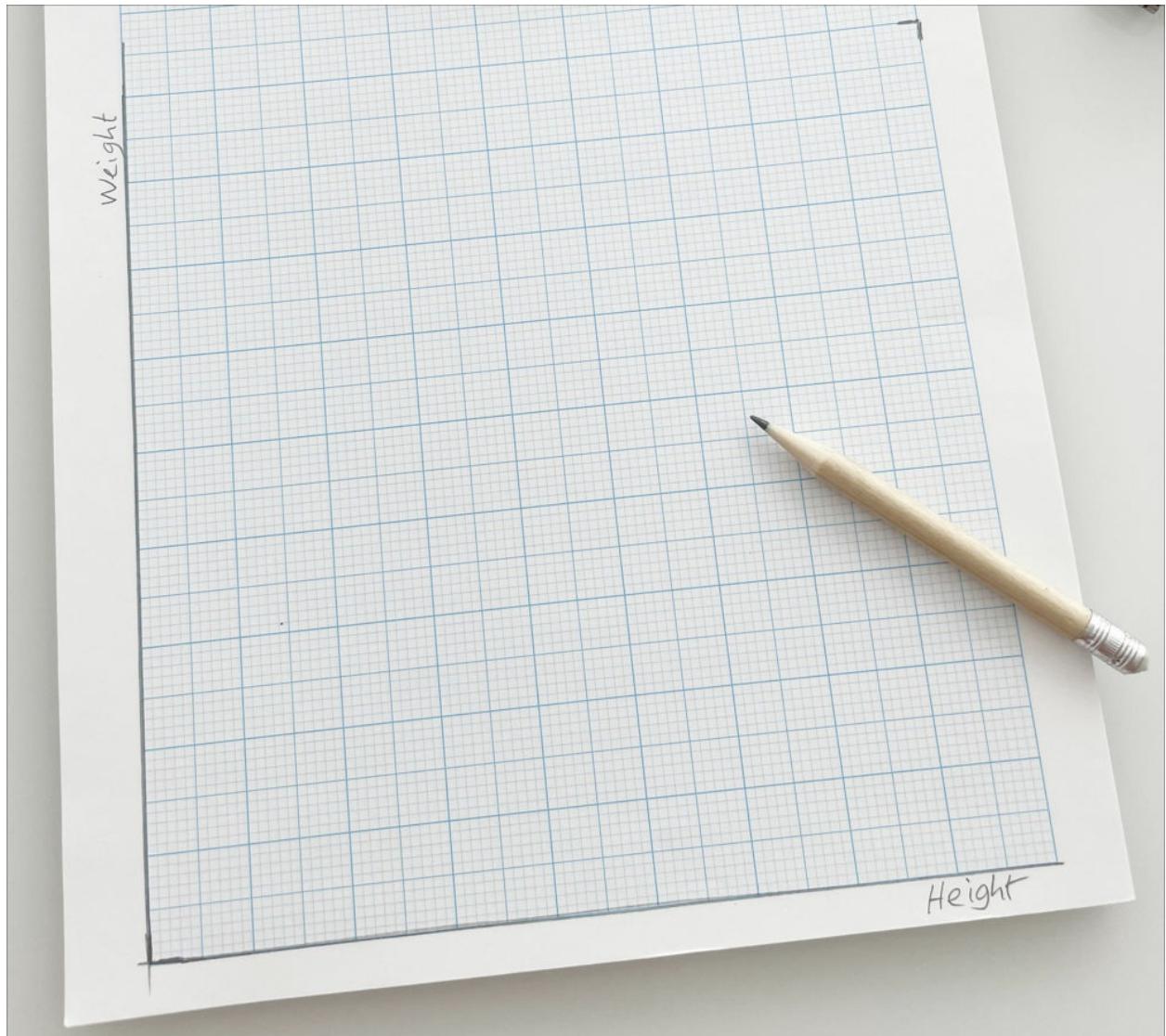


**Abbildung 4.2** *Die grafische Umsetzung der vorliegenden Größen- und Gewichtsdaten*



**Abbildung 4.3** *Eine Nahaufnahme der Datenpunkte aus*  
*Abbildung 4.2*

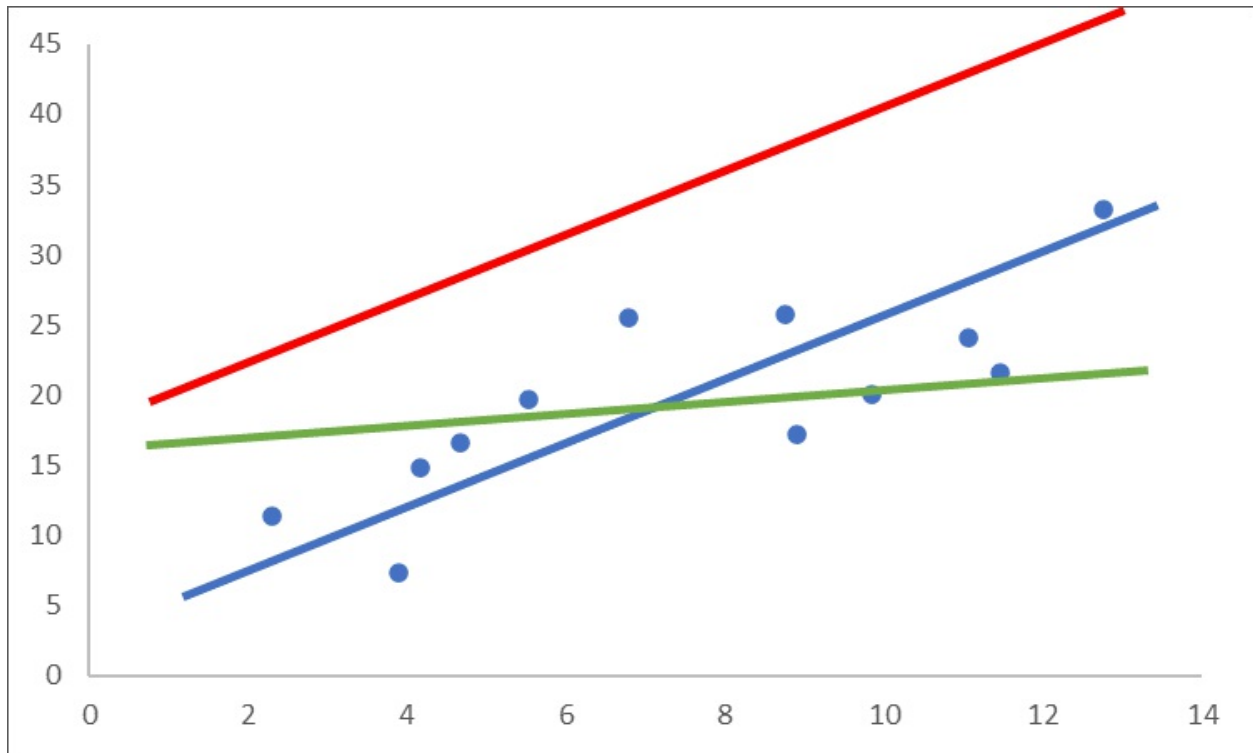




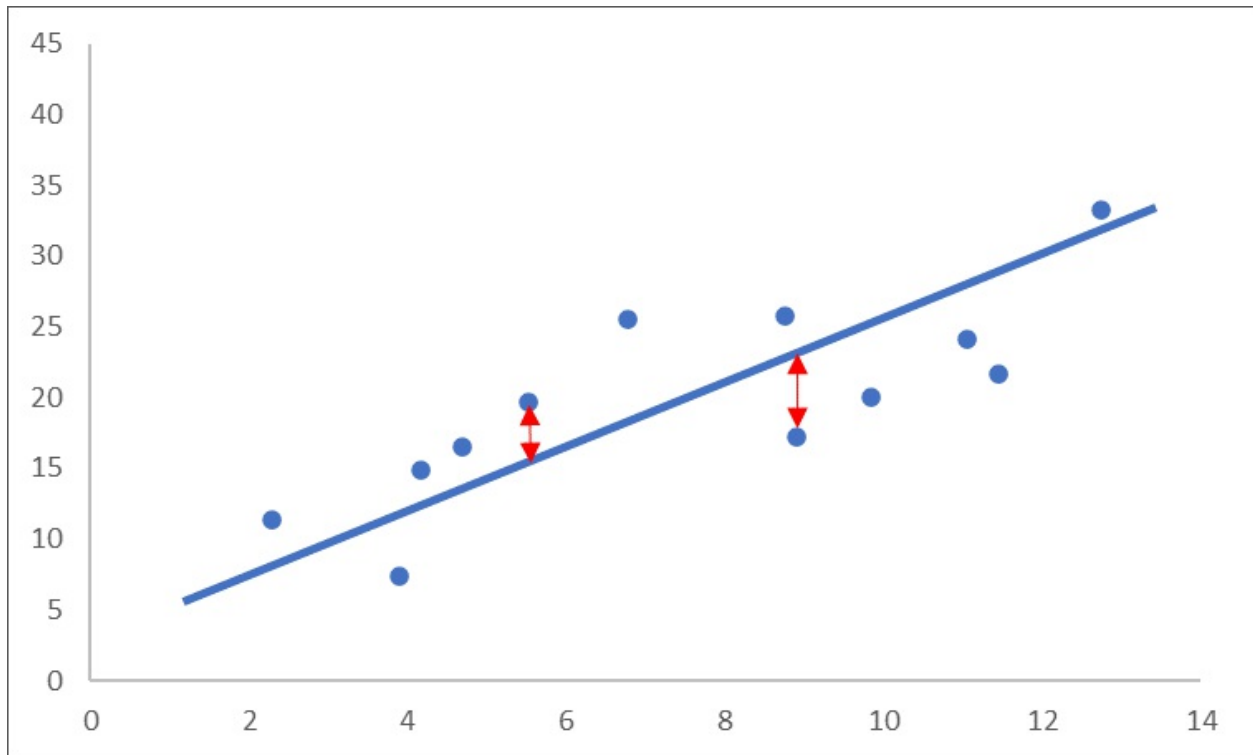
**Abbildung 4.4** *Millimeterpapier mit einem vorgezeichneten Quadrat darauf. Das ist die Vorbereitung für dein erstes Projekt.*





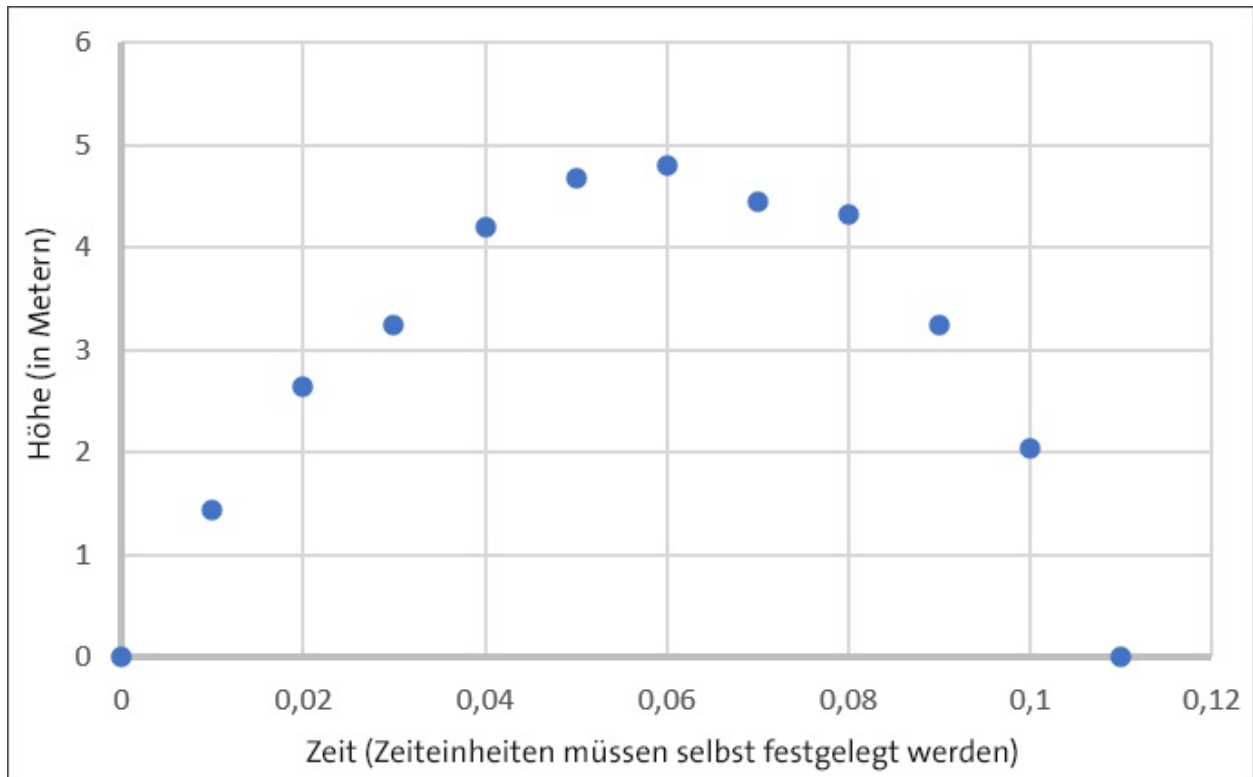


**Abbildung 5.1** *Drei mögliche Graphen für eine Gruppe an Daten*

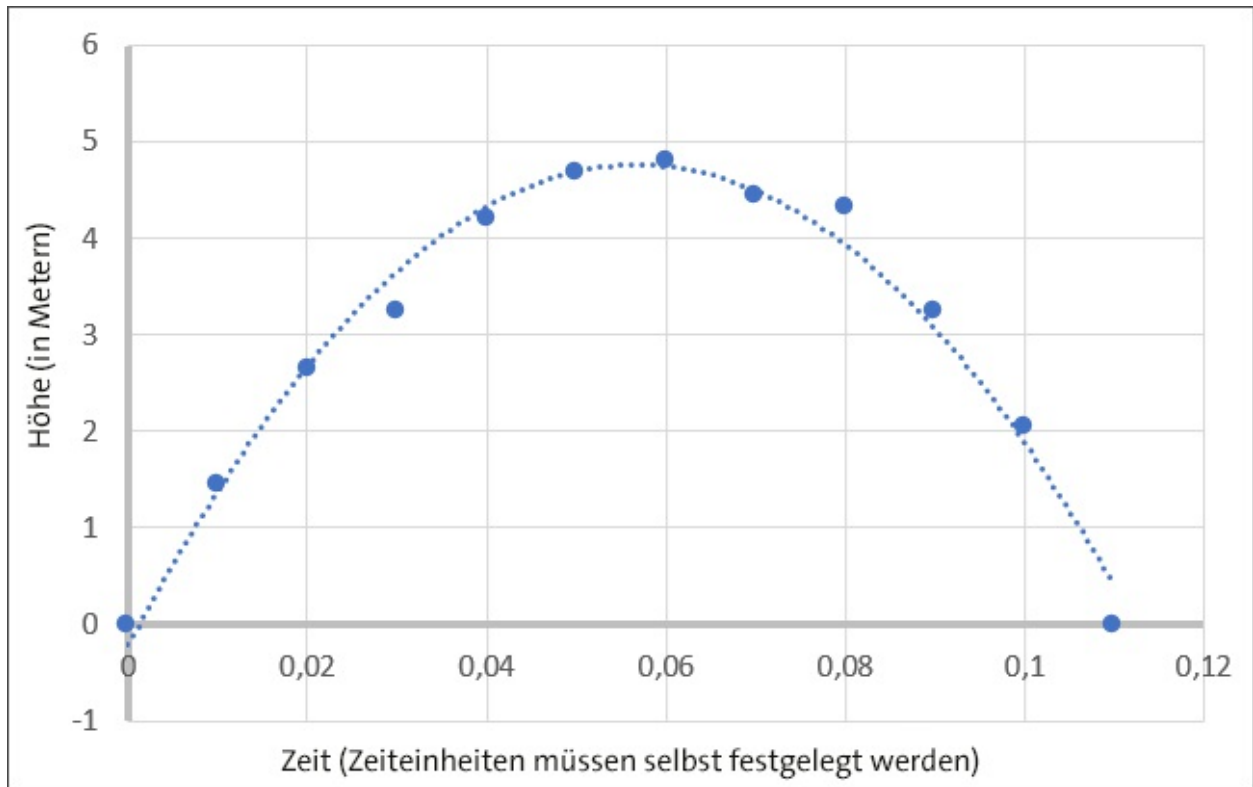


**Abbildung 5.2** *Abstände einzelner Datenpunkte zum ausgewählten Graphen*

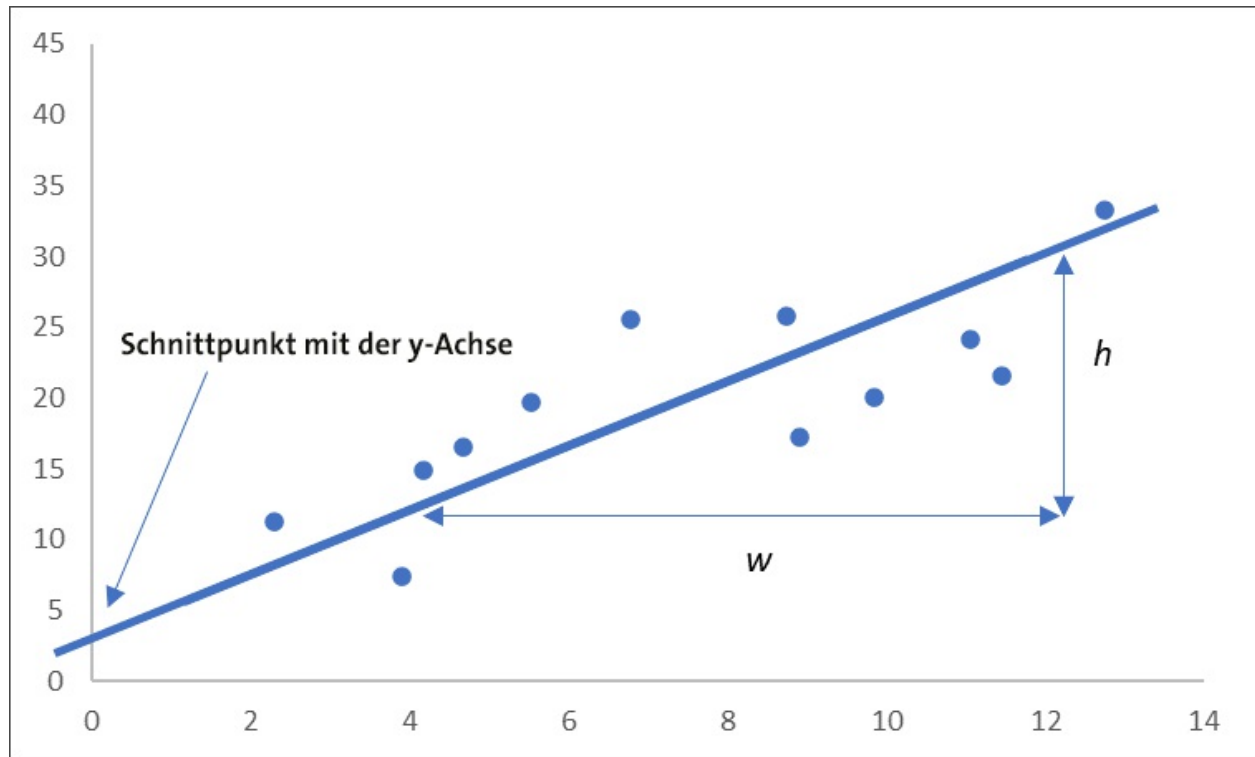




**Abbildung 5.3** Mit den Zeit- und Höhendaten, die wir aus unserem Video ermittelt haben, können wir einen solchen Graphen zeichnen.



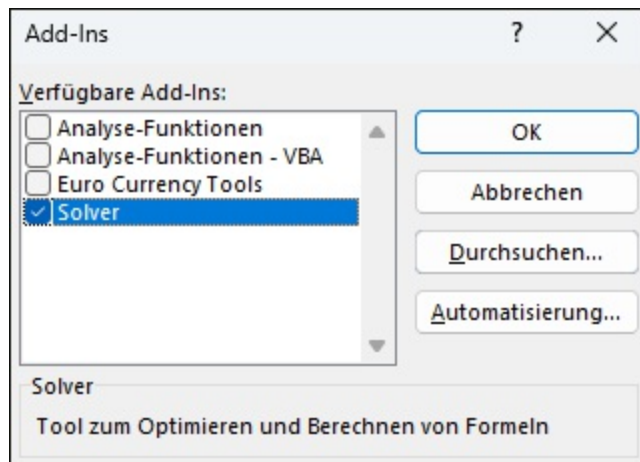
**Abbildung 5.4** *Im Diagramm ist neben den erhobenen Daten auch der Graph eingezeichnet, der den korrekten Kurvenverlauf der Beschleunigung des Balls anzeigt.*



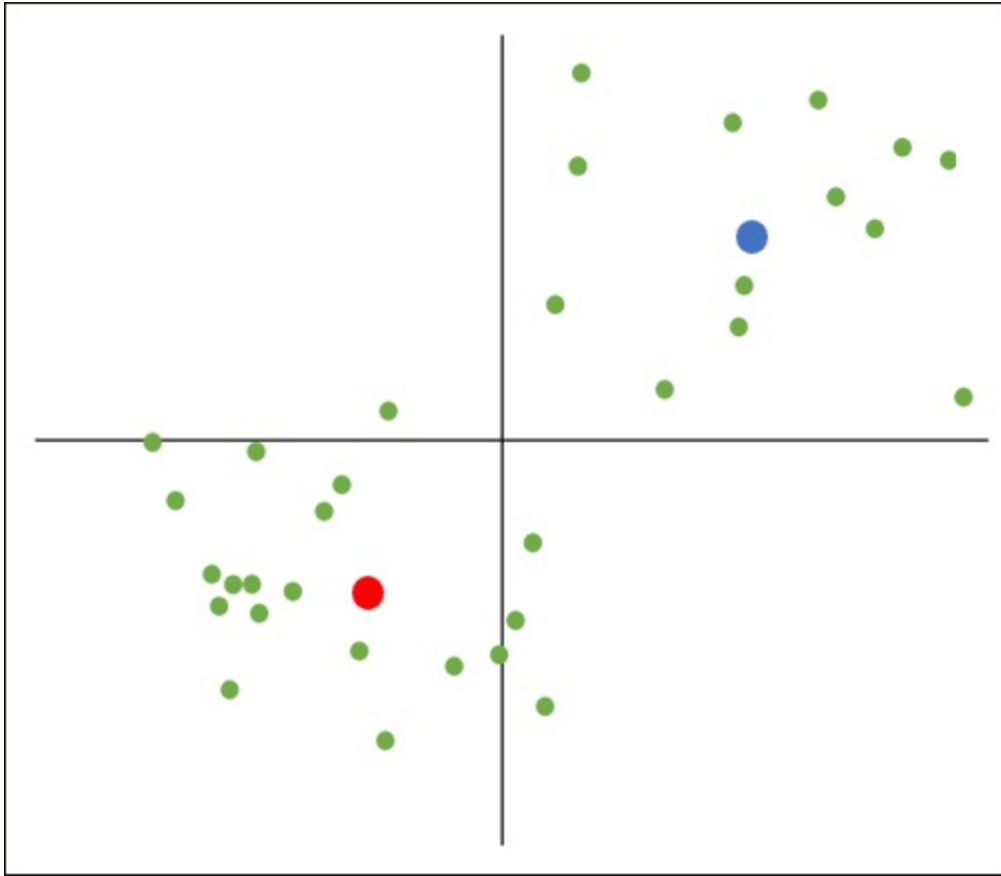
**Abbildung 5.5** Die Steigung des ausgewählten Graphen kann mithilfe von  $h$  und  $w$  berechnet werden.

	A	B	C	D	E	F	G
1			a	1,54465			
2			b	6,15445			
3							
4		x	y	Abweichung			
5		2,3	11,3	-1,59286	=B5*\$D\$1+\$D\$2-C5		
6		3,9	7,4	4,77857			
7		4,7	16,6	-3,18571			
8				35,5207	=SUMSQ(D5:D7)		
9							
10							

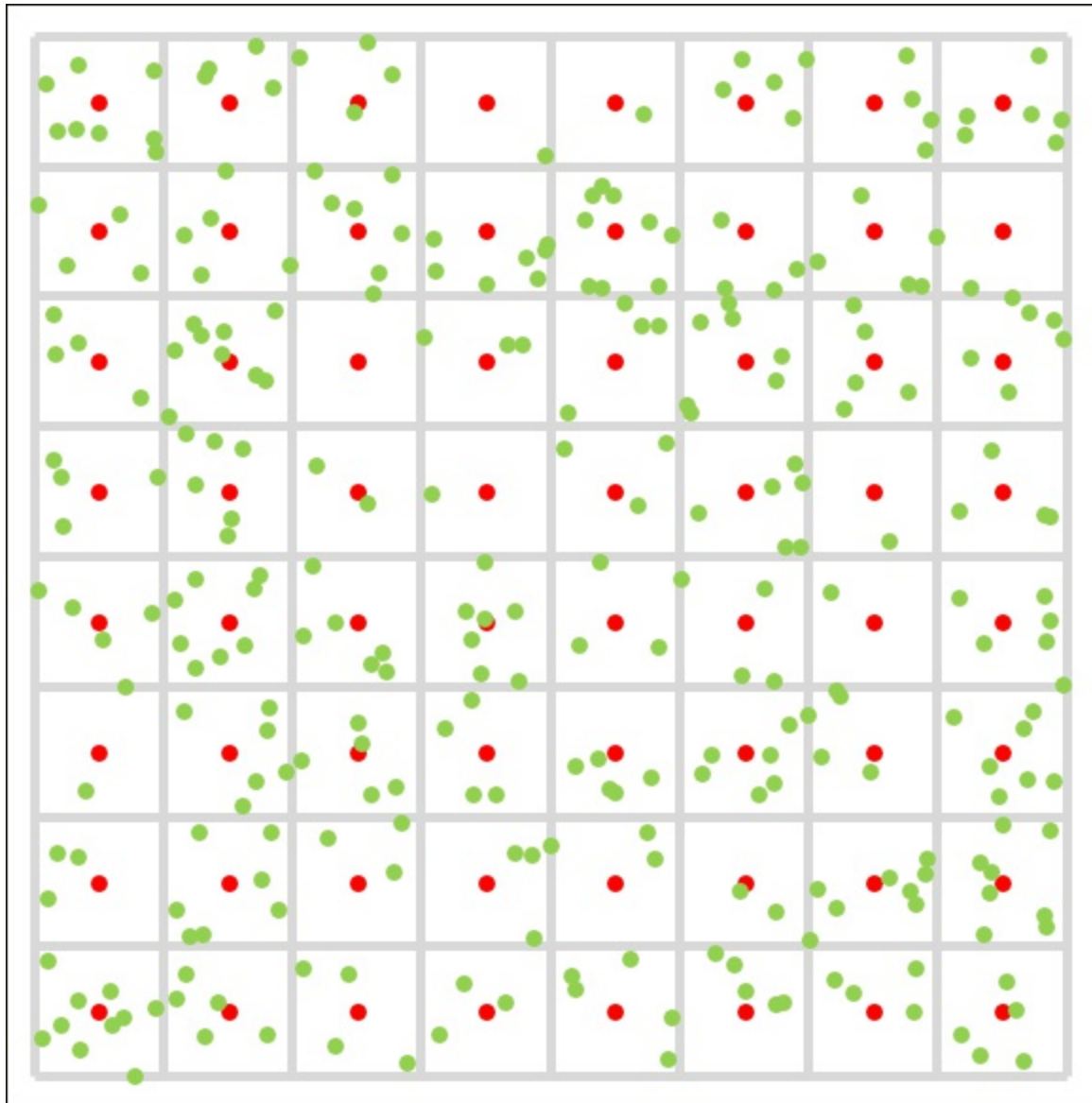
**Abbildung 5.6** In Excel lässt sich die Kostenfunktion auch schnell und simpel durch Ausprobieren ermitteln.



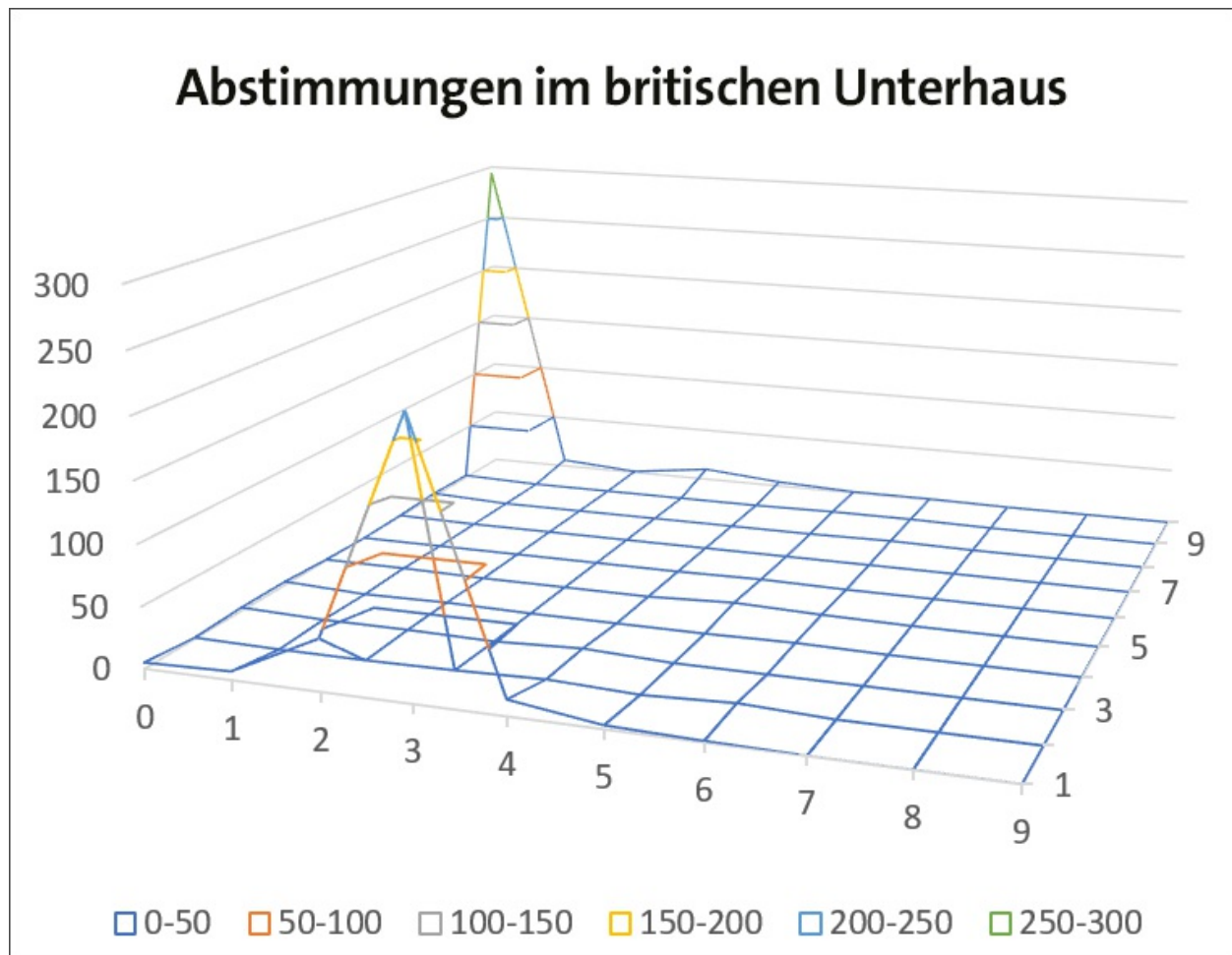
**Abbildung 5.7** Nach einem Klick auf »Add-Ins« in den Optionen von Excel kannst du den Solver mit einem Mausklick aktivieren.



**Abbildung 6.1** *In einem zweidimensionalen Diagramm werden Datenpunkte sowie zwei Cluster-Schwerpunkte dargestellt.*

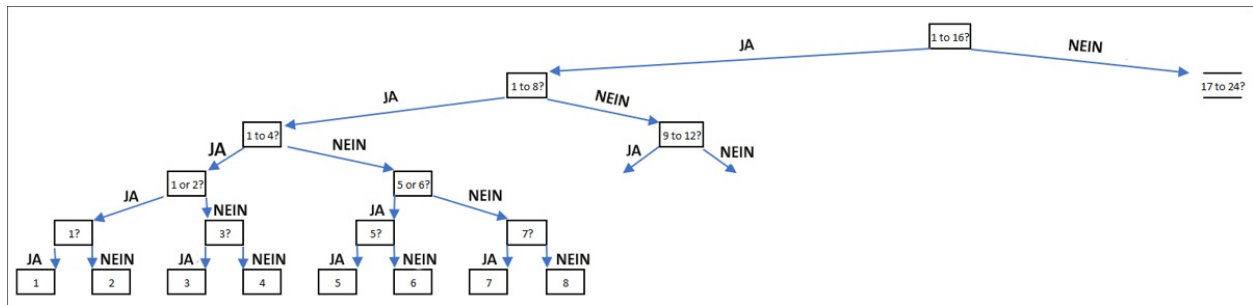


**Abbildung 6.2** In einem zweidimensionalen Diagramm werden Datenpunkte und ihre zugehörigen repräsentativen Vektoren dargestellt.

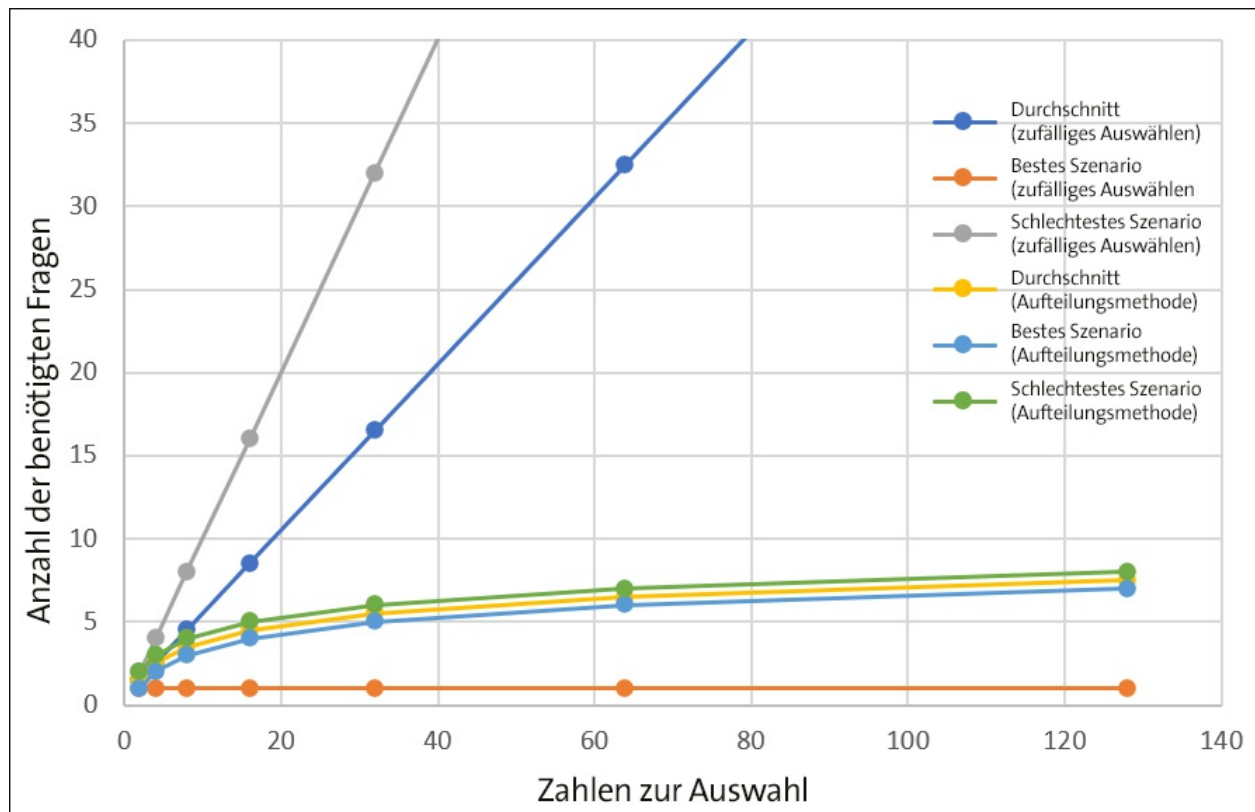


**Abbildung 6.3** Die Abstimmungen im britischen Unterhaus, in einem zweidimensionalen Diagramm dargestellt

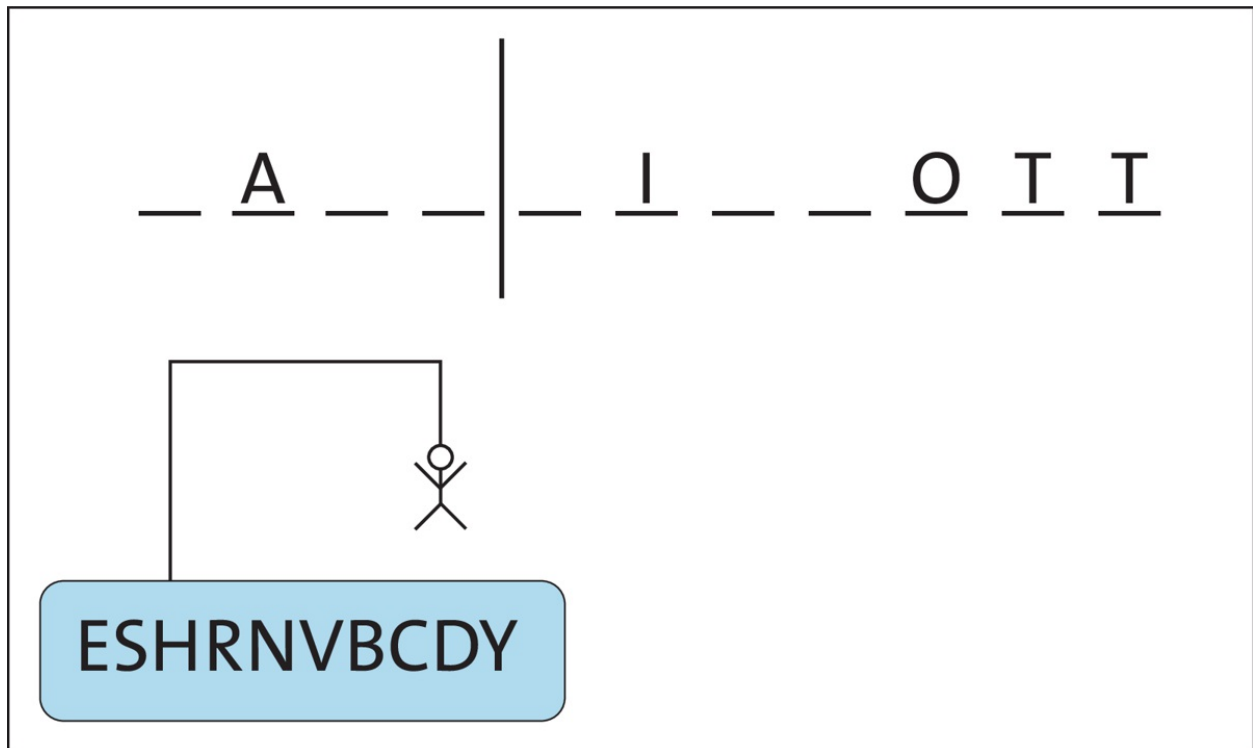




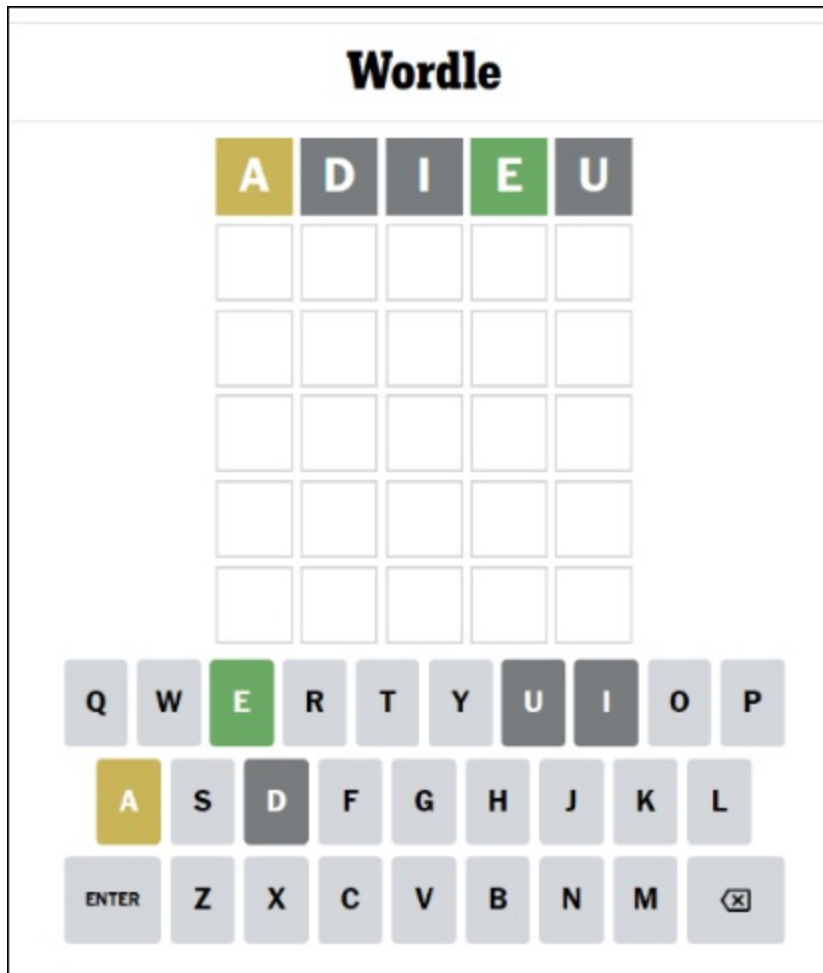
**Abbildung 7.1** *Der Entscheidungsbaum für die Suche nach einer zufälligen ganzen Zahl zwischen 1 und 32*



**Abbildung 7.2** Die Anzahl an Fragen (auf der y-Achse), die zum Erraten einer Zahl (x-Achse) benötigt werden, bei steigender Menge möglicher Zahlen



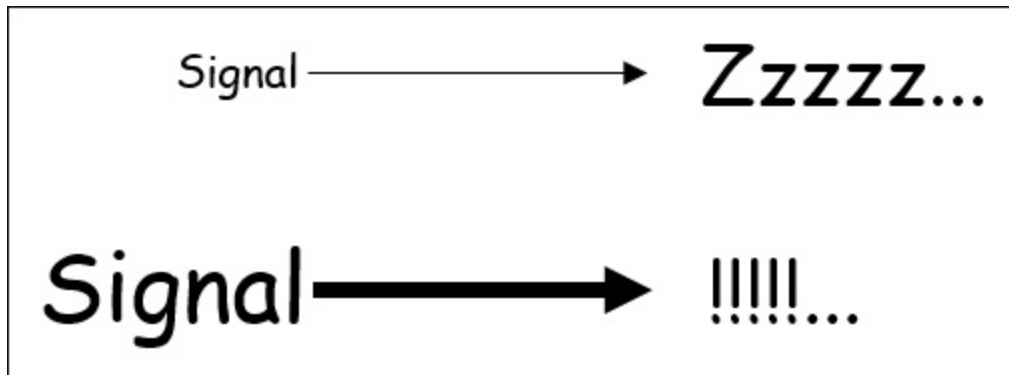
**Abbildung 7.3** *Finaler Stand einer Partie »Galgenmännchen« – das Galgenmännchen wurde vervollständigt, bevor die Lösung erraten werden konnte.*



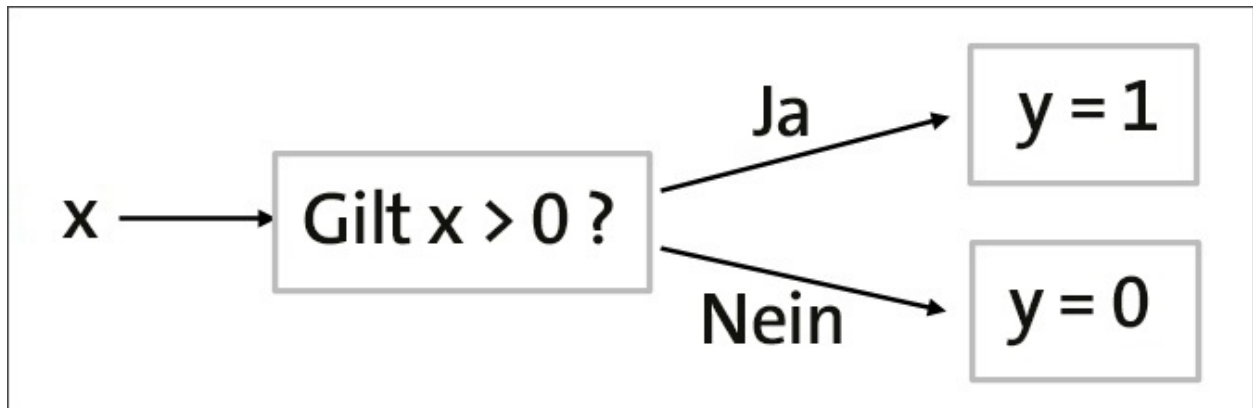
**Abbildung 7.4** Ein Wordle-Spiel nach dem Eingeben des ersten geratenen Wortes – der Buchstabe A kommt vor, steht aber noch nicht an der richtigen Stelle im Wort.



**Abbildung 7.5**    *Ein möglicher Spielstand einer Partie »Mastermind«*

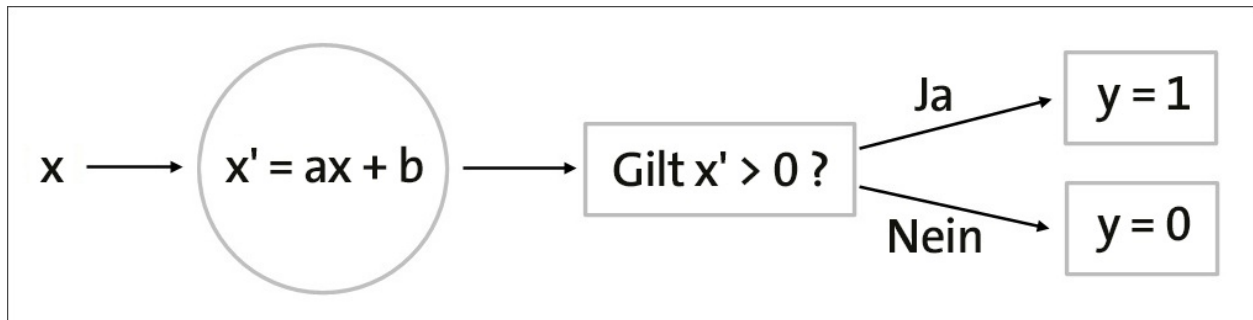


**Abbildung 8.1** *Die Funktionsweise von Neuronen im Gehirn, vereinfacht dargestellt*

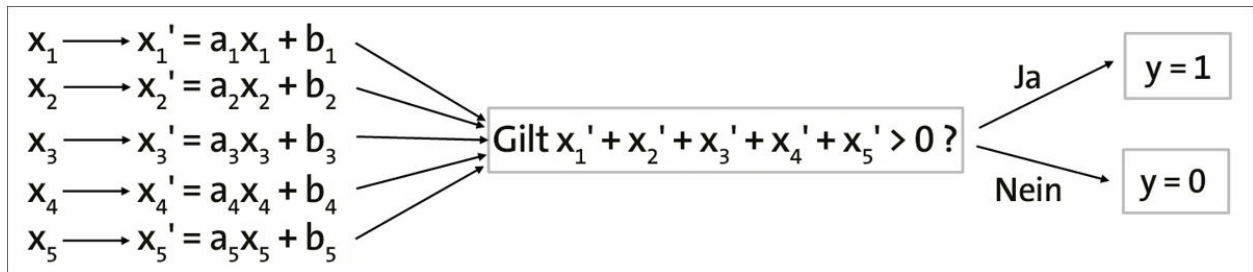


**Abbildung 8.2** *Schematische Darstellung der Funktionsweise von Neuronen im Gehirn*

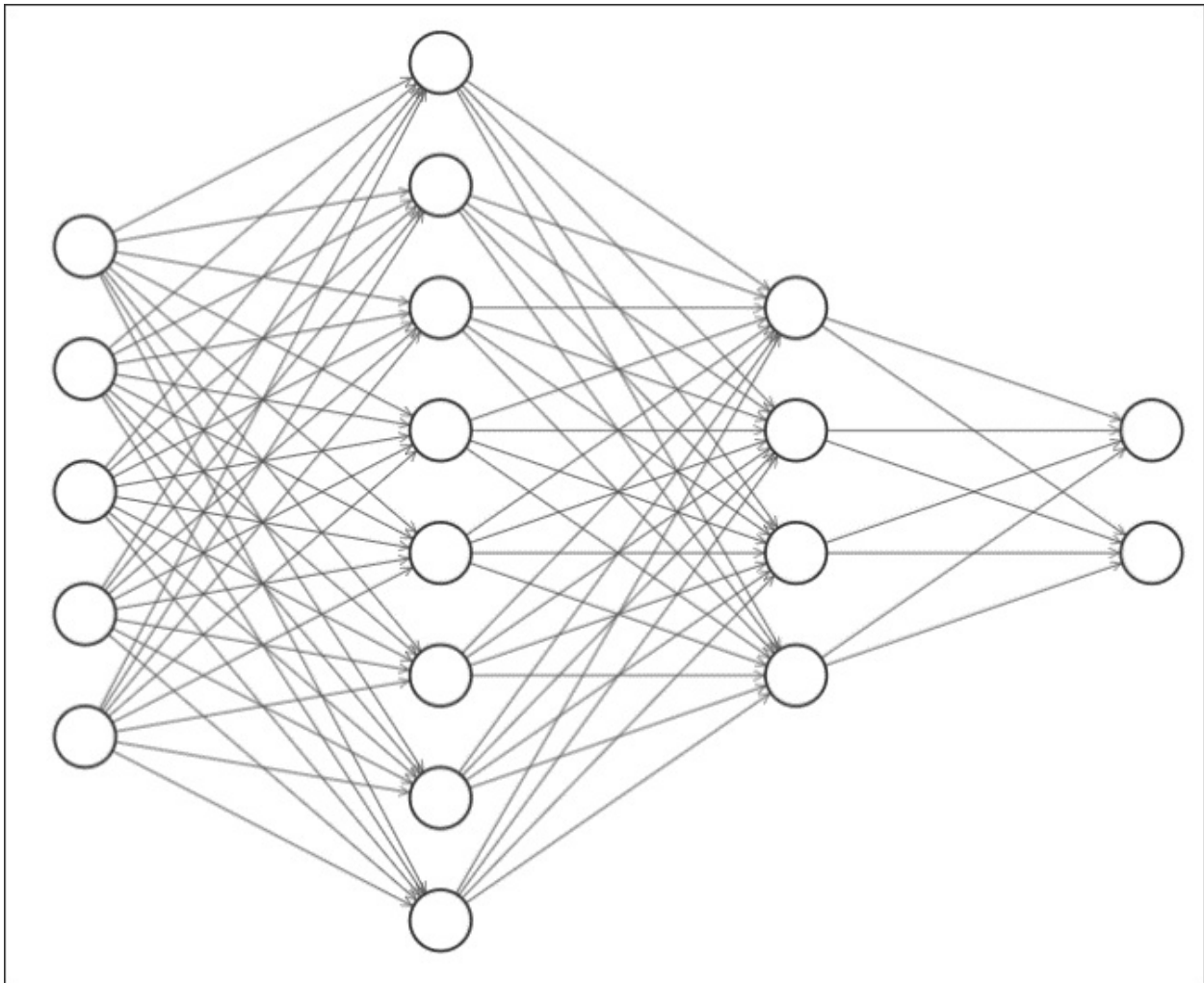




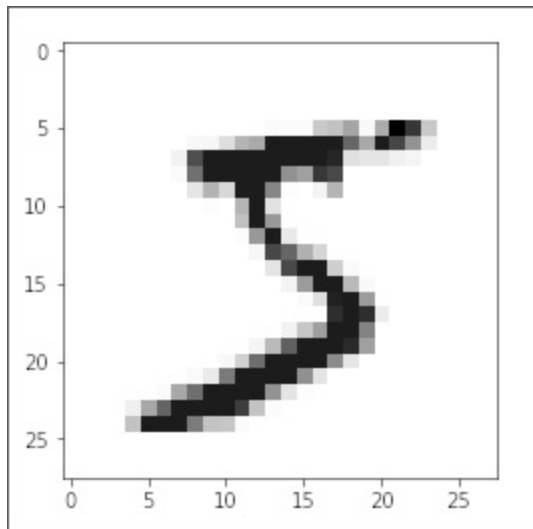
**Abbildung 8.3**    *Schematische Darstellung eines künstlichen neuronalen Netzes mit modifiziertem Input*



**Abbildung 8.4**    *Ein neuronales Netz, das einen Vektor als Input verarbeitet*

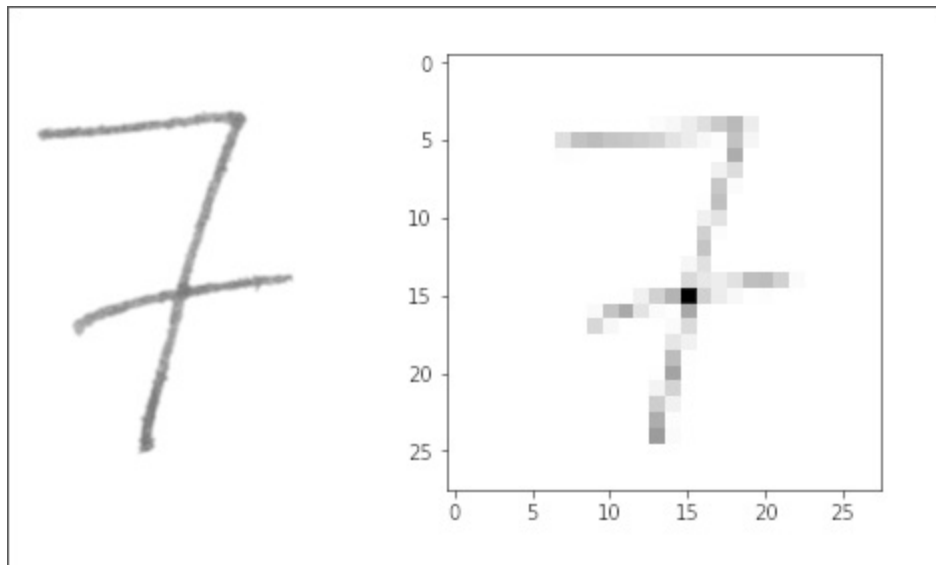


**Abbildung 8.5**    *Schematische Darstellung eines neuronalen Netzes mit mehreren Berechnungsebenen*

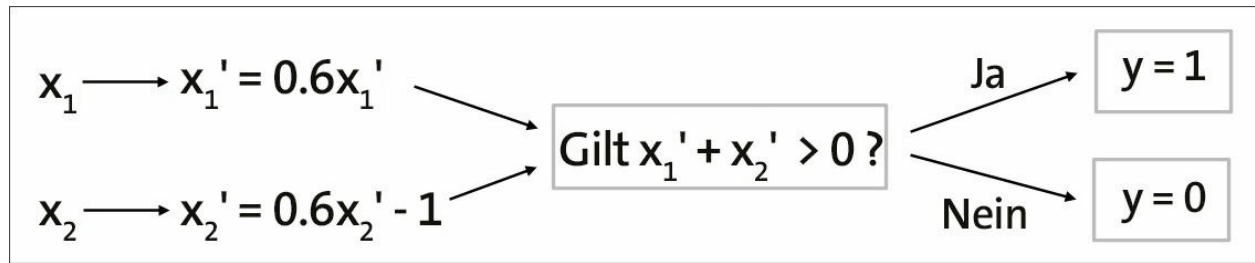


**Abbildung 8.6** *Die digitalisierte Abbildung einer handschriftlichen Ziffer*

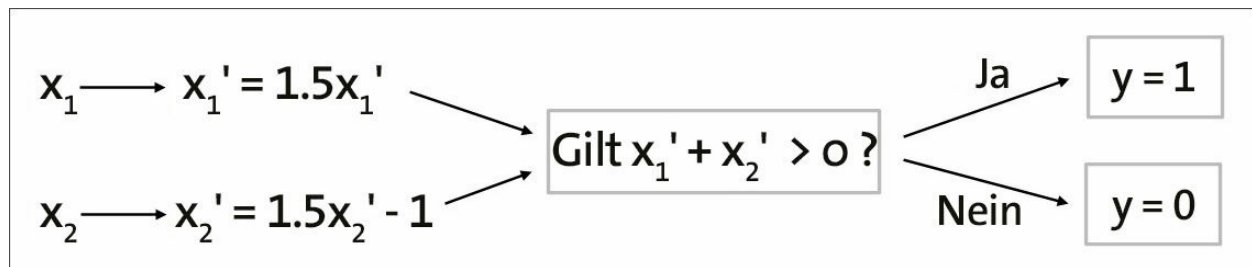




**Abbildung 8.8** *Eine von mir handschriftlich verfasste 7 mit ihrer digitalisierten Entsprechung*

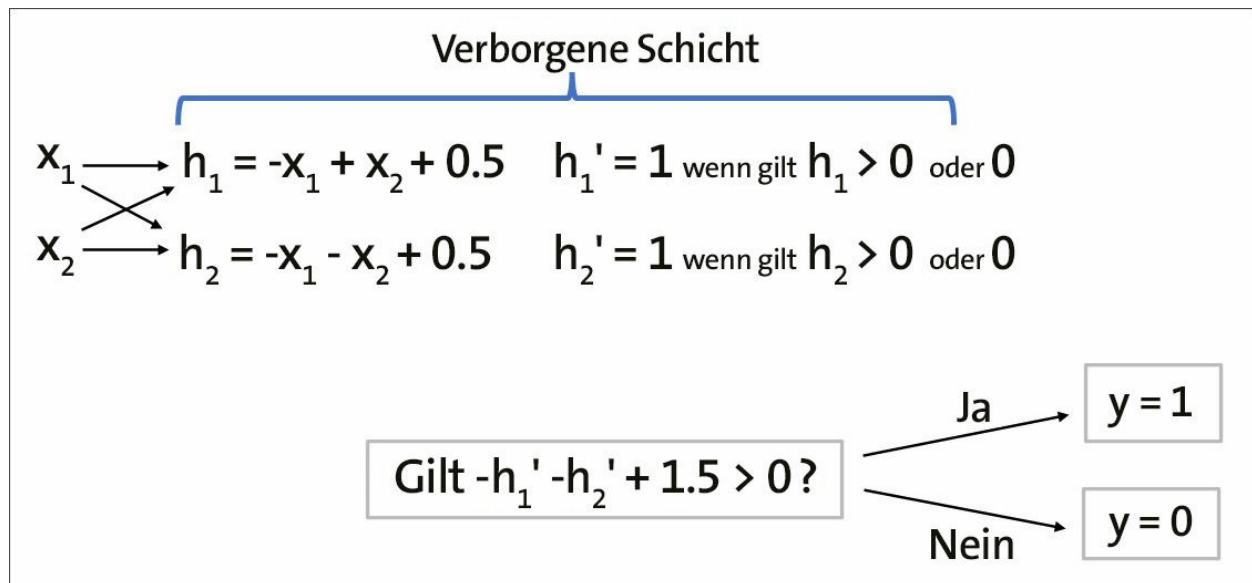


**Abbildung 8.9** *Der erste Beispielalgorithmus ist von einer bekannten Logikfunktion abgeleitet.*



**Abbildung 8.10** *Ein zweiter Beispiellgorithmus, um die Funktionsweise neuronaler Netze besser kennenzulernen*

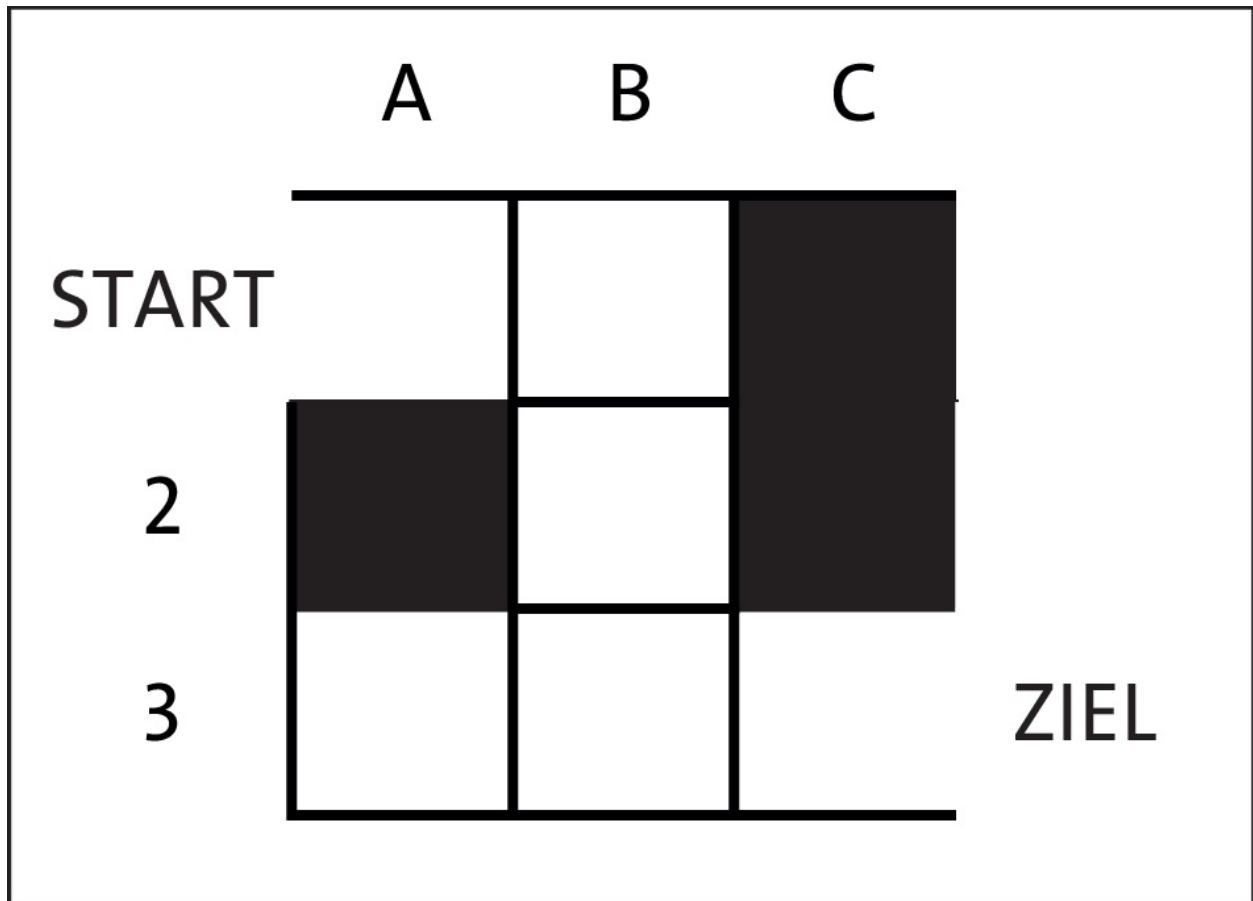




**Abbildung 8.11** *Ein drittes, etwas komplexeres Beispiel für ein neuronales Netz*



**Abbildung 9.1** *Blick in ein Casino mit diversen Spielautomaten*



**Abbildung 9.2** Ein Labyrinth, das aus  $3 \times 3$  Feldern besteht und ein Start- und ein Zielfeld aufweist

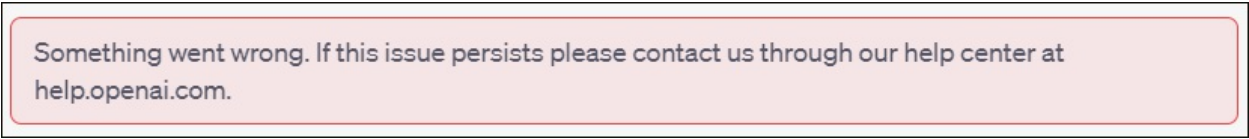
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19				A1	B1	B2	B3	A3	C3
A1	x		x																				A1		8				
B1		x		x																			B1	9		7			
B2					x																		B2				6		
B3						x		x		x													B3					3	1
A3							x		x														A3				2		
C3											x																		

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19				A1	B1	B2	B3	A3	C3
A1	x				x																		A1		10				
B1		x		x		x		x															B1	11		7			
B2			x				x		x		x												B2		8		4		
B3										x		x		x									B3				5		3 1
A3													x										A3				2		
C3															x														

**Abbildung 9.3** *Zwei Versuchsläufe, um aus unserem Labyrinth zu entkommen*



**Abbildung 9.4** *Ein besonders ungewöhnliches Labyrinth mit komplexer Wegführung*

A screenshot of a ChatGPT error message. The message is displayed in a light red rectangular box with rounded corners and a thin black border. The text inside the box is in a dark gray, sans-serif font and reads: "Something went wrong. If this issue persists please contact us through our help center at help.openai.com."

Something went wrong. If this issue persists please contact us through our help center at [help.openai.com](https://help.openai.com).

**Abbildung A.1**    *Auf meinen Prompt hin gibt ChatGPT eine Fehlermeldung aus: »Etwas ist schiefgelaufen. Wenn dieses Problem weiterhin besteht, kontaktieren Sie uns bitte über unser Hilfe-Center auf [help.openai.com](https://help.openai.com).«*



