Tom Alby

Data Science in der Praxis

Eine verständliche Einführung in alle wichtigen Verfahren

Für Praktiker mit und ohne Mathematikkenntnisse

+ Erfolgsfaktoren und Best Practices für alle Projektphasen
+ Inkl. Einführung in die Data-Science-Programmiersprache R
+ Von der Anforderungsanalyse über die Datenakquise bis zur Visualisierung

Alle Beispielprojekte zum Download

Rheinwerk Computing

Liebe Leserin, lieber Leser,

Data Science ist ein recht junges Fachgebiet. Zurzeit werden Studiengänge eingerichtet und Curricula geschrieben, und neue Methoden der Datenanalyse verbreiten sich in der Industrie.

Die Verfahren basieren auf der Mathematik, insbesondere der Statistik, die nicht nur sehr viel älter ist als Data Science, sondern auch nicht immer leicht zugänglich. Brauchen Sie deshalb eine Mathematikvorlesung, bevor Sie mit Data Science loslegen?

Nein, denn Tom Alby hat einen Zugang entwickelt, der mit wenigen, ausgewählten mathematischen Erklärungen auskommt. In vielen Lehrveranstaltungen mit Studierenden erprobt, liegt er auch diesem Buch zugrunde.

Es ist wie bei einer Mikrowelle: Dahinter steckt Physik, die Sie nicht zu kennen brauchen, um das Gerät zu benutzen. Sie sollten aber z. B. wissen, dass kein Metall hineindarf oder dass eine doppelte Menge Wasser sich entsprechend langsamer erwärmt. Auch das ist Physik, aber viel leichter zu erfassen.

So ist auch die Mathematik in diesem Buch: Überschaubar, aber wichtig. Egal, ob Sie sich mit Mathematik nun eher wohlfühlen oder eher schwertun – was Sie brauchen, um die verschiedenen Verfahren richtig auszuwählen und anzuwenden, lernen Sie hier.

Das Buch wurde mit großer Sorgfalt geschrieben, geprüft und produziert. Sollte dennoch einmal etwas nicht so funktionieren wie erwartet, wenden Sie sich an mich. Auch Fragen, Anregungen und konstruktive Kritik sind uns herzlich willkommen.

Viel Erfolg mit der Datenanalyse wünscht Ihnen

Ihre Almut Poll
Lektorat Rheinwerk Computing

almut.poll@rheinwerk-verlag.de
www.rheinwerk-verlag.de
Rheinwerk Verlag · Rheinwerkallee 4 · 53227 Bonn

Kapitel 3
Ablauf eines Data-Science-Projekts

»Sag mir, wie Dein Projekt anfängt, und ich sage Dir, wie es aufhört.«
(Weisheit aus der Projektmanagement-Welt, Urheber unbekannt)

In diesem Kapitel werden Sie alles lernen, was zum Managen eines Data-Science-Projekts notwendig ist. Fast jedes Data-Science-Projekt versucht, ein in der realen Welt vorhandenes Problem zu lösen, und damit dieser Ambition zum Erfolg verholfen wird, haben sich einige Projektphasen und »Best Practices« etabliert. Dies beginnt mit der Anforderungsanalyse, führt über die Akquise und Reinigung der Daten sowie die weichen Faktoren der Auswirkungen eines Modells bis zum Testen und Ausrollen eines Projekts.

3.1 Der allgemeine Ablauf eines Data-Science-Projekts

Auch wenn Data Science noch ein junges Feld ist, haben sich bereits einige Best Practices durchgesetzt. Sie stammen zum Teil aus verwandten Gebieten wie dem Data Mining. Wir schauen uns zunächst Prozesse an, die in der (zum Teil noch jungen) Literatur erwähnt werden, und schauen uns dann in den weiteren Abschnitten grundsätzliche Muster genauer an.

3.1.1 Die CRISP-DM Stages

Das Akronym CRISP-DM steht für *CRoss-Industry Standard Process for Data Mining*, das ab 1996 von einem Konsortium im Rahmen eines EU-Förderprojekts entwickelt wurde.[1] Dieser Prozess umfasst eine granulare Übersicht der einzelnen Schritte, die ich hier ergänze mit Hinweisen, wo Sie diese Schritte in diesem Buch finden):

1. Verständnis des Geschäfts und seiner Abläufe (dieses Kapitel)
   - Bestimmen der Geschäftsziele für dieses Projekt
   - Bewertung der gegenwärtigen Situation

1 Siehe Shearer 2000.

- Festlegen der Ziele für das Data Mining (beziehungsweise des Data-Science-Projekts)
- Erstellen eines Projektplans

2. Verständnis der Daten (siehe Kapitel 5 über die explorative Datenanalyse)
   - erste Daten sammeln
   - Daten beschreiben
   - Daten erforschen
   - Überprüfen der Datenqualität
3. Vorbereitung der Daten (ebenso Kapitel 5)
   - Daten auswählen
   - Daten reinigen
   - Daten transformieren (im Englischen *construct*, hier interpretiere ich das aber als Transformation)
   - Daten integrieren
   - Daten formatieren
4. Modeling (Kapitel 6 bis 9)
   - Modelltechnik auswählen (hier ist übertragen auf Data Science der Algorithmus gemeint, wobei meistens mehrere Algorithmen und Modelle gegeneinander getestet werden)
   - Testentwurf generieren
   - Modell erstellen
   - Modell bewerten
5. Evaluation (dieses Kapitel sowie Kapitel 6 bis 9)
   - Ergebnisse auswerten
   - Prozess überprüfen
   - nächste Schritte bestimmen
6. Deployment (Einsatz des Systems in Produktion, siehe dieses Kapitel sowie Kapitel 10)
   - Planen des Deployments
   - Planen des Monitorings und der Maintenance des Modells
   - Erstellen eines finalen Reports
   - Projekt-Review (zum Beispiel Post Mortem, Lessons learned)

Auch wenn dies aussieht wie ein linearer Prozess, so zeigt Abbildung 3.1, dass die Schritte im CRISP-DM nicht wie in einem Wasserfallmodell abgearbeitet werden, sondern dass Iterationen notwendig sein können. So kann es sein, dass man nach der Evaluation noch mal ganz zurück muss zum Verstehen des Geschäfts, weil sich aus den Daten ganz neue Sachverhalte ergeben haben. Ist das selten? Eher nein, allerdings hängt es ganz klar davon ab, wie viel Zeit man in diese erste Phase gesteckt hat.

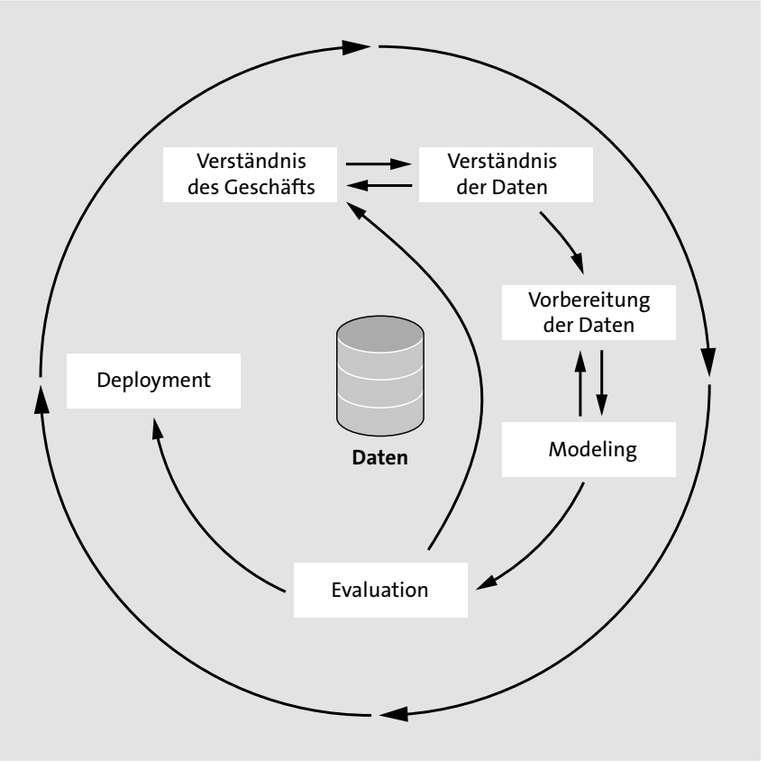

Abbildung 3.1 Das CRISP-DM-Modell (nach dem IBM SPSS Modeler CRISP-DM Guide)

3.1.2 ASUM-DM

ASUM-DM steht für *Analytics Solution Unified Method for Data Mining* und wurde von IBM 2015 als Weiterentwicklung des CRISP-DM von IBM veröffentlicht. Es umfasst fünf Phasen:

1. Analyse
2. Design
3. Konfiguration und Herstellung (Configure & Build)
4. Inbetriebnahme (Deploy)
5. Betrieb und Optimierung (Operate & Optimize)

Auch hier lassen sich die Phasen mehrmals durchlaufen. Parallel läuft ein Projektmanagement-Stream. Eine genaue Beschreibung findet sich unter *https://alby.link/asum*.

3.1.3 Der Ablauf nach Hadley Wickham

Hadley Wickham, Chief Data Scientist bei RStudio (Kapitel 4, »Einführung in R«), Schöpfer des Tidyverse (Kapitel 5, »Explorative Datenanalyse«) sowie mehrfacher Buchautor, hat einen eigenen Ablauf entwickelt, den Sie in Abbildung 3.2 sehen. Sein Prozess fokussiert sich eher auf die Analysetätigkeit des Data Scientists und klammert die Business-Sicht komplett aus.

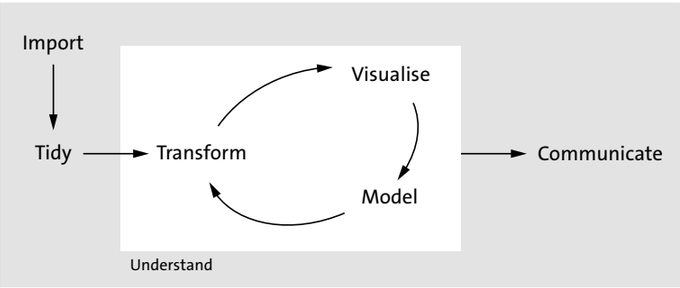

Abbildung 3.2 Hadley Wickhams Ablauf eines Data-Science-Projekts, Abdruck mit seiner freundlichen Genehmigung

Seine etwas andere Sicht bedeutet nicht, dass sein Ansatz schlechter ist, er hat einfach eine andere Perspektive, sozusagen eine andere Brennweite in seinem Objektiv, durch das er einen bestimmten Ausschnitt aus einem Data-Science-Projekt genauer sieht. Diese Sicht ist durch sein R-Package Tidyverse untermauert, das Sie in Kapitel 5 kennenlernen werden.

Wickhams Prozess besteht aus:

- Importieren der Daten
- Säubern der Daten

- Transformation der Daten
- Visualisierung der Daten
- Modellierung
- Kommunikation

Die Schritte Transformation, Visualisierung und Modellierung hat er zusätzlich als Prozess des Verstehens gekennzeichnet. Während der Ansatz des CRISP-DM erst das Verständnis der Daten voraussetzt, geht Wickham davon aus, dass ein genaueres Verständnis erst während der Transformation, Visualisierung und Modellierung entstehen kann. Diese Sichtweise kann ich nur unterstreichen, da sich erst bei genauerem Hinsehen weitere Fragen ergeben. Allerdings ist dies eher die Sichtweise des Data Scientists. Das Businessproblem schließt Wickham komplett aus.

3.1.4 Welcher Ansatz ist für mich der richtige?

Die Antwort auf diese Frage hängt vom Projekt, aber auch vom Unternehmen ab. Wie im vorherigen Abschnitt besprochen, eignet sich der Ansatz von Wickham vor allem für Data Scientists selbst, doch auch ein Data Scientist kann nicht die Augen vor den Bedürfnissen der Organisation verschließen. Der CRISP-DM-Ansatz hat das ganze Unternehmen im Blick sowie die Schritte außerhalb des Gewerks der Data Scientists. Meine Empfehlung ist eine Mischung aus beiden Ansätzen, wobei ich Wickhams Ansatz vor allem für die explorative Datenanalyse und die Modellbildung empfehle, also alles, was Sie hier in den Kapiteln 5 bis 9 finden.

3.2 Business Understanding: Welches Problem soll gelöst werden?

Bevor es an das Entwickeln von Modellen geht, müssen Sie zunächst verstehen, was das Problem ist, das gelöst werden soll, und was die Erwartungen der Anwender sind.

3.2.1 Senior-Management-Unterstützung und Einbeziehung der Fachabteilung

Data-Science-Projekte scheitern meistens aus zwei Gründen:

- Die Anforderungen wurden nicht richtig verstanden, oder das Problem wurde nicht richtig gelöst
- Es fehlte die Unterstützung des Managements und der Fachabteilung.

Für das englische Wort *commitment* gibt es meiner Meinung nach keine gute deutsche Übersetzung, es ist mehr als eine Zusage oder sich verpflichtet zu fühlen. Genau das

wird benötigt von den Stakeholdern, die ein Data-Science-Projekt zum Fallen bringen können, dem Senior Management (Vorstandsebene, Geschäftsführung und -leitung) und der Fachabteilung. Die Fachabteilung kann aus unterschiedlichsten Gründen ihr Interesse verlieren, sei es, weil alles komplizierter ist als gedacht und man auch noch mehr mitarbeiten muss als erwartet (siehe Abschnitt 3.7.5), sei es, weil es latente Störgefühle gibt, da sich etwas ändert (siehe Abschnitt 3.2.3).

Mit fehlender Unterstützung ist allerdings nicht nur das Involviertsein während des Projekts gemeint, sondern auch die Zeit, nachdem eine Data-Science-Anwendung den Anwendern zur Verfügung gestellt wurde. Es können die besten Systeme der Welt erstellt werden, die aber nichts bringen, wenn sie nicht genutzt werden. Auch das muss von Beginn an besprochen werden: Wie kann sichergestellt werden, dass das zu erstellende System auch eine angemessene Nutzung haben wird? Dies geht nur, wenn der erhoffte Nutzen dann auch tatsächlich vorhanden ist. Zu verstehen, worin dieser Nutzen genau besteht, darum geht es im nächsten Abschnitt.

3.2.2 Anforderungen verstehen

Die größte Herausforderung in einem Data-Science-Projekt ist häufig nicht das Trainieren und Optimieren eines Modells, sondern vielmehr, zu verstehen, worum es eigentlich geht und wie tatsächlich ein Mehrwert generiert werden kann (ganz abgesehen davon, die richtigen Daten zu bekommen und sie zu reinigen). Dazu gehört auch, dass man das Geschäftsmodell versteht, in dem die Nutzer arbeiten, wozu auch ein Verständnis der Ziele der jeweiligen Abteilung gehört.

Beispiel: Zu Beginn der Corona-Krise arbeitete ich an einem Data-Science-Projekt für ein Inkassounternehmen. Ich hatte keinerlei Vorwissen über Inkasso, und tatsächlich funktioniert Inkasso ganz anders, als es sich die meisten wahrscheinlich vorstellen. Wer denkt bei Inkasso nicht an ein paar muskelbepackte Männer, die schlagkräftige Argumente hervorbringen, um an das Geld zu kommen? Die Welt des Inkassos ist aber viel komplexer und spannender als erwartet, aber es hat mehrere Wochen gekostet, diese Welt zu verstehen (die Menschen, die Schuldner besuchen, haben übrigens Verhandlungserfahrung und keine Muskelpakete). Es ist notwendig, genau nachzuvollziehen, wie die Nutzer arbeiten und wie das geplante Werkzeug sie unterstützen soll. Dies kann häufig erfordern, vor Ort den Mitarbeitern bei ihrer Arbeit zuzusehen, sie zu beobachten und Fragen zu stellen.

Aber es sind nicht nur wir Data Scientists, die lernen müssen. Die Erfahrung zeigt, dass sich die Nutzerinnen und Nutzer häufig Funktionen wünschen, die sie nachher nicht verwenden, oder dass sie sich Funktionen wünschen, die sie gar nicht nutzen können,

weil sie keinen Sinn ergeben. Es mag nicht intuitiv wirken, aber manchmal sollte etwas komplett anderes getan werden als das, was sich die Nutzerinnen und Nutzer wünschen. Mein Lieblingsbeispiel hierfür ist der erste Apple iMac. Als er 1998 vorgestellt wurde, prophezeiten viele das Ende von Apple, denn dem iMac fehlte ein Diskettenlaufwerk. Zu dieser Zeit arbeitete man noch mit 3,5-Zoll-Disketten, auf die 1,44 Megabyte passten. Hätte Steve Jobs Team die Nutzer gefragt, so hätten sie sich wahrscheinlich größere Diskettenlaufwerke gewünscht (selbstbrennbare CDs gab es schon, aber sie waren noch extrem teuer, sowohl die Rohlinge als auch die CD-Brenner selbst). Stattdessen hatte Apple ein Modem in den Rechner eingebaut, schließlich könne man auch damit Daten austauschen.

Ein Beispiel aus der Data-Science-Welt: Ein Autohersteller wünscht sich auf der Website eine Personalisierung, so, wie es andere Unternehmen auch schon haben. Tatsächlich ist dies schwer zu realisieren, denn für Spotify ist es einfach, Songempfehlungen zu personalisieren, da genügend Daten vorliegen. Bei Besuchern einer Website eines Autoherstellers ist das nicht unbedingt der Fall (auch wenn die Folien von Unternehmensberatern etwas anderes behaupten). Können Daten dennoch helfen, die Website relevanter zu gestalten? Absolut, allerdings nicht so, wie sich die Marketing-Mitarbeiter das vorgestellt haben.

Aber auch Data Scientists können sich schnell in etwas verlieren, was keinen Wert erzeugt (und hiermit meine ich keinen Data Scientist, den ich persönlich kenne, sondern kann das aus eigener Erfahrung berichten). So wie man sich und das Ziel in einer explorativen Datenanalyse verlieren kann, funktioniert es auch bei vielen anderen Data-Science-Tätigkeiten. Einen neuen Algorithmus ausprobieren? Eine neue Entwicklungsumgebung? Selbstverständlich muss auch hierfür Zeit vorhanden sein, aber schnell ist ein System erstellt, das zwar akademisch sehr interessant ist, aber überhaupt nichts zur Lösung der eigentlichen Fragestellung beiträgt.

3.2.3 Widerstände überwinden: Wer hat Angst vor der bösen KI?

Dem Einsatz von KI oder auch nur von Machine-Learning-Modellen wird nicht nur mit Jubelstürmen begegnet. Menschen fürchten um ihre Jobs, wenn eine Maschine in Zukunft Daten automatisiert auswerten und darauf basierend Entscheidungen treffen soll. Hier treten existenzielle Ängste auf, die nicht mit »Corporate Blabla« besänftigt werden können. Medien befeuern diese Befürchtungen noch, indem sie Dystopien skizzieren, in denen selbst hochqualifizierte Mitarbeiter keine Chance mehr gegen die künstliche Intelligenz haben (siehe Abbildung 3.3). Gleichzeitig haben Meldungen wie die, dass Zalando Marketing-Mitarbeiter entlässt und künstliche Intelligenz die Steue-

rung übernimmt, natürlich für Ängste gesorgt.[2] Selbst wenn viele Studien von Unternehmensberatern, wie viele Menschen in Zukunft ihren Job verlieren werden, nur deswegen geschrieben werden, weil damit die Aufmerksamkeit von CEOs gewonnen werden soll, so ist nicht von der Hand zu weisen, dass es Veränderungen geben wird.

Abbildung 3.3 Titelseite der ZEIT Nr. 29 aus dem Jahr 2014

Die emotionalen Widerstände, die dadurch entstehen, sollten auf keinen Fall ignoriert werden. Es ist wichtig, dass die Mitarbeiter auch auf dieser Ebene mitgenommen werden auf der Reise zu einer Automatisierung durch KI. Eher rational orientierte Menschen mögen das als lästig empfinden, aber entfällt dieser Schritt, so ist mit Widerstand zu rechnen, der in Sabotage des neuen Systems enden kann. Und dann hätte man nichts erreicht, außer viel Zeit zu verlieren.

Doch wie macht man das? Das *House of Change* von Claes F. Janssen, basierend auf der Trauerkurve von Elisabeth Kübler-Ross, skizziert die Phasen, die Mitarbeiter durchleben:

- 1. Zimmer, Zufriedenheit: Alles soll so bleiben, wie es ist.
- 2. Zimmer, Ablehnung: Diese Änderung ist schlecht, sie kann nicht funktionieren.

2 Siehe *https://www.heise.de/newsticker/meldung/KI-gesteuertes-Marketing-Zalando-streicht-250-Arbeitsplaetze-3990425.html*.

- 3. Zimmer, Verwirrung: Ich komme nicht zurecht mit dieser Situation; hier kommt auch die rationale und eventuell auch emotionale Einsicht.
- 4. Zimmer, Akzeptanz: Ich lerne etwas Neues, und jetzt verstehe ich erst, warum das gut ist.

Es kann nicht erwartet werden, dass die Mitarbeiter gleich von Zimmer 1 in Zimmer 4 gelangen, sie müssen in der Regel die Zimmer 2 und 3 auch durchlaufen. Dabei können sie unterstützt werden, allerdings werden sie ihre eigene Zeit dafür benötigen. In der Regel ist es nicht der Data Scientist, der diese Transformationsaufgabe leitet, aber er muss sie verstehen. Da er selbst Teil der Erneuerung ist, kann er die Probleme der Mitarbeiter nicht immer adäquat nachvollziehen. Das Management ist hier gefordert, geeignete Maßnahmen zu finden, die die Transformationsbemühungen so gestalten, dass möglichst viele Mitarbeiter mitgenommen werden können bis in das 4. Zimmer.

3.3 Grundsätzliche Ansätze im Machine Learning

In diesem Abschnitt geht es darum, das Grundgerüst für die Aufgaben im Machine Learning zu erwerben. Auch wenn es nun etwas mathematischer wird (ohne Formeln, nun ja, fast), sollten Sie diesen Abschnitt dennoch lesen, da Sie hier alles lernen, was Sie in einem Gespräch mit einem Data Scientist benötigen.

3.3.1 Supervised versus Unsupervised versus Reinforcement Learning

Stellen wir uns eine Kundendatenbank vor, die Informationen wie Alter, Geschlecht, bisherige Käufe und Adresse enthält, die für den Geschäftsbetrieb notwendig sind. Viele Unternehmen behaupten, dass sie viele Daten haben, sie aber nicht nutzen, und meistens hängt das damit zusammen, dass nicht die richtigen Fragen an die Daten gestellt werden. Was könnten Fragen an die Daten in der Kundendatenbank sein?

Eine erste Frage könnte lauten, ob es Gruppen gibt innerhalb der Kundschaft, die so bisher nicht offensichtlich sind. Das Ziel ist, die Kundengruppen besser zu verstehen, aber es gibt keine *Zielvariable* oder *Target*, anhand dessen ein Algorithmus lernen könnte. Dies bezeichnet man als *Unsupervised Learning*. Der Algorithmus versucht, ohne eine Anleitung herauszufinden, welche Gruppen sich bilden lassen. Wir werden uns solche Algorithmen in Kapitel 7, »Clustering«, ansehen.

Eine andere Frage könnte lauten, welche Kundinnen und Kunden eine höhere Wahrscheinlichkeit haben, dass sie zum Beispiel ihren Vertrag kündigen werden. Hier gibt es eine Zielvariable, nämlich Kündigung oder keine Kündigung. Es wird in diesem Fall in

historischen Daten geschaut nach Kund*innen, die gekündigt haben, und Kund*innen, die nicht gekündigt haben, und diese Daten werden entsprechend *gelabelt*, zum Beispiel 1 für Kündigung und 0 für keine Kündigung. Das *Label* enthält dann die Zielvariable. Dieser Ansatz fällt unter das *Supervised Learning* und wird in Kapitel 8, »Klassifikation«, besprochen. Supervision kann mit »Betreuung« übersetzt werden, und das ist auch das Bild, das Sie in Ihrem Kopf haben können für diesen Ansatz: Der Algorithmus wird betreut, das heißt, ihm wird mit den Labels vorgegeben, wie bisherige Fälle ausgesehen haben, damit er das erworbene Wissen auf neue Fälle anwenden kann. Im Fall des *Clusterings* (Kapitel 7) fehlt eine solche Betreuung, und der Algorithmus ist auf sich allein gestellt. Nun ja, nicht ganz, denn etwas tut der Mensch hier schon, indem geeignete Variablen ausgewählt werden.

Neben diesen beiden Machine-Learning-Paradigmen existiert ein drittes, das *Reinforcement Learning*, seltener auch *bestärkendes* oder *verstärkendes Lernen* genannt. Dabei werden beide Welten gewissermaßen kombiniert, denn der Algorithmus entwickelt eine eigene Strategie und erhält Feedback, sozusagen eine Belohnung, so dass der Algorithmus versucht, die Anzahl der Belohnungen zu erhöhen. Häufig wird der Begriff *Agent* in diesem Zusammenhang verwendet, und tatsächlich kann man sich das vorstellen wie ein virtuelles Wesen, das auf Basis von Lob und Tadel seine eigene Strategie entwickelt, um eine Aufgabe zu lösen.

3.3.2 Feature Engineering

Als *Feature Engineering* bezeichnet man das Selektieren und Aufbereiten von *Variablen* beziehungsweise *Features*, die einen hohen Informationsgehalt für die Fragestellung haben und später einem Machine-Learning-Modell zugeführt werden. Beispiel: Wir wollen prognostizieren, ob ein Kredit von einem potenziellen Kreditnehmer zurückgezahlt werden kann. Die folgenden Features stehen zur Verfügung:

- Höhe des Kredits
- bisherige Kredithistorie
- Sternzeichen des Kreditnehmers
- Kontostand

Offensichtlich sind einige dieser Variablen sehr interessant, das Sternzeichen jedoch wird sehr wenig dazu beitragen. Dies ist ein offensichtliches Beispiel, aber in der Realität ist das nicht immer so klar. Wir werden in den nächsten Kapiteln mehrere Datensätze kennenlernen und auch hier die Variablen daraufhin untersuchen, ob sie uns bei der Lösung einer Frage helfen oder nicht.

3.4 Performancemessung

Nachdem die Features und die Parameter eines Modells gewählt wurden, wird das Modell angewendet. Danach wird gemessen, wie gut das Modell funktioniert.

3.4.1 Test- und Trainingsdaten

Zur Überprüfung eines Modells werden für das Training nicht alle Daten verwendet (die sogenannten *Trainingsdaten*), sondern nur ein Teil. Der andere Teil wird zum Testen genutzt (*Testdaten*), und die Ergebnisse des Modells werden mit dem tatsächlichen Ergebnis verglichen. Das Testset wird dann auch als *Holdout Data* bezeichnet. Auf keinen Fall dürfen die Trainingsdaten auch für das Testen verwendet werden. Im Unsupervised Learning werden keine Trainingsdaten benötigt, da wir ja keine Zielvariablen haben, anhand derer der Algorithmus lernen könnte.

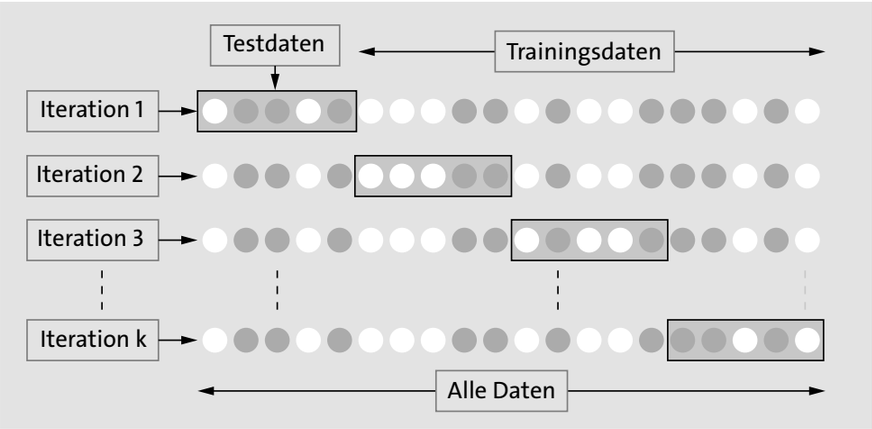

Abbildung 3.4 Ansicht einer k-fold Cross Validation

Eine besondere Form des Testens ist die sogenannte *Cross Validation*, seltener auch *Kreuzvalidierung* genannt. Sie ist in Abbildung 3.4 visualisiert. In der ersten Iteration wird ein Teil der Daten als Trainingsset genutzt, der andere Teil als Testset. Danach werden die Daten neu aufgeteilt, und zwei andere Teilmengen werden als Trainings- und als Testset genutzt. Dies wird mehrmals wiederholt, in der Abbildung steht k für die Zahl der Iterationen. Keine Validierung ist allerdings so gut wie die mit neuen, frischen Daten.

In der Literatur wird mitunter unterschieden zwischen Testdaten und *Validierungsdaten*, allerdings werden die Begriffe auch häufig synonym verwendet, was zu Verwirrung führen kann.[3] Tatsächlich gibt es aber einen guten Grund, zu unterscheiden. Denn angenommen, dass ein Modell zu Beginn nicht so gute Ergebnisse liefert anhand der Testdaten, dann wird anhand dieser Ergebnisse das Modell optimiert (zum Beispiel, indem Parameter justiert werden). Das bedeutet dann aber auch, dass diese Optimierung genau für diese Testdaten funktioniert. Daher wird häufig vorgeschlagen, dass die Trainingsdaten noch einmal unterteilt werden in zwei Teile, Trainingsdaten und Validierungsdaten. Das eigentliche Testset aber, das wir zu Beginn abgetrennt hatten, wird aus diesem Prozess herausgehalten bis zum Schluss, wenn ein Modell fertig optimiert wird. Erst dann wird dieses Testset genutzt, um das fertig optimierte Modell zu testen und auszuschließen, dass die Optimierung nur für die Validierungsdaten gut funktioniert.

In diesem Buch werden wir vor allem mit der simplen Form der Validierung arbeiten und nur ein Holdout Set verwenden. Ein Beispiel für die Cross Validation finden Sie auf der Website zum Buch.

3.4.2 Fehler ist nicht gleich Fehler: False Positives und False Negatives

Es ist unwahrscheinlich, dass ein Machine-Learning-Modell fehlerfreie Ergebnisse abliefert. Sie müssen mit Fehlern rechnen. Aber nicht alle Fehler sind gleich schlimm.

In der Statistik wird zwischen dem Fehler 1. Art und dem Fehler 2. Art unterschieden, auch α-*Fehler* (Alpha-Fehler) und β-*Fehler* (Beta-Fehler) genannt. Ein Fehler 1. Art liegt zum Beispiel dann vor, wenn einer nicht schwangeren Frau gesagt wird, dass sie schwanger sei, obwohl sie es nicht ist, und zwar aufgrund eines *False Positive*. Der Schwangerschaftstest ist positiv, obwohl die Frau tatsächlich nicht schwanger ist. Im Gegensatz dazu liegt ein Fehler 2. Art vor, wenn der Schwangerschaftstest negativ ist, die Frau aber schwanger (*False Negative*). Man mag sich in diesem Fall streiten, was schlimmer ist. Besteht ein starker Wunsch nach einem Kind, dann kann die Enttäuschung, dass trotz positivem Test keine Schwangerschaft vorliegt, emotional herausfordernd sein. Verhält sich eine schwangere Frau ungesund, weil sie einen negativen Schwangerschaftstest hatte, bedroht sie die Gesundheit ihres ungeborenen Kindes.

Ähnlich verhält es sich, wenn eine Krankheit diagnostiziert wird, die tatsächlich nicht vorhanden ist, und die den vermeintlich schwer kranken Patienten zu Handlungen veranlasst, die er normalerweise nicht tun würde. Umgekehrt hätte ein schwer kranker Patient eventuell einen anderen Fokus in seinem Leben, wenn er von seiner Krankheit

3 Siehe zum Beispiel Provost und Fawcett 2013.

wüsste. Ein anderes Beispiel ist ein Brandmelder: Geht der Alarm los, obwohl es kein Feuer gibt, dann ist das zwar nervig, aber insgesamt nicht so schlimm. Geht kein Alarm los, obwohl es brennt, dann kann das desaströse Folgen haben.

Auch im Geschäftsleben haben die beiden Fehlerarten Auswirkungen. Geht ein Machine-Learning-System davon aus, dass ein Kreditantrag zu einem Ausfall kommen würde, obwohl der Antragsteller solvent genug ist, den Kredit zu bedienen, dann geht dem Kredithaus ein Geschäft verloren. Wird ein Kredit genehmigt, obwohl der Antragsteller ihn nicht bedienen können wird, so erleidet das Kredithaus einen Schaden. Zwar ist immer mit Schäden zu rechnen, aber diese können nur dadurch in Kauf genommen werden, dass auch genug Geschäft stattfindet. Ist also ein Algorithmus zu konservativ und lehnt zu viele Anträge ab, dann hätte das Kredithaus zwar weniger Schäden, aber zugleich auch weniger Geschäft. Hat das Kredithaus zu großen Risikoappetit, dann macht es zwar mehr Geschäft, wird aber auch mehr Schäden haben. Die Schäden könnten dann die Gewinne durch mehr Geschäft auffressen, denn wenn nur 1 % Zinsen auf einen Kredit gezahlt wird bei einem durchschnittlichen Kreditvolumen von 10000 €, dann werden 100 solcher Kredite benötigt, um einen Kreditausfall in dieser Höhe zu kompensieren!

Es ist die Aufgabe des Projektteams, zusammen mit dem Business zu bestimmen, welcher Fehler akzeptabler ist. Unterschiedliche Algorithmen können auch unterschiedliche Tendenzen in die eine oder andere Richtungen haben, ebenso können auch Parameter in der Modellbildung modifiziert werden, um das Ergebnis zu beeinflussen. Auch kann ein Schwellenwert anders gesetzt werden, zum Beispiel wenn es darum geht, eine Wahrscheinlichkeitsprognose für die Vorhersage zu nutzen. Um bei dem Beispiel des Kreditinstituts zu bleiben, wenn der Schwellenwert für die Ablehnung eines Kreditantrags bei einer Wahrscheinlichkeit für den Ausfall bei 70 % liegt, wie viel Schaden würde das bedeuten und wie viel Gewinn? Hier kommen weitere interessante Fragen hinzu, zum Beispiel welche Rolle die Höhe eines Kredits spielt. Eventuell existiert im Modell bereits ein Einfluss der Kredithöhe auf die Wahrscheinlichkeit, ob der Kredit bedient werden kann.

Aber, es wäre fatal, wenn sich das Machine-Learning-Modell nur dann irrt, wenn es um hohe Summen geht, aber immer bei den niedrigen Krediten richtig liegt. Auch hier bedarf es einer gründlichen Analyse, um die Auswirkungen des Modells zu testen. Jedes Unternehmen wird versuchen wollen, beide Fehler zu minimieren, und eventuell sogar fordern, dass das Modell beide Fehler auf 0 optimieren muss. Dies ist aber in der Regel nicht möglich, da die False-Positive-Rate und die False-Negative-Rate negativ miteinander korrelieren können.

3.4.3 Confusion Matrix

Eine *Confusion Matrix*, seltener *Konfusionsmatrix*, *Entscheidungstabelle* oder *Wahrheitsmatrix* genannt, wird zur Beurteilung eines *binären Klassifikators* verwendet. Ein solcher Klassifikator entscheidet, ob etwas in die eine oder in die andere Klasse gehört: schwanger oder nicht schwanger, kreditwürdig oder nicht kreditwürdig, Spam-E-Mail oder legitime E-Mail. Es existieren auch Klassifikatoren und Konfusionsmatrizen mit mehr als zwei Klassen. Hier ein Beispiel mit zwei Klassen:

	Kredit zurückgezahlt	Kredit ausgefallen
Tatsächlich zurückgezahlt	2839	232
Tatsächlich ausgefallen	23	3212

In diesem Zusammenhang sollen zwei weitere Konzepte und ihre Begriffe eingeführt werden, und zwar *Spezifität* (*Specificity*) und *Sensitivität* (*Sensitivity*), manchmal auch *Empfindlichkeit* genannt. Ein Klassifikator trifft nicht nur falsche Entscheidungen, um die es in dem vorherigen Abschnitt ging, sondern auch korrekte. In der Confusion Matrix werden zusätzlich die True Positives und die True Negatives aufgeführt. Die Sensitivität gibt an, mit welcher Wahrscheinlichkeit ein Klassifikator ein positives Ereignis korrekterweise als positiv klassifiziert, daher wird sie manchmal auch als *True Positive Rate* bezeichnet. Die Spezifizität gibt an, mit welcher Wahrscheinlichkeit ein Klassifikator ein negatives Ereignis korrekt als negativ klassifiziert, daher auch der alternative Begriff *True Negative Rate*. Mit einem Blick auf die Tabelle ist es also möglich, zu sehen, wie gut ein Modell in jeder Klasse abschneidet.

Sensitivität und Spezifizität sind nicht nur im Machine Learning bekannt, sie finden zum Beispiel auch bei medizinischen Tests Verwendung. Wenn ein COVID-Schnelltest eine Sensitivität von 97,56 % und eine Spezifität von >99,9 % erreicht, dann bedeutet dies, dass der Test bei 97,56 % der tatsächlich an COVID erkrankten Patienten dies auch festgestellt hat und bei mehr als 99,9 % ein tatsächlich gesunder Mensch auch als solcher erkannt wurde. In einer Confusion Matrix sieht das so aus:

	Test positiv	Test negativ
Tatsächlich positiv	9976	24
Tatsächlich negativ	5	9995

Offensichtlich würde man sich hier wünschen, dass die Sensitivität besser wäre, die Anzahl der False Negatives also geringer wäre.

Ein »Nachteil« der Confusion Matrix ist, dass sie nicht einen einzigen Wert liefert, mit dem die Performance zwischen verschiedenen Modellen verglichen werden kann. Das ist auch gleichzeitig ihr Vorteil, denn hier können die Stärken und Schwächen eines Modells einfacher nachvollzogen werden, insbesondere wenn es darum geht, dass die Stakeholder eingebunden werden. Dazu mehr im kommenden Abschnitt.

3.4.4 ROC AUC

Das Akronym *ROC* steht für *Receiver Operating Characteristics* und stammt aus einer Zeit, in der Machine Learning noch nicht mal am Anfang stand und der Begriff genau genommen auch nichts damit zu tun hatte. Während des Zweiten Weltkrieges wurde die ROC-Kurve für die Analyse von Radarsignalen verwendet. Erst später hielt das Konzept der Receiver Operating Characteristics im Machine Learning Einzug. Das Akronym *AUC* steht für *Area Under the Curve* und bezeichnet genau das: Die Fläche unterhalb des Receiver-Operator-Characteristics-Graphen.

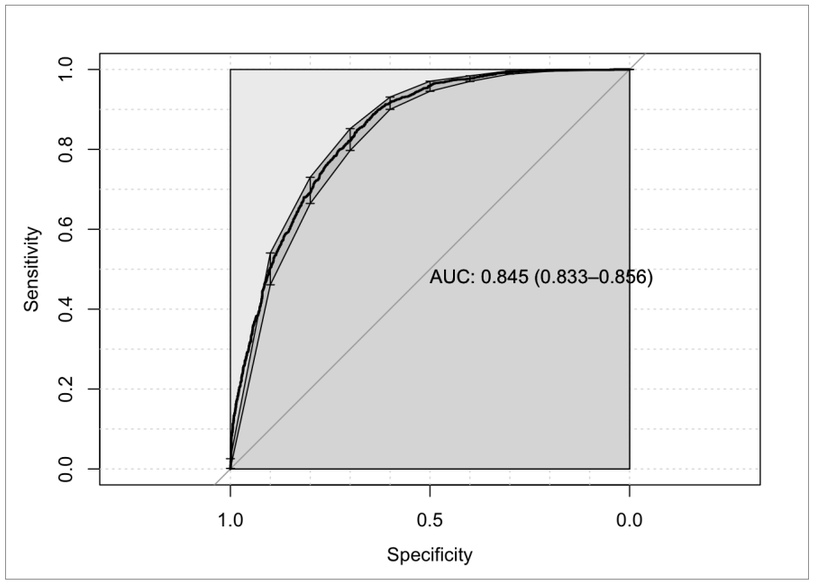

Abbildung 3.5 Ein ROC-AUC-Graph

Ein Beispiel für einen ROC-AUC-Graphen zeigt Abbildung 3.5. Wir sehen zwei Bekannte aus dem vorherigen Abschnitt, nämlich die *Spezifität* und die *Sensitivität*. Gleichzeitig sollte auffallen, dass die Skalen der Achsen sich unterscheiden: Die der Spezifizität beginnt links bei 1 und endet rechts bei 0, wohingegen die der Sensitivität unten links bei 0 beginnt und mit 1 oben endet. Liegt die Spezifizität bei 1, dann liegt die Sensitivität bei 0, was logisch ist, denn wenn alle Fälle als negativ eingestuft werden, dann werden auf jeden Fall alle negativen Fälle erwischt. Es hat nur die Nebenwirkung, dass damit gleichzeitig alle Fälle, die eigentlich positiv sind, nun auch als negativ klassifiziert werden. Umgekehrt, wenn alle Fälle einfach als positiv klassifiziert werden, dann ist die Sensitivität bei 1, denn schließlich hat man alle positiven Fälle korrekt klassifiziert. Hier ist die Nebenwirkung, dass alle Ereignisse, die eigentlich negativ sind, auch als positiv klassifiziert wurden und somit die Spezifizität bei 0 liegt.

Je mehr der Graph sich in die linke obere Ecke orientiert, desto besser, denn schließlich kommt er dann in die Nähe der 1 für beide Dimensionen, Sensitivität und Spezifität. Dadurch wird die Fläche unterhalb des Graphen größer, und genau das ist die Area Under the Curve. Bei einem idealen Modell würde die Linie des Graphen auf der X-Achse bei 1 gerade hochgehen und dann auf der Y-Achse bei 1 gerade parallel zur X-Achse verkaufen.

Die gerade Linie in dem Graphen, die die Punkte (0|1) und (1|0) verbindet, bedeutet eine ROC AUC von 0,5 und steht für einen Klassifikator, der auch einfach zufällig unterscheiden könnte. Er würde in ungefähr der Hälfte der Fälle richtig entscheiden und in der anderen Hälfte falsch.

Der Vorteil der ROC AUC ist, dass man nun eine einzige Zahl hat, mit der die Performance des Modells mit der anderer Modelle verglichen werden kann. Allerdings reicht die Zahl allein dennoch nicht, denn der Graph selbst enthält weitere Informationen, wie Sie in dem Beispiel oben sehen. So kann an dem Verlauf der Kurve sehen, wo das Modell besser und wo es schlechter ist.

3.4.5 Precision Recall Curve

Die Begriffe *Precision* und *Recall* sind vor allem im *Information Retrieval* bekannt, wo sie für die Evaluation zum Beispiel von Suchmaschinen verwendet werden. Ganz vereinfacht gesagt ist der Recall die Menge der relevanten Dokumente, die gefunden wurde, in Bezug auf alle Dokumente im Index, die relevant sind. Precision ist eine Maßzahl für die Genauigkeit des Rankings (sind die relevantesten Dokumente oben?).

Im Machine Learning ist Recall dasselbe wie *Sensitivität*. Precision ist die Ratio der True Positives geteilt durch die Summe der True und Negative Positives. Ein Beispiel für eine solche *Precision-Recall-Kurve* sehen Sie in Abbildung 3.6.

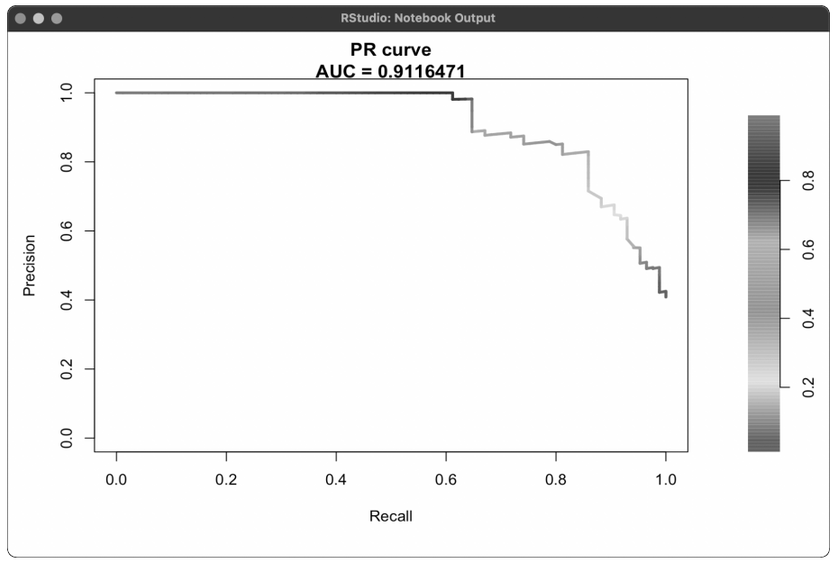

Abbildung 3.6 Eine Precision Recall-Kurve

Hilfreich ist die *Precision Recall Curve*, manchmal auch nur *PR Curve* genannt, vor allem dann, wenn es eine *Class Imbalance* gibt (siehe Kapitel 5, »Explorative Datenanalyse«). Damit ist gemeint, dass von einer Klasse nur wenige Fälle vorhanden sind, zum Beispiel wenige nachgewiesene Versicherungsbetrugsfälle im Vergleich zu vielen legitimen Versicherungsschadensmeldungen. Durch die Kurve kann man das Verhalten für eine Klasse genauer nachvollziehen.

3.4.6 Wirkung des Modells

Ein Modell kann einen enorm guten ROC AUC haben oder in einer anderen Performancemetrik ganz vorn liegen, das heißt aber nicht, dass das Modell im realen Leben auch wirklich gut funktioniert. Da das Training mit historischen Daten stattfindet, kann zum einen *Overfitting* ein Problem sein. Sehr selten ist die Performance eines Modells in der Realität genauso gut wie im Labor. Aber auch sich ändernde Bedingungen, die zu ande-

ren Daten führen, können einen Einfluss haben. Die Corona-Pandemie ist leider ein gutes Beispiel dafür.

Alle diese Data-Science-Metriken sind für die Stakeholder aber nicht spannend; sie interessieren sich nicht für den ROC AUC. Stattdessen gibt es für das Business oder die Organisation andere Metriken, zum Beispiel:

- Umsatz
- Anzahl der Kunden, die neu gewonnen werden können
- Anzahl der Kunden, die ihren Vertrag doch nicht gekündigt haben
- Geschwindigkeit, in der Geld eingeholt werden kann
- Anzahl der Kredite, die nicht mehr ausfallen
- Anzahl der Melanome, die erkannt werden

Die ROC AUC oder welche andere Metrik auch verwendet wird, sollte idealerweise mit den KPIs des Business korrelieren. Je besser das Modell, desto mehr Geschäfts- oder Behandlungserfolg sollte auch messbar sein.

Um die Wirkung eines Modells zu testen, werden üblicherweise *A/B-Tests* beziehungsweise *Split-Tests* durchgeführt. Das bedeutet, dass ein Teil der Kunden oder Nutzer oder Patienten die bisherige Anwendung erfährt, der andere Teil die Ergebnisse des Systems. So kann überprüft werden, ob die Ergebnisse des Modells statistisch signifikant besser sind.

3.4.7 Data Science ROI

Ein Data Scientist kostet Geld, die Infrastruktur für die Durchführung eines Data-Science-Projekts kostet Geld, und alle Einheiten, die das Data-Science-Projekt unterstützen, werden Geld verbrauchen. Und wenn das Ergebnis des Projekts zu einem späteren Zeitpunkt in Produktion geht, so werden auch dann noch Kosten anfallen, für den Betrieb der Maschinen und die weitere Pflege des Modells. Data Science ist keine günstige Angelegenheit.

Ein Data-Science-Projekt darf also nicht allein aus der Brille der Modellgüte gesehen werden, sondern muss in Bezug zu den Kosten gesetzt werden. Dies ist der *Return on Investment*, kurz ROI.

3.5 Kommunikation mit Stakeholdern

Was ist ein *Stakeholder*? Laut der Definition des Project Management Body of Knowledge (PMBOK) ist ein Stakeholder

> *eine Einzelperson, Gruppe oder Organisation, die auf ein Projekt, Programm oder Portfolio einwirken oder von dessen Auswirkungen betroffen sind oder der Ansicht sind [... davon ...] betroffen zu sein oder zu werden.*

Die Erfahrung zeigt, dass sich nie alle Stakeholder ausreichend informiert fühlen. Die goldene Regel ist: Man kann nie genug kommunizieren.

3.5.1 Reporting

Regelmäßiges Reporting ist ein wichtiger Baustein für den Erfolg eines Projekts. Zum einen kann bei einem regelmäßigen Reporting kein Stakeholder hinterher sagen, man habe nichts gewusst, zum andern zwingt es einen auch, sich immer wieder zu überlegen, was man eigentlich an »Business Value« geschaffen hat mit dem Projekt bisher. Die Frequenz hängt vom Projekt ab, ein monatliches Reporting sollte aber das Minimum sein. In einem solchen Reporting ist es wichtig, schon konkrete Zahlen zu liefern. Also nicht »Wir haben gutes Feedback erhalten« (das liest sich für Manager wie »Wir haben bei einer Tasse Kaffee nett geplaudert«), sondern eher »7 der 9 Datenquellen wurden angeschlossen«. Vorsicht vor Prozentzahlen wie »Wir sind zu 80 % fertig«, denn häufig dauern die letzten 20 % genauso lange, wie die ersten 80 % benötigten. Idealerweise enthält ein Report die folgenden Punkte, frei inspiriert durch Scrum:

- Was wurde erreicht (in Zahlen)?
- Was steht an?
- Was sind Blocker?

In Kapitel 5 werde ich auf Visualisierungen eingehen, die sich hervorragend dazu eignen, Stakeholder über den Fortschritt eines Data-Science-Projekts zu informieren, aber auch, um sie in Details einzuführen.

3.5.2 Storytelling

Auch wenn es nicht zu den Persönlichkeitspräferenzen mancher Menschen gehört, Kommunikation ist alles in einem Data-Science-Projekt. Manche bösen Zungen behaupten, dass es für einen Data Scientist wichtiger sei, gute PowerPoint-Folien erstellen zu können, als tolle Analysen und Modelle zu generieren. Das ist natürlich übertrieben,

aber es ist auch nicht von der Hand zu weisen, dass ein bisschen Verkaufen auch zu dem Profil eines Data Scientists gehört.

Storytelling kann unterschiedliche Formen annehmen. So können Muster bekannter Geschichten verwendet werden, was den großen Vorteil hat, dass Menschen gleich besser zuhören, denn sie kennen die Grundform einer solchen Geschichte bereits und wollen unbedingt hören, ob sie auch das bekannte Ende nimmt. Beispiel: Sie hatten große Probleme, ein Feature zu finden, das bei einer Klassifikation hilft. Alle bisherigen Versuche waren mit wenig Erfolg gekrönt. Nun haben sie ein Feature identifiziert, das bisher wenig beachtet wurde, aber das hat einen enormen Unterschied ausgemacht. Das klingt doch eindeutig nach Aschenputtel! Nicht nur können Sie so die bisherigen erfolglosen Aufwände gut verpacken, Sie zeigen gleichzeitig, dass es nicht einfach war, dieses Feature zu finden.

Sie müssen sich aber auch nicht gleich eines Märchens bedienen. So kann eine Story auch auf Basis der folgenden Bestandteile funktionieren:

- Setting the Stage: Was ist das Problem? Wie groß ist es? Warum sollte es gelöst werden?
- Was wurde bereits versucht? Warum war es nicht erfolgreich?
- Was wird jetzt unternommen? Was sind die bisherigen Erfolgszahlen?
- Was kann jetzt noch dazwischenkommen?
- Ab wann kann man mit dem Happy End rechnen?

Berücksichtigen sollten Sie, dass die Empfänger meistens nicht so tief in einem Projekt drinstecken wie Sie selbst. Sie benötigen mehr Kontext (»Was war das noch mal für ein Projekt? Warum machen wir das?«), was für den Data Scientist selbst nervig sein kann (»Hab ich schon mal alles erklärt«). Hier ist Verständnis notwendig, dass das eigene Projekt nun einmal nicht das Zentrum der Welt ist.

Vermeiden Sie Fachbegriffe, die unverständlich sein könnten. Es reicht z. B., von einem Modell und von Ergebnissen zu sprechen. Dass XG Boost ausgereizt wurde und die Werte der Confusion Matrix toll aussehen, können Sie im Normalfall für sich behalten.

3.6 Aus dem Labor in die Welt: Data-Science-Applikationen in Produktion

Ist ein passendes Modell erstellt worden, soll es in der Regel auch in den produktiven Einsatz kommen. Die goldene Regel ist, dass schon zu Beginn an das Ende gedacht wird, um einen Projekterfolg zu sichern.

3.6.1 Von Data Pipelines und Data Lakes

Die Daten für eine Data-Science-Applikation werden nur selten, wenn überhaupt von einem Nutzer in Echtzeit eingegeben. Daten können eher statisch sein oder sich selten ändern, wie zum Beispiel historische Daten aus Bestellvorgängen, es können aber auch absolut »frische Daten« sein, zum Beispiel Daten, die über eine Website oder eine Kundendatenbank hereinfließen. Solche Daten kommen zum Beispiel in einen *Data Lake*, von wo aus sie mittels *ETL*-Prozessen (*Extract, Transform und Load*, meistens aus verschiedenen Systemen) über eine *Data Pipeline* in das Zielsystem gelangen. Das Aufsetzen solcher Architekturen ist in der Regel nicht Teil des Aufgabengebiets eines Data Scientists, aber es muss von vornherein geplant werden.

Ein weiterer Punkt, den Sie schon von Beginn an bedenken müssen, ist, zu prüfen, wie festgestellt werden kann, dass ein Modell aktualisiert werden muss. Dies kann über ein Monitoring geschehen, aber auch über Überprüfungen in definierten Intervallen. Gleichzeitig sollten Sie bereits zu Beginn definieren, welche KPIs genutzt werden sollen, um die Notwendigkeit einer Aktualisierung feststellen zu können.

3.6.2 Integration in andere Systeme

Häufig soll ein ML-System in ein anderes System integriert werden, zum Beispiel über eine *API* (*Application Programming Interface*). Dadurch können Daten in ein Modell einfließen und die Ergebnisse wieder aus dem Modell in ein anderes System kommen. Ältere Systeme verfügen oft nicht über die APIs, die notwendig sind, und müssen deswegen kostspielig angepasst werden. Zu überprüfen, dass die notwendigen APIs vorhanden sind, ist daher Bestandteil der frühen Projektphase. Kann eine API zu einem späteren Zeitpunkt zur Verfügung gestellt werden, so ist kontinuierlich zu überprüfen, dass diese API auch die Erwartungen erfüllt. Nichts ist schlimmer als ein Data-Science-Projekt, das zwar tolle Ergebnisse liefert, dem Anwender aber nichts nutzt, weil es nicht integriert werden kann.

3.7 Die verschiedenen Rollen in einem Data-Science-Projekt

Data Science besteht nicht nur aus dem Erarbeiten von Modellen, sondern erfordert meistens auch Expertise im Bereich Engineering und vor allem Anwendungswissen für den Bereich, in dem ein Data-Science-Projekt durchgeführt werden soll. Kleine Teams können sich aber häufig nicht den Luxus leisten, entsprechende Experten für alle Aufgaben einzustellen, so dass auch Zeit für die weiteren Aufgaben eingeplant und Expertise aufgebaut werden muss.

3.7.1 Data Scientist

Wie aus den verschiedenen Definitionen von Data Science deutlich wird, könnte man einen Data Scientist auch als eierlegende Wollmilchsau bezeichnen. Wahrscheinlich existieren nur wenige Menschen auf diesem Planeten, die all die Anforderungen erfüllen können, die an einen idealen Data Scientist gesetzt werden:

- ein fortgeschrittenes Wissen in Statistik und wie Daten effektiv visualisiert werden
- Beherrschen von mehreren Programmiersprachen, die für Data Science wichtig sind
- Business Knowledge
- sehr gute Kommunikations- und Präsentationsfähigkeiten

Es ist eher wahrscheinlich, dass ein Data Scientist in einigen dieser Bereiche gut ist, aber nicht in allen, und sich die Teammitglieder gegenseitig ergänzen.

Wie wird man eigentlich Data Scientist? Manche Universitäten bieten einzelne Veranstaltungen an (so wie meine an der HAW Hamburg), andere haben sogar ganze Studiengänge eingerichtet. Die meisten Data Scientists sind aber Quereinsteiger, wobei das wahrscheinlich nicht das richtige Wort ist, denn da es den Beruf »Data Scientist« bis vor wenigen Jahren nicht gab, sind die Data Scientists ohne einen gleichnamigen Studiengang eher diejenigen, die durch ihr Wissen und ihre Expertise aus anderen Domänen das Berufsbild eines Data Scientists geprägt haben. Eine Ausbildung in Statistik, Informatik, Mathematik oder Physik ist auf jeden Fall sinnvoll, aber auch andere Studiengänge können dazu befähigen, einen Zugang zu Data Science zu finden. Ganz ohne Mathe wird es aber nicht funktionieren.

Was hat Tom eigentlich studiert?

Ich habe 2000 ein Studium in den Fächern Anglistik, Germanistik und Informatik mit dem Schwerpunkt Computerlinguistik abgeschlossen. Hier beschäftigt man sich mit großen Datenmengen, Natural Language Processing und zum Teil auch schon Klassifikation mithilfe von Machine Learning, zum Beispiel für Suchmaschinen. Seitdem habe ich immer wieder neue Bereiche kennengelernt, sei es Online-Marketing oder Data Mining in CRMs. Heute beschäftige ich mich unter anderem mit Modellen zur Prognose der Erfolgswahrscheinlichkeit von Inkassofällen. Mein Beispiel zeigt, dass ein Studium nur der Anfang eines Wegs ist, der sich kontinuierlich verändert und dazu führt, dass man stets neu dazulernen muss.

Was ein Data Scientist genau können muss, hängt aber auch von der Umgebung ab, in der sie oder er arbeitet. Insbesondere kleine Teams haben nicht genug Ressourcen, um einen Data Engineer oder einen Data Science Architect in Vollzeit zu beschäftigen. In

solchen Fällen muss dies auch vom Data Scientist übernommen werden. Ganz abgesehen davon sind nicht alle Methoden im Data-Science-Bereich bereits so ausgereift, dass sie ausreichend formalisiert an andere Bereiche übergeben werden können, wie Provost und Fawcett (2013) es formulieren (eigene Übersetzung):

> *Wir müssen darauf hinweisen, dass die Data Science, ebenso wie die Informatik, ein junges Gebiet ist. Die besonderen Belange der Datenwissenschaft sind relativ neu, und die allgemeinen Grundsätze beginnen sich gerade erst herauszubilden. Der Zustand des Data-Science-Gebiets kann mit dem der Chemie in der Mitte des 19. Jahrhunderts verglichen werden, als Theorien und allgemeine Grundsätze formuliert wurden und das Gebiet weitgehend experimentell war. Jeder gute Chemiker musste ein kompetenter Labortechniker sein. In ähnlicher Weise ist es schwer vorstellbar, dass es einen Data Scientist gibt, der nicht mit bestimmten Softwaretools vertraut ist.*

Auch wenn diese Aussage schon vor einigen Jahren getroffen wurde, so ist sie heute immer noch zutreffend, wenngleich sich schon einige Prinzipien entwickelt haben.

3.7.2 Data Engineer

Das Berufsfeld des *Data Engineers* ist noch relativ jung und ergibt sich aus dem Zitat aus dem vorherigen Abschnitt. So wie ein Chemikerin in den frühen Tagen auch eine gute Labortechnikerin sein musste, sind die Data Engineers im übertragenen Sinne die Labortechniker in der Data Science. Sie stellen Data Pipelines zur Verfügung, zum Beispiel damit Daten aus einem Data Warehouse für ein Modell bereitgestellt werden können. Diese Daten können eventuell auch unterschiedliche Strukturen haben und werden dann vom Data Engineer in ein einheitliches Format gebracht. Ebenso gehört es zu den Aufgaben eines Data Engineers, die erstellten Anwendungen zu überwachen und bei Ausfällen wieder zum Laufen zu bringen.

3.7.3 Data Science Architect

Ebenso wie der Data Engineer ist der *Data Science Architect* ein neues Berufsbild. Er entwirft Architekturen auf Cloud-Services wie AWS, Microsoft Azure oder der Google Cloud Platform, hat aber gleichzeitig die Businessanforderungen im Blick in Hinsicht auf Kosten und zeitliche Einschränkungen.

3.7.4 Business Intelligence Analyst

Ein *Business Intelligence Analyst* wird nicht selten mit einem Data Scientist verwechselt, da sie oder er bereits über erweiterte Fähigkeiten verfügt, die einen kompetenten

Umgang mit einem *Data Warehouse* ermöglichen. Auch ein Business Intelligence Analyst kann und soll Erkenntnisse aus Daten ziehen, um Verbesserungsmöglichkeiten innerhalb einer Firma aufzuzeigen, allerdings in der Regel nicht mit Data-Science-Modellen. Auch wenn die Grenzen zum Data Scientist fließend sind, so sind Business Intelligence Analysts näher am Business dran, präsentieren häufiger ihre Erkenntnisse und nutzen eher Tableau und andere BI-Tools zur Visualisierung. Sie programmieren in der Regel weniger als Data Scientists, und Prognosemodelle liegen eher beim Data Scientist als beim Business Intelligence Analyst. Aber dies kann von Firma zu Firma unterschiedlich sein.

3.7.5 Der Subject Matter Expert

In der Regel ist es notwendig, dass sich der Data Scientist in die Zusammenhänge einer Aufgabe einarbeitet. Dabei hilft der *Subject Matter Expert*, ganz trivial auch Fachexperte, der dem Data Scientist alle notwendigen Informationen zur Verfügung stellt. Idealerweise sitzt der Data Scientist sogar neben den Mitarbeitern, die an Aufgaben arbeiten, die mit dem zu erstellenden System zu tun haben. Gleichzeitig kann der Data Scientist den Subject Matter Expert früh einbeziehen, um Prototypen zu zeigen und Feedback einzuholen, ob die Zusammenhänge auch richtig verstanden wurden. PowerPoint-Folien sind das eine, existierende Software das andere.

Das ist nicht immer gern gesehen. Meistens haben Kollegen schon genug Arbeit auf dem Tisch, und sie sind es auch nicht unbedingt gewohnt, stark in die Entwicklung einbezogen zu werden. So kann es sein, dass die Erwartung vorherrscht, dass Anforderungen einfach über den Zaun geworfen werden und das Entwicklungsteam dann später mit dem fertigen Ergebnis zurückkommt. Ganz abgesehen davon, dass diese Art der Softwareentwicklung für viele Bereiche überholt ist, wird sie dem Endergebnis nicht zum Vorteil gereichen. Stattdessen sollte gleich von Beginn an ein Commitment eingeholt werden, dass Subject Matter Experts ausreichend einbezogen werden können.

3.7.6 Projektmanagement

Auch wenn manche Data Scientists nicht an die Notwendigkeit von Projektmanagement glauben oder sogar der Überzeugung sind, dass Projektmanagement für Data Science nicht möglich sei, so ist dies eher dem fehlenden Wissen über Projektmanagement zuzuschreiben, so dass die Möglichkeiten hier verkannt werden. In manchen Data-Science-Büchern wird Projektmanagement gar nicht erst erwähnt, so zum Beispiel bei Papp et al. 2019.

Ein häufig gehörter Einwand ist, dass man bei Data-Science-Projekten nicht absehen könne, wie lange etwas dauert, denn es sei ja alles neu, was man erarbeitet. Tatsächlich aber geht es genau darum bei Projekten, wie die 6. Ausgabe des Project Management Body Of Knowledge Guide (PMBOK) des Project Management Institutes (PMI) aus dem Jahr 2017 es definiert:

> *Ein Projekt ist ein zeitlich begrenztes Vorhaben mit dem Ziel, ein einmaliges Produkt, eine einmalige Dienstleistung oder ein einmaliges Ergebnis zu schaffen.*

So wird im PMBOK auch die Entwicklung eines neuen pharmazeutischen Präparats als Beispiel für ein Projekt genannt, und die Vergangenheit zeigt, dass nicht jedes dieser Projekte von Erfolg gekrönt ist.

Der Projektmanager oder die Projektmanagerin hat mehrere Aufgaben innerhalb eines Projekts. In der klassischen Sicht des PMBOK ist sie oder er für »die Erreichung der Zielvorgaben des Projekts verantwortlich«.

- Der Projektmanager oder die Projektmanagerin muss nicht der direkte Vorgesetzte der Teammitglieder sein, ist aber dafür verantwortlich, was das Team produziert.
- Zu den Aufgaben des Projektmanagements gehört die Erstellung und Überwachung der Einhaltung der Zeit- und Budgetpläne.
- Umfangsmanagement
- Vom Projektmanager oder der Projektmanagerin wird die Kommunikation mit dem Sponsor des Projekts sowie mit anderen Stakeholdern und den Teammitgliedern geleitet. Vor allem ist die Koordination verschiedener und manchmal auch sich widersprechenden Ziele der Stakeholder zu nennen, die ein Projektmanager oder die Projektmanagerin ins Gleichgewicht bringen muss.
- Der Projektmanager oder die Projektmanagerin muss selbst kein Data Scientist sein. Im PMBOK wird die Rolle mit dem Dirigieren eines Orchesters verglichen, wofür man nicht jedes Instrument beherrschen muss, aber ausreichend Wissen, Verständnis und Erfahrung in der Materie braucht.
- Häufig konkurrieren Projekte um Ressourcen. Die Projektmanagerin oder der Projektmanager muss dabei sicherstellen, dass das Projekt, für das sie oder er verantwortlich ist, ausreichend Ressourcen zur Verfügung hat.
- Festlegen und Überprüfen des Einhaltens von Qualitätskriterien
- Procurement, häufig auch einfach »Einkauf« genannt

Auch wenn die Rolle des Projektmanagers oder der Projektmanagerin manchmal als »alles, was die anderen nicht wollen oder können« missverstanden wird, so ist gutes Projektmanagement ein wesentlicher Faktor für den Erfolg eines Data-Science-Projekts.

Oder, um es mit einem weiteren Sprichwort aus dem Projektmanagement zu sagen: *If you fail to plan, you plan to fail.*

Der klassische Projektmanagement-Prozess, der im PMBOK definiert ist, sieht die folgenden Schritte vor:

- Initiierung
- Planung
- Ausführung
- Überwachung und Steuerung
- Abschluss

Diese Schritte sehen gar nicht so anders aus als die eines Data-Science-Projekts, die ich zu Beginn dieses Kapitels genauer vorgestellt habe:[4]

- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation
- Deployment

Das bedeutet nicht, dass die Schritte ohne Iterationen ablaufen können; so wie das Project Management Institute sich den agilen Methoden geöffnet hat, können auch Data-Science-Projekte nicht nur mit dem Wasserfallmodell geplant werden.

3.7.7 Citizen Data Scientist

Die Rolle des *Citizen Data Scientist* ist eine neue Entwicklung und auch noch nicht gesetzt. Ein Citizen Data Scientist ist (noch) kein richtiger Data Scientist, aber besitzt ein Wissen, das für mehr als einfache Datenanalyse-Tätigkeiten geeignet ist. Tatsächlich aber werden auch nicht immer perfekt ausgebildete Data Scientists für Projekte benötigt. Unter den Tätigkeiten, die in Abschnitt 3.7.1 genannt sind, finden sich genug, die nicht unbedingt von einem Data Scientist durchgeführt werden müssen. Darunter fallen unter anderem:

- Datenbeschaffung
- Datenreinigung

4 Siehe hierzu auch Alby 2017.

- explorative Datenanalysen, wobei es hier vor allem um »Actionable Insights« geht
- das Erstellen von Dashboards und Reports

Citizen Data Scientists bringen einen ganz großen Vorteil für Unternehmen: Anstatt dass bei jedem Projekt, das Daten enthält, ein Data Scientist dabei sein muss, kann erst einmal ein Citizen Data Scientist die Vorarbeit übernehmen und einen Data Scientist dann einbeziehen, wenn komplexere Probleme bearbeitet werden müssen. Häufig sind Citizen Data Scientists auch Subject Matter Experts, so dass kein Informationsverlust beim Transfer von geschäftlichen Anforderungen in das Data-Science-Vokabular entsteht.

3.7.8 Weitere Rollen

Ein *Security Advisor* unterstützt ein Projektteam dabei, die für ein Projekt notwendigen Sicherheitsanforderungen zu erfüllen. Dies kann die Verbindung zu APIs betreffen, die Sicherheit des Netzwerkes, in dem die Data-Science-Server stehen, aber auch die Datenflüsse innerhalb einer Anwendung. Data Leaks gibt es jede Woche neue, und niemand möchte, dass der eigene Arbeitgeber oder die eigene Firma in die Zeitung kommt, weil Daten plötzlich herunterladbar waren. Für kleinere Firmen, die keinen eigenen Security Advisor haben, empfiehlt sich zumindest das Buchen eines Beraters, der diese Aufgabe übernehmen kann.

Vor allem in größeren Unternehmen spielt auch die Rechtsabteilung eine gewichtige Rolle in Data-Science-Projekten, zum Beispiel wenn es darum geht, welche Daten überhaupt verarbeitet werden dürfen. In der Regel sind Rechtsanwälte nicht für Data-Science-Themen geschult und benötigen zusätzliche Informationen, um zu einer Einschätzung zu gelangen. Hinzu kommt, dass zu vielen Themen innerhalb von Data-Science-Themen noch überhaupt keine Rechtsprechung existiert. So fällt es Juristen natürlich schwer, zu einer Einschätzung zu gelangen, denn woran sollen sie sich orientieren? Häufig ist es die Aufgabe der Hausjuristen, das Unternehmen vor Schaden zu bewahren, und so ist es manchmal besser, neue und schwer einzuschätzende Sachverhalte genau zu prüfen oder ihnen zumindest ein Risiko zu bescheinigen. Die bedeutet für beide Seiten ein Frustrationspotenzial, dabei erfüllen beide Seiten lediglich ihre Aufgaben.

In manchen Firmen gibt es einen *Data Steward*, der die Aufgabe hat, die Qualität der Daten und ihrer Quellen sicherzustellen. Er sorgt dafür, dass die strategische Ausrichtung der sogenannten *Data Governance* fachlich korrekt umgesetzt wird. Unter einer Data Governance werden die Prozesse, Regeln und Standards innerhalb einer Organisation verstanden. Ein ganz einfaches Beispiel: Wer darf eigentlich Daten in einem Cus-

tomer-Relationship-Management-System ändern, wenn Daten offensichtlich falsch sind? Die Kundenbetreuer? Die Kundschaft selbst? Angenommen, dass eine Namensänderung durch Heirat erfolgt ist, muss dafür eine Urkunde vorgelegt werden im Rahmen eines *Know-Your-Customer*-Prozesses? Wer überprüft diese Urkunde? Wie wird das dokumentiert? Hier geht es um einen einzigen Datenpunkt, aber jede Firma hat viele Datenpunkte, die unterschiedlich behandelt werden. In Kapitel 11, »Ethischer Umgang mit Daten und Algorithmen«, werden Sie mehr über Datenschutz lernen, aber sicherlich (hoffentlich!) ist jedem klar, dass bestimmte Daten besonders geschützt werden müssen und zum Beispiel nicht jeder Mitarbeiter darauf Zugriff haben darf.

Die vorherigen Abschnitte haben die Vielfalt der verschiedenen Aufgaben innerhalb der Data-Science-Welt beschrieben, und es ist nicht auszuschließen, dass weitere Tätigkeitsfelder hinzukommen werden.

Kapitel 7
Clustering

Wie können in Daten Muster identifiziert werden, aus denen sich Gruppen ableiten lassen, zum Beispiel Kundensegmente? Die Antwort darauf bieten Clustering-Verfahren.

Während wir bei der Klassifikation schon wissen, wie Spam aussieht, und diesen Spam für das Training unseres Klassifikators verwenden können, suchen wir beim *Clustering* nach neuen Mustern, die einen Datensatz in Gruppen unterteilen können.

7.1 Hierarchisches Clustering

Hierarchisches Clustering ist ein Ansatz des *Unsupervised Machine Learnings*. Der Algorithmus basiert auf der Berechnung der Distanzen zwischen allen Datenpunkt in Bezug auf ihre Merkmalsausprägungen und den daraus entstehenden Gruppen, die *Cluster* genannt werden.

7.1.1 Einführung in die Vorgehensweise

Im vorherigen Kapitel zu Prognosen hatten wir zum ersten Mal darüber gesprochen, wie uns Distanzen helfen, die ideale Gerade durch unsere Punkte zu finden. Auch das hierarchische Clustering arbeitet mit Distanzen, wenn auch in etwas anderer Form. Während es bei der Regression darum geht, vorherzusagen, wie ein Wert y aussieht, wenn x einen bestimmten Wert annimmt, geht es beim Clustering darum, überhaupt erst einmal Muster in den Daten zu finden, die dazu führen, dass wir die Datenpunkte sinnvoll gruppieren können. Dies möchte ich an einem einfachen Beispiel veranschaulichen.

Stellen Sie sich die Situation an einer Schule vor: Es gibt ca. 800 Schülerinnen und Schüler, die zwischen 10 und 20 Jahren alt und zwischen 1,30 Meter und 2 Meter groß sind. Dazu gibt es ca. 100 Lehrkräfte, die zwischen 30 und 65 Jahre alt und zwischen 1,60 Meter und 2 Meter groß sind. Wir wissen ja schon, dass wir eigentlich zwei Gruppen haben, Lehrkräfte einerseits und Schülerinnen und Schüler andererseits – nennen wir sie

»Lehrkräfte« und »Schulkinder«, auch wenn manche der »Schulkinder« junge Erwachsene sein mögen. Wie würde ein Algorithmus vorgehen, der das Konzept der beiden Gruppen noch nicht kennt? Bei dem Alter hätten wir zwei klar voneinander getrennte Gruppen, bei der Größe ist das allerdings etwas komplizierter. Ein »Schulkind« kann sehr viel größer sein als eine Lehrkraft, aber nicht älter.

Um den ersten Schritt zu vereinfachen, fokussieren wir uns zunächst auf das Alter. Denn um Cluster identifizieren zu können, also welche Elemente sich so ähnlich sind, dass sie zusammen in einem Cluster gruppiert werden können, müssen die Elemente zunächst miteinander verglichen werden, jedes Element mit jedem anderen. Da wir hier numerische Werte haben, ist das einfach. Eine Person, die 60 Jahre alt ist, ist weiter entfernt von einer Person mit 15 Jahren als eine Person, die 40 Jahre alt ist. Da wir uns aber schlecht 900 Menschen vorstellen können und ihre Distanzen zueinander, vereinfachen wir das Beispiel noch weiter und nehmen eine ganz kleine Schule mit 20 Schulkindern und 5 Lehrkräften, und auch hier konzentrieren wir uns auf das Alter.

Zunächst generieren wir zufällige Zahlen für das Alter beider Gruppen:

```
> ages_pupils <- sample(10:20, 20, replace = TRUE)
> ages_teachers <- sample(30:65, 5, replace = TRUE)
> ages <- c(ages_pupils, ages_teachers)
> ages
 [1] 14 11 11 18 17 13 15 14 12 19 12 19 19 13 16 20 11 11 18 13 42
37 42 41 39
>
```

Wir sehen, dass das erste Schulkind (14 Jahre) einen Abstand von 3 Jahren zum zweiten Schulkind (11 Jahre) hat, ebenso zum dritten, und dann vier zum nächsten Schulkind (18 Jahre). Der Abstand wird also einfach durch eine Subtraktion des zweiten vom ersten Wert ermittelt. Schauen wir uns die Daten in einem Histogramm an:

```
hist(ages, breaks=20)
```

Das Histogramm sehen Sie in Abbildung 7.1, ganz klar haben wir hier zwei Gruppen, die durch die Altersabstände voneinander getrennt sind.

Im nächsten Schritt müssten wir also den Abstand von jedem Schulkind und jeder Lehrkraft zueinander ausrechnen. Netterweise existiert bereits eine Funktion, mit der wir die Distanzen zwischen den einzelnen Personen automatisch ausrechnen lassen können, so dass wir das nicht per Hand tun müssen.

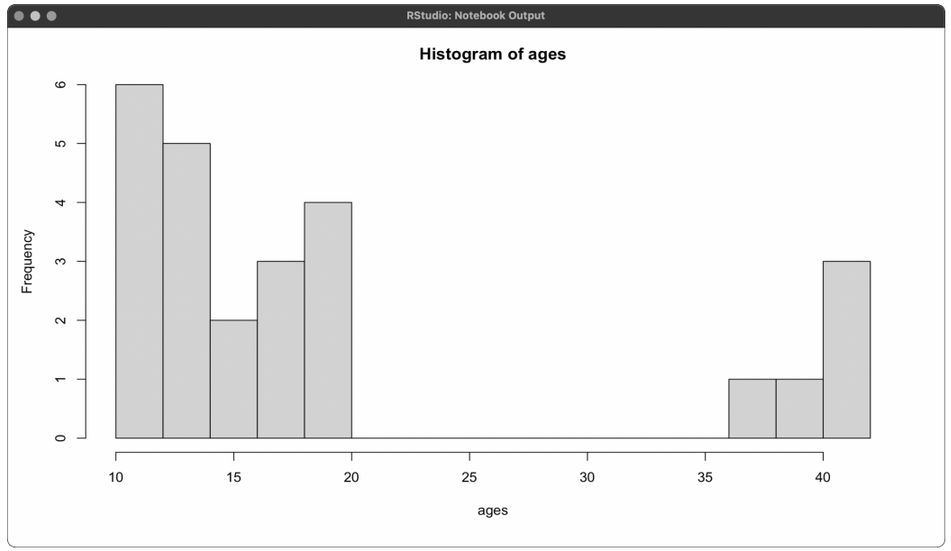

Abbildung 7.1 Histogramm der Altersdaten

Das Ergebnis nennt sich *Distanzmatrix* (*distance matrix* im Englischen, manchmal auch *dissimilarity matrix*):

```
> ages.dist <- dist(ages)
> ages.dist
    1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
2   3
3   3  0
4   4  7  7
5   3  6  6  1
6   1  2  2  5  4
7   1  4  4  3  2  2
8   0  3  3  4  3  1  1
9   2  1  1  6  5  1  3  2
10  5  8  8  1  2  6  4  5  7
11  2  1  1  6  5  1  3  2  0  7
12  5  8  8  1  2  6  4  5  7  0  7
13  5  8  8  1  2  6  4  5  7  0  7  0
14  1  2  2  5  4  0  2  1  1  6  1  6  6
15  2  5  5  2  1  3  1  2  4  3  4  3  3  3
16  6  9  9  2  3  7  5  6  8  1  8  1  1  7  4
```

```
17  3  0  0  7  6  2  4  3  1  8  1  8  8  2  5  9
18  3  0  0  7  6  2  4  3  1  8  1  8  8  2  5  9  0
19  4  7  7  0  1  5  3  4  6  1  6  1  1  5  2  2  7  7
20  1  2  2  5  4  0  2  1  1  6  1  6  6  0  3  7  2  2  5
21 28 31 31 24 25 29 27 28 30 23 30 23 23 29 26 22 31 31 24 29
22 23 26 26 19 20 24 22 23 25 18 25 18 18 24 21 17 26 26 19 24  5
23 28 31 31 24 25 29 27 28 30 23 30 23 23 29 26 22 31 31 24 29  0  5
24 27 30 30 23 24 28 26 27 29 22 29 22 22 28 25 21 30 30 23 28  1  4  1
25 25 28 28 21 22 26 24 25 27 20 27 20 20 26 23 19 28 28 21 26  3  2  3  2
>
```

Unsere Beobachtung für die ersten Werte können wir auch hier nachvollziehen. Auch sehen wir in den letzten Reihen, dass der Abstand der Lehrkräfte zu den Schulkindern größer ist als untereinander. Wir plotten ein erstes *Dendrogramm* (siehe Abbildung 7.2), das die Abstände der Cluster zueinander visualisiert.

```
ages.hc <- hclust(ages.dist, method = "complete")
plot(ages.hc)
```

Das Dendrogramm in Abbildung 7.2 zeigt deutlich zwei Cluster, links die Lehrkräfte, rechts die Schulkinder. Die Zahlen stehen für die Position in der Liste, die wir oben generiert haben. Die Lehrkräfte untereinander wie auch die Schulkinder untereinander sind jeweils weiter unterteilt.

Auch das wollen wir uns einmal genauer ansehen. Links im Lehrkräfte-Cluster sehen wir folgende Untercluster:

- ein Untercluster mit zwei weiteren Unterclustern:
- Datenpunkt 24: 41 Jahre alt
- Datenpunkte 21 und 23: beide 42 Jahre alt
- ein Untercluster mit den Datenpunkten 22 und 25, 37 und 39 Jahre alt

Die Datenpunkte 21 und 23 haben keine Distanz zueinander, da die Personen gleich alt sind. Es ergibt also Sinn, sie zusammen zu clustern. Der Datenpunkt 24 hat einen Abstand von 1 zu den Datenpunkten 21 und 23. Es ergibt also auch Sinn, ihn nah an die Punkte 21 und 23 zu platzieren. Die Lehrkräfte im Alter von 37 und 39 haben eine Distanz von 2 untereinander und eine Distanz von 2 zu Datenpunkt 24, aber da Datenpunkt 24 näher an den Datenpunkten 21 und 23 ist, sind die beiden Lehrkräfte weiter von ihm distanziert.

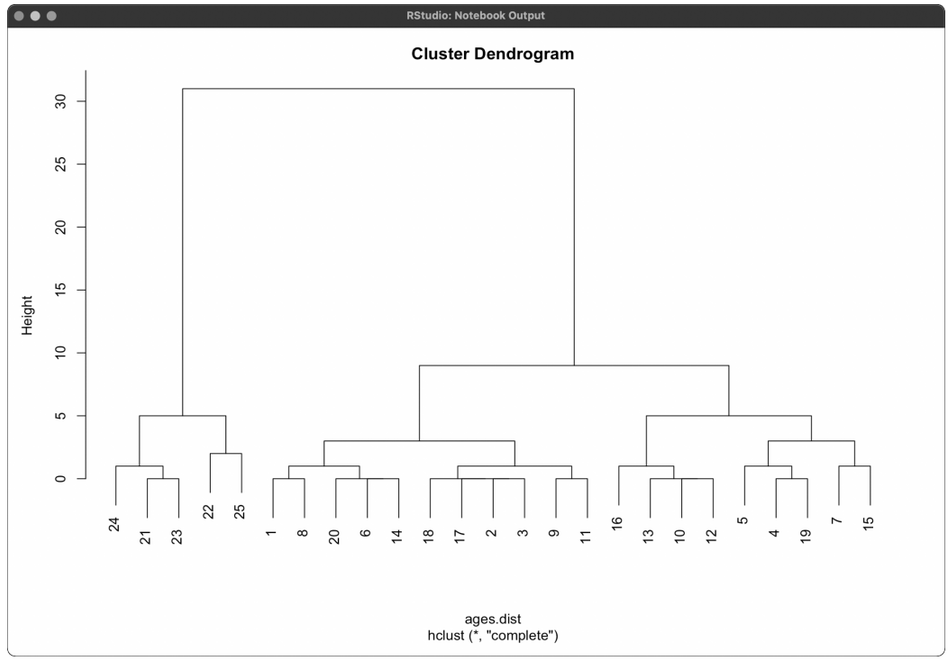

Abbildung 7.2 Unser erstes Dendrogramm

Ein Dendrogramm wird von unten nach oben berechnet. Wir schauen uns erst die einzelnen Datenpunkte und ihre Distanzen untereinander an, bilden daraus die ersten Cluster, und dann werden basierend auf den Distanzen dieser Cluster zueinander die nächsten höheren Cluster gebildet, bis wir ganz oben nur noch ein Cluster haben. Dieses *Bottom-up-Verfahren* wird auch als *agglomerativ* bezeichnet. Es existieren auch *Top-down-Verfahren* (*divisiv*), bei denen von oben nach unten gegangen wird.

Damit sind bereits alle wesentlichen Grundzüge des hierarchischen Clusterings erläutert. Aber wie rechnen wir Distanzen aus, wenn wir mehr als eine Variable haben?

7.1.2 Die euklidische Distanz und ihre Konkurrenten

In unserem ersten Clustering-Versuch haben wir nur einen Teil des Datensatzes genutzt, das Alter. Hier ist die Distanz ganz einfach zu berechnen, die Distanz der Personen zueinander in Bezug auf ihr Alter. Wir haben aber noch eine zweite Variable, die

Größe. Für die Lehrkräfte generieren wir wieder Zufallszahlen, als Einheit verwenden wir Zentimeter:

```
> heights_teachers <- sample(160:200, 5, replace = TRUE)
> heights_teachers
[1] 186 164 164 197 178
```

Daraus erstellen wir einen Data Frame:

```
> teachers <- data.frame(ages_teachers,heights_teachers)
```

Bei den Schulkindern ist das nicht ganz so einfach mit der Größe. Das Alter und die Größe korrelieren, das heißt, ein 11-Jähriger mit einer Größe von 2 Metern wird eher selten auftreten, ebenso eine 18-Jährige mit 1,40 Meter Größe. Ich generiere auch hier Zufallszahlen, passe sie aber manuell an:

```
> heights_pupils <- c(175,147,144,187,180,156,166,164,159,187,153,182,177,158,
160,186,138,134,168,150)
> pupils <- data.frame(ages_pupils,heights_pupils)
```

Die beiden Data Frames verbinde ich miteinander und bilde dann die Distanzmatrix. Vorher muss ich noch ein wenig aufräumen, da die Spaltennamen unterschiedlich sind. Für `rbind()` müssen beide Data Frames dieselben Spaltennamen haben.

```
pupils <- pupils %>%
  rename(age = ages_pupils, height = heights_pupils)
teachers <- teachers %>%
  rename(age = ages_teachers, height = heights_teachers)
> school <- rbind(pupils, teachers)
> dist(school)

          1         2         3         4         5         6         7         8
2  28.160256
3  31.144823  3.000000
4  12.649111 40.607881 43.566042
5   5.830952 33.541020 36.496575  7.071068
6  19.026298  9.219544 12.165525 31.400637 24.331050
7   9.055385 19.416488 22.360680 21.213203 14.142136 10.198039
8  11.000000 17.262677 20.223748 23.345235 16.278821  8.062258 2.236068
9  16.124515 12.041595 15.033296 28.635642 21.587033  3.162278 7.615773 5.385165
```


rbind() und cbind() aus dem Base R

Ich hätte hier auch einen `join`-Befehl aus dem Tidyverse nutzen können, aber ich wollte Ihnen auch einmal einen anderen Befehl aus dem Base R vorstellen. `rbind()` steht für *row bind*, es gibt auch `cbind()` für *column bind*. Manchmal kann man mit diesen Befehlen etwas schneller arbeiten.

Danach nutzen wir wieder `hclust()` und plotten ein Dendrogramm, das Sie in Abbildung 7.3 sehen. Drei Lehrkräfte hängen noch zusammen, aber sind dann mit Schulkindern in einem Cluster verbunden. Ebenso sind zwei andere Lehrrkäfte mit Schulkindern verbunden.

Ich habe oben nur einen Teil der Distanzmatrix wiedergegeben, wir sehen aber schon, dass die Distanzen etwas anders aussehen als zuvor. Wie kommen diese Werte zustande?

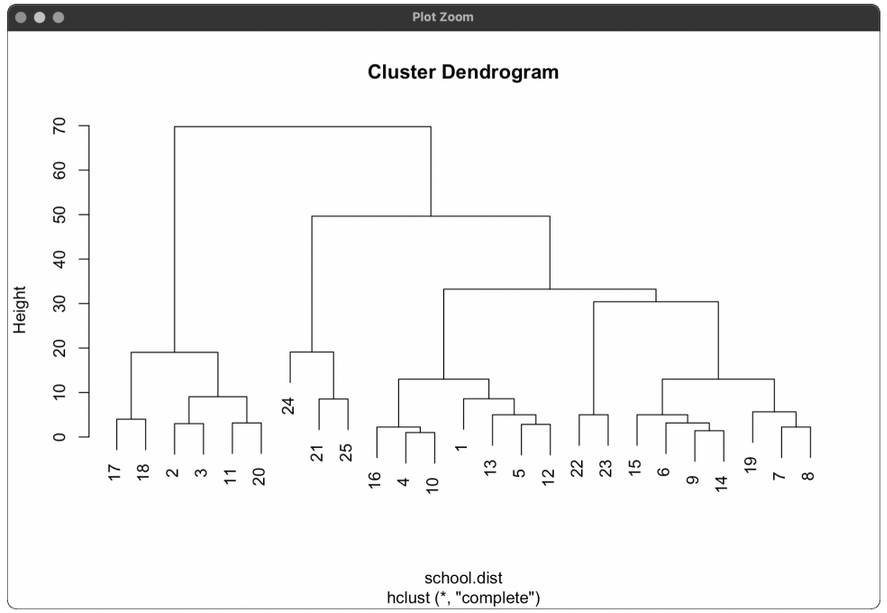

Abbildung 7.3 Das Dendrogramm mit Alter und Größe, allerdings sind die Daten vorher nicht skaliert worden.

Zuvor hatten wir bei einer Variablen einfach einen Wert von dem zweiten Wert subtrahiert. Nun haben wir zwei Variablen, und tatsächlich gehen wir gar nicht so anders vor.

Wir können nun einfach wie bisher jeweils die Datenpunkte voneinander abziehen und das Ergebnis addieren.

Lassen Sie uns das an dem Beispiel der ersten beiden Schulkinder durchgehen. Schulkind 1 ist 14 Jahre alt und 175 cm groß. Schulkind 2 ist 11 Jahre alt und 147 cm groß. Wie wir zuvor schon gesehen hatten, beträgt die Distanz 3. Aber – und normalerweise würde irgendjemand nun im Unterricht aufschreien – manchmal, wenn wir Werte subtrahieren, hätte in der Distanzmatrix eine negative Zahl herauskommen müssen, zum Beispiel bei den Datenpunkten 1 und 4. Wie Sie schon in Kapitel 5, »Explorative Datenanalyse«, gesehen haben, gibt es einen kleinen Trick, um das Vorzeichen loszuwerden, und zwar, indem Sie das Ergebnis der Subtraktion erst quadrieren und später die Wurzel ziehen. Als Formel schreibt man das für zwei Variablen so:

$$\sqrt{(a-b)^2+(c-d)^2}$$

Dabei stehen a und b für die Datenpunkte 1 und 2 der ersten Variablen und c und d für die Datenpunkte 1 und 2 der zweiten Variablen. Setzen wir die Werte aus unserem Beispiel einmal ein:

$$\sqrt{(14-11)^2+(175-147)^2} = \sqrt{9+784} = \sqrt{793} = 28{,}16$$

Wir schauen noch einmal in der berechneten Distanzmatrix oben nach und sehen, dass wir auf das gleiche Ergebnis kommen. Hätten wir drei Variablen, dann könnten wir noch einen Term anhängen und bei einer vierten Variablen noch einen und so weiter.

Was wir hier errechnet haben, nennt sich *euklidische Distanz*. Es ist grob gesagt die »Fluglinie« zwischen zwei Datenpunkten, wenn wir uns diese wie eine Landkarte in einem Koordinatensystem mit zwei Achsen vorstellen. Die euklidische Distanz ist die Standardvorgehensweise, wenn Sie den Befehl `dist()` nutzen. Neben der euklidischen Distanz existieren weitere Optionen, und zwar `maximum`, `manhattan`, `canberra`, `binary` und `minkowski` (anzugeben mit `method = OPTION`). Ich werde nicht alle Optionen im Einzelnen erklären, aber einige können Sie sich sicherlich selbst vorstellen. Die *Manhattan-Distanz* als Beispiel geht sozusagen »um den Block«, hat also nicht die kürzeste Verbindung über die Fluglinie. Als kleine Daumenregel können Sie sich merken, dass Sie für die meisten Fälle die euklidische Distanz nutzen können. Wenn Sie viele Dimensionen haben, wird zur Manhattan-Distanz geraten, dies hat mit dem *Fluch der Dimensionalität* zu tun. Die *Minkowski-Distanz* ist eine verallgemeinerte Form der euklidischen und der Manhattan-Distanz. Mehr dazu finden Sie auf meinem Blog. Für den Anfang reichen Ihnen die euklidische und die Manhattan-Distanz.

Vielleicht ist Ihnen noch eine Unschönheit aufgefallen: Das Alter bewegt sich irgendwo zwischen 10 und 43 Jahren, die Größe aber zwischen 130 und 200 Zentimetern. Mit

anderen Worten: Die Größe kann einen stärkeren Einfluss auf die Distanzberechnung haben. Dies wollen wir noch korrigieren.

Auf einen Blick

Inhalt

Materialien zum Buch

Auf der Webseite zu diesem Buch stehen folgende Materialien für Sie zum Download bereit:

- **alle Beispielprogramme aus dem Buch**

Gehen Sie auf *www.rheinwerk-verlag.de/5341*. Klicken Sie auf den Reiter MATERIALIEN. Sie sehen die herunterladbaren Dateien samt einer Kurzbeschreibung des Dateiinhalts. Klicken Sie auf den Button HERUNTERLADEN, um den Download zu starten. Je nach Größe der Datei (und Ihrer Internetverbindung) kann es einige Zeit dauern, bis der Download abgeschlossen ist.