Angewandtes Machine Learning

Martin Sterchi

2023-10-03

Contents

Chapter 1

Über das Buch

Link zu Kaggle und UC Irvine

ML Beispiele

This is a *sample* book written in **Markdown**. You can use anything that Pandoc's Markdown supports; for example, a math equation $a^2 + b^2 = c^2$.

1.1 Usage

Each **bookdown** chapter is an .Rmd file, and each .Rmd file can contain one (and only one) chapter. A chapter *must* start with a first-level heading: `# A good chapter`, and can contain one (and only one) first-level heading.

Use second-level and higher headings within chapters like: `## A short section` or `### An even shorter section`.

The `index.Rmd` file is required, and is also your first book chapter. It will be the homepage when you render the book.

1.2 Render book

You can render the HTML version of this example book without changing anything:

1. Find the **Build** pane in the RStudio IDE, and
2. Click on **Build Book**, then select your output format, or select "All formats" if you'd like to use multiple formats from the same book source files.

Or build the book from the R console:

```
bookdown::render_book()
```

To render this example to PDF as a `bookdown::pdf_book`, you'll need to install XeLaTeX. You are recommended to install TinyTeX (which includes XeLaTeX): https://yihui.org/tinytex/.

1.3 Preview book

As you work, you may start a local server to live preview this HTML book. This preview will update as you edit the book when you save individual .Rmd files. You can start the server in a work session by using the RStudio add-in "Preview book", or from the R console:

```
bookdown::serve_book()
```

Chapter 2

Mathematik- und Statistik-Grundlagen

In diesem Kapitel repetieren wir die wichtigsten Grundlagen aus der Mathematik und Statistik, die es braucht, um Machine Learning Modelle zu verstehen. Das Thema *Lineare Algebra* wird für die meisten von Ihnen wahrscheinlich Neuland sein.

2.1 Funktionen

Eine Funktion, die wir in der Mathematik typischerweise mit f bezeichnen, ordnet jedem **Argument** x aus dem Definitionsbereich D (engl. *Domain*) **genau einen Wert** y aus dem Wertebereich W (engl. *Codomain*) zu. Oft sind D und W die Menge der reellen Zahlen, also $\mathbb{R}$. Die Menge der reellen Zahlen enthält alle möglichen Zahlen, die Sie sich vorstellen können.[1] Zum Beispiel die Zahlen 3, -4.247, $\sqrt{14}$, $5/8$, etc.

Wie eine Funktion grafisch aussieht, ist aus Panel (a) der Abbildung 2.1) ersichtlich. Hier zeigen wir die Form einer Funktion in einem kartesischen Koordinatensystem. Die Funktionskurve weist jedem Wert x auf der x-Achse genau einen Wert y auf der y-Achse zu. Der wichtigste Teil der oben aufgeführten Definition ist der Teil "genau einen Wert", denn eine Funktion kann einem Element x nicht zwei oder mehr Werte zuweisen, sondern nur genau einen. Genau aus diesem Grund handelt es sich bei Panel (b) in Abbildung 2.1 *nicht* um eine Funktion, da gewissen x-Werten mehrere Werte y zugeordnet werden. *Wichtig*: das heisst aber nicht, dass zwei verschiedenen x-Werten, nennen wir sie x' und x'', derselbe y-Wert zugeordnet werden kann (vgl. Panel (a)).

[1] Einzige Ausnahme sind die komplexen Zahlen.

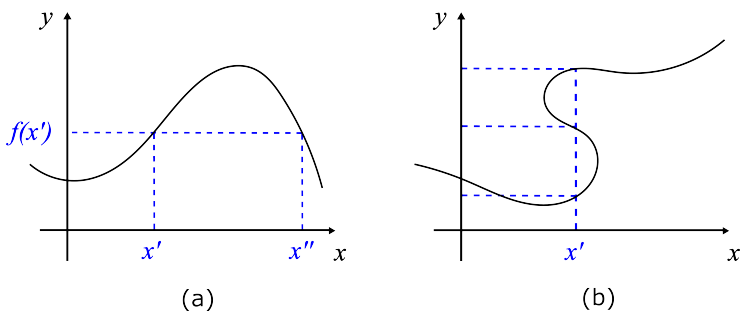

Figure 2.1: (a) Eine Funktion, die jedem x-Wert genau einen y-Wert zuweist. (b) Keine Funktion.

Mathematisch wird diese allgemeine Definition einer Funktion häufig wie folgt beschrieben:

$$f : x \mapsto y$$

Wir haben also eine Funktion f, die jedem Element x genau einen Wert y zuweist. Der Pfeil in obiger mathematischer Schreibweise beschreibt genau dieses Mapping. Wie genau dieses Mapping einem Argument x den entsprechenden y-Wert zuordnet, wird durch die Funktion $f(x)$ beschrieben. In den folgenden Abschnitten schauen wir uns typische Beispiele von Funktionen an, angefangen mit linearen Funktionen. Doch vorher wollen wir uns kurz überlegen, warum Funktionen für das Machine Learning überhaupt wichtig sind. Ein grosser Teil des Machine Learnings, der **Supervised Learning** genannt wird, befasst sich mit dem Problem, wie eine Zielvariable y mithilfe von einem oder mehreren Prädiktoren x vorhergesagt werden kann. Ein Machine Learning Modell ist darum nichts anderes als eine Funktion $y = f(x)$, die basierend auf den Prädiktoren x die Zielvariable y möglichst gut beschreiben kann.[2]

2.1.1 Lineare Funktionen

Nun schauen wir uns an, wie eine **lineare** Funktion aussieht. Eine lineare Funktion kann allgemein wie folgt geschrieben werden:

$$y = f(x) = a \cdot x + b$$

Obige Funktionsgleichung besagt, dass wir den entsprechenden y-Wert kriegen, indem wir den Wert des Arguments x mit a multiplizieren und danach eine Konstante b addieren. a und b sind die **Parameter** dieser Funktion. Die konkreten Zahlenwerte dieser beiden Parameter definieren, wie die Funktion am Schluss genau aussieht.

[2] Zumindest aus einer nicht-probabilistischen Perspektive.

Eine lineare Funktion hat auch eine geometrische Interpretation und zwar entspricht eine lineare Funktion einer Gerade. Das ist auch der Grund, warum wir diese Funktionen **linear** nennen, sie können graphisch durch eine "Linie" dargestellt werden. Der Parameter a ist die Steigung dieser Geraden und der Parameter b entspricht dem Ort, wo die Gerade die y-Achse schneidet (sogenannter y-Achsenabschnitt).

Am besten schauen wir uns ein paar konkrete Beispiele an (Abb. 2.2).

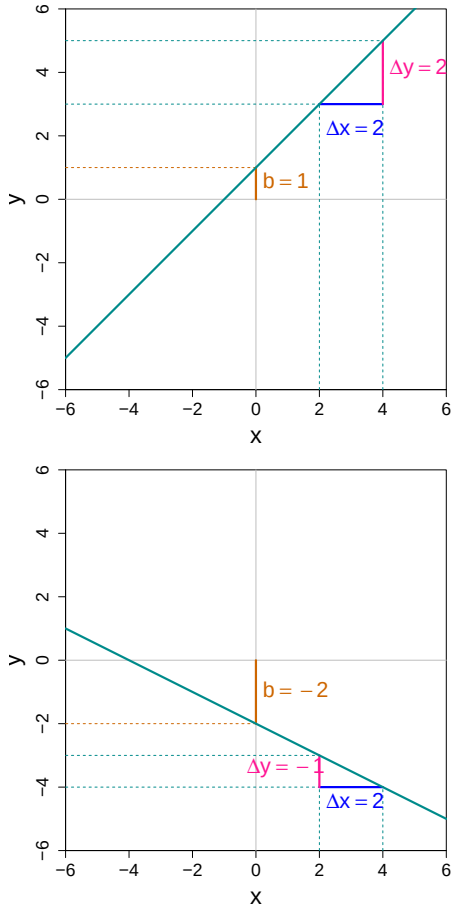

Figure 2.2: Beispiele linearer Funktionen.

Aus der linken Abbildung können wir ablesen, dass die Steigung dieser Geraden $\frac{\Delta y}{\Delta x} = \frac{2}{2} = 1$ ist und dass die Gerade die y-Achse am Ort 1 schneidet. Die entsprechende lineare Funktion kann dementsprechend als $y = x+1$ geschrieben werden.[3]

[3]Wir müssen hier die Steigung 1 nicht explizit schreiben, aber selbstverständlich ist es nicht

Aus der rechten Abbildung können wir ablesen, dass die Steigung $\frac{\Delta y}{\Delta x} = \frac{-1}{2} = -0.5$ ist und dass die Gerade die y-Achse am Ort -2 schneidet. Die entsprechende lineare Funktion kann dementsprechend als $y = -0.5 \cdot x - 2$ geschrieben werden.

Es ist wichtig zu sehen, dass der Effekt einer Veränderung von x (also Δx) auf y überall derselbe ist. Es spielt also keine Rolle, ob wir von $x = -2$ zu $x = -1$ gehen oder von $x = 100$ zu $x = 101$, die entsprechende Veränderung in y (also Δy) wird dieselbe sein. Das muss so sein, denn die Gerade steigt (oder sinkt) mit konstanter Steigung.

Aufgaben

1. Zeichnen Sie die Funktion $y = 2 \cdot x$ in ein Koordinatensystem ein. Warum fehlt der Parameter b?
2. Zeichnen Sie die Funktion $y = -3$ in ein Koordinatensystem ein. Ist das überhaupt eine Funktion nach obiger Definition?

2.1.2 Quadratische Funktionen

Nun wollen wir uns eine etwas interessantere (und flexiblere) Familie von Funktionen anschauen, nämlich **quadratische** Funktionen. Auch hier wollen wir die Funktion erstmal allgemein aufschreiben:

$$y = f(x) = a \cdot x^2 + b \cdot x + c$$

Eine quadratische Funktion hat drei **Parameter**, nämlich a, b und c. Grafisch entspricht die quadratische Funktion einer **Parabel** (vgl. Abb. 2.3). Die Parameter sind hier nicht mehr so einfach grafisch zu interpretieren, aber die vier Beispiele in unten stehender Abbildung geben Anhaltspunkte, was passiert, wenn die Parameterwerte sich ändern.

Aufgaben

1. Sie haben folgende quadratische Gleichung: $y = 2 \cdot x^2 + x - 2$. Berechnen Sie mit der bekannten Lösungsformel $x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$ die Orte auf der x-Achse, wo die Parabel die Achse schneidet (oder einfacher gesagt die Nullstellen).
2. Verwenden Sie folgenden R-Code, um beliebige quadratische Funktionen grafisch darzustellen, indem Sie die Parameterwerte auf der ersten Code-Zeile verändern.

falsch die lineare Funktion als $y = 1 \cdot x + 1$ zu schreiben.

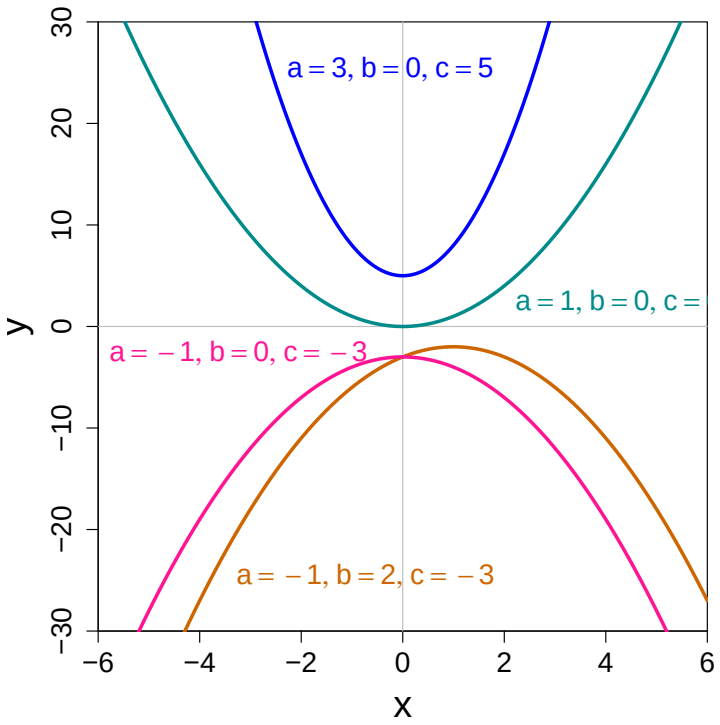

Figure 2.3: Beispiele quadratischer Funktionen.

```r
# Parameter setzen
a <- 2; b <- 0; c <- 1
# Quadratische Funktion
quad <- function(x, a, b, c) {a * x^2 + b * x + c}
# x-Werte
x <- seq(-6, 6, 0.01)
# y-Werte
y <- quad(x, a, b, c)
# Plot
plot(x, y, type = "l", lwd = 2, col = "darkcyan")
```

Sie wundern sich nun vielleicht, könnte man nicht auch eine Funktion antreffen, in der x^3, x^4, etc. vorkommen? Das ist selbstverständlich möglich. In diesem Fall spricht man dann von einem sogenannten **Polynom**. Die höchste Potenz des Arguments x definiert den Grad des Polynoms.

Schauen wir uns doch am besten gleich wieder ein Beispiel an:

$$y = f(x) = 1 \cdot x^4 - 2 \cdot x^3 - 5 \cdot x^2 + 8 \cdot x - 2$$

Die Visualisierung dieser Funktion ist in Abb. 2.4 gegeben. Diese Funktion ist nun bereits enorm flexibel und kann je nach Parameterwerten ganz unterschiedliche Zusammenhänge abbilden.

Aufgaben

1. Eine quadratische Funktion ist ein Polynom welchen Grades?
2. Handelt es sich bei der Funktion $y = 2x^5 + x + 1$ immer noch um ein Polynom? Falls ja, ein Polynom welchen Grades?
3. Handelt es sich bei der Funktion $y = x^{0.5} + 2$ um ein Polynom?

2.1.3 Funktionen mehrerer Argumente

Bisher haben wir nur Funktionen mit **einem Argument** x angeschaut, doch die meisten für das Machine Learning interessanten Funktionen sind Funktionen **mehrerer Argumente**.

Der Einfachheit halber schauen wir uns hier nur mal eine **lineare** Funktion zweier Argumente, nennen wir sie x_1 und x_2, an, denn diese können wir in 3D immer noch visualisieren. Wir betrachten folgende Funktion: $y = f(x_1, x_2) = 1 \cdot x_1 + 0.5 \cdot x_2 + 5$.

Aha! Während eine lineare Funktion eines Arguments grafisch einer Gerade entspricht, sehen wir nun, dass eine lineare Funktion zweier Argumente nichts anderes als eine Ebene darstellt. Wir sehen, dass die Ebene die y-Achse am Punkt 5 schneidet. Etwas schwieriger zu sehen ist die Steigung der Ebene in

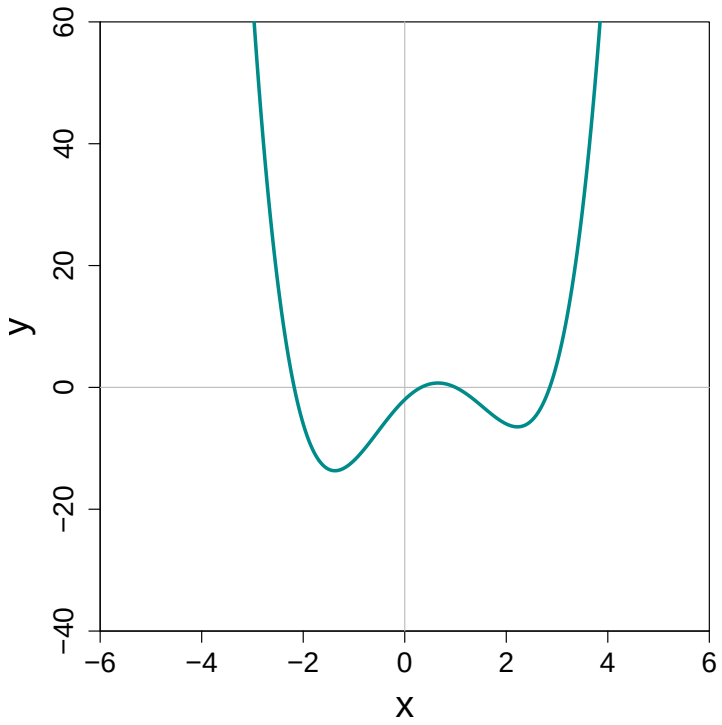

Figure 2.4: Beispiel einer polynomischen Funktion vierten Grades.

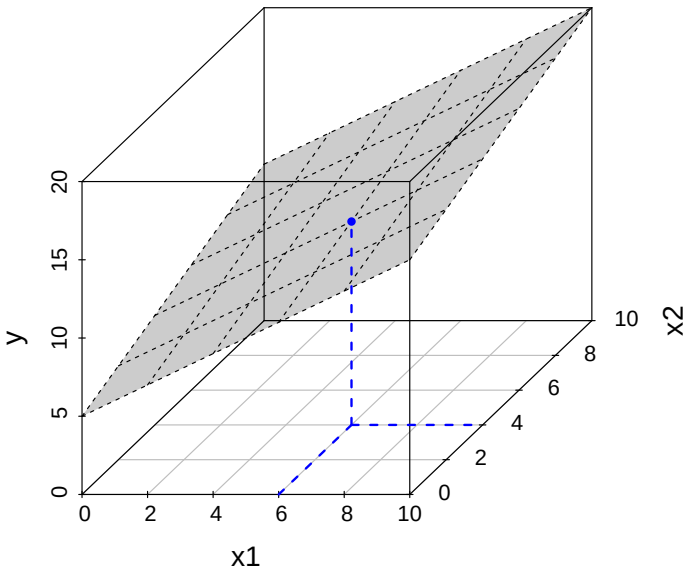

Figure 2.5: Lineare Funktion zweier Argumente (Ebene).

die Richtung der x_1-Achse und in die Richtung der x_2-Achse. Sie können aber vielleicht bereits erraten, dass die (partiellen) Steigungen 1 und 0.5 betragen.

Die Funktion ordnet jeden möglichen Punkt (x_1, x_2) einem Punkt auf der Ebene zu. Wir können zum Beispiel für den in Abb. 2.5 eingezeichneten Punkt $(6, 4)$ den entsprechenden Punkt auf der Ebene ausrechnen:

$$\begin{aligned} y &= 1 \cdot x_1 + 0.5 \cdot x_2 + 5 \\ &= 1 \cdot 6 + 0.5 \cdot 4 + 5 \\ &= 13 \end{aligned}$$

Selbstverständlich könnten wir uns nun auch quadratische Funktionen oder Polynome mehrerer Argumente anschauen, aber darauf verzichten wir vorerst.

2.1.4 Potenzen und Logarithmen

Blabla...

2.2 Integral- und Differentialrechnung

2.3 Lineare Algebra

2.4 Wahrscheinlichkeitsrechnung

2.4.1 Diskrete Zufallsvariablen

Wir werden später sehen, dass im Machine Learning oftmals Dinge als **Zufallsvariablen** modelliert werden. Eine Zufallsvariable X ist eine Variable, für die der konkrete Wert nicht von vornherein klar ist. Wir können mit X zum Beispiel das Resultat eines Münzwurfs modellieren. Die zwei möglichen Resultate sind Kopf und Zahl. Vor dem Münzwurf ist nicht klar, ob Kopf oder Zahl erscheinen wird. Genau darum modellieren wir das Resultat des Münzwurfs als Zufallsvariable.

Es gibt in diesem einfachen Beispiel nur zwei mögliche Resultate (Kopf und Zahl), d.h. die Anzahl möglicher Resultate ist endlich (= nicht unendlich). Darum handelt es sich in diesem Fall um eine **diskrete** Zufallsvariable.

2.5 Verteilungen

Chapter 3

Einführung in das Programmieren mit R

leaRn Materialen

tidymodels

Referenzen auf andere Ressourcen (Hadley et al.)

Chapter 4

Lineare Regression

Chapter 5

Lineare Klassifikation

Chapter 6

Machine Learning Pipeline

Chapter 7

Decision Trees

Chapter 8

Ensembles

Chapter 9

Support Vector Machines

Chapter 10

Artificial Neural Networks

Chapter 11

Convolutional Neural Networks

Chapter 12

Recurrent Neural Networks

Chapter 13

Generative AI