Elemente der Datenanalyse und der Künstlichen Intelligenz

Eine Vorlesung für Lehramts-Studierende im Fach Mathematik

Paul Breiding

Paul Breiding, Universität Osnabrück, pbreiding@uni-osnabrueck.de.

Finanzielle Unterstützung durch die Deutsche Forschungsgemeinschaft
(DFG, German Research Foundation) – Projektnummer 445466444.

Einige Passagen dieses Vorlesungsskripts wurden mit Hilfe eines Large Language Models (LLM) erstellt. Können Sie erraten welche?

Inhaltsverzeichnis

1 Einleitung

1.1 Warum KI und Datenanalyse für Lehrer:innen?

Vielen Dank an Amelie Mühlmeyer, Mara Prehn und Timotheus Tevs für Ihre Beiträge, die bei der Enstehung der folgendenen Einleitung geholfen haben.

Künstliche Intelligenz und Datenanalyse wird für den Lehrer:innenberuf zunehmend wichtig. Wissen in diesem Bereich trägt zur Verbesserung der Unterrichtsqualität und zur Förderung von Datenkompetenz und digitaler Mündigkeit bei.

In einer zunehmend digitalisierten Welt und mit dem Aufkommen von KI gewinnen Daten auch im Bildungsbereich eine immer größere Bedeutung. Wenn Lehrer:innen lernen, diese Daten zu verstehen und zu nutzen, können Sie ihren Unterricht dadurch verbessern. Schon bei der Auswahl und Nutzung von digitalen Lernangeboten spielt Datenanalyse eine wichtige Rolle. Lehrkräfte müssen einschätzen können, welche Apps und Plattformen pädagogisch sinnvoll sind und wie zuverlässig ihre Datenauswertung funktioniert. Nur wenn sie verstehen, wie diese digitalen Tools Daten erheben und interpretieren, können sie deren Ergebnisse kritisch hinterfragen und gezielt im Unterricht einsetzen. Auf diese Weise lassen sich verschiedene digitale Lernmöglichkeiten vergleichen und fundierte Entscheidungen darüber treffen, welche Anwendungen den Lernfortschritt der Schüler:innen tatsächlich unterstützen.

Ein grundlegendes Verständnis von Datenanalyse und Statstik ermöglicht Lehrkräften, Lernprozesse gezielt zu beobachten, Daten selbst zu erheben und erhobene Daten richtig zu interpretieren. Die Arbeit mit Ergebnissen digitaler Lernangebote oder auch Lernstandserhebungen ermöglicht es, den Unterricht evidenzbasiert zu gestalten. Beispielsweise zeigen Auswertungen der Vergleichsarbeiten in der Grundschule (VERA) [17] nicht nur Rohwerte, wie die Anzahl richtig gelöster

Aufgaben, sondern ordnen die Ergebnisse auch in Kompetenzstufen ein und lassen so Vergleiche mit Klassen oder Landeswerten zu. Um aus diesen Daten sinnvolle Schlüsse zu ziehen sind Kenntnisse in Datenanalyse unerlässlich.

Lehrkräfte, die verstehen, wie Daten generiert, ausgewertet und interpretiert werden, können daraus methodisch sinnvolle Konsequenzen ziehen, beispielsweise wo systematische Schwächen zu finden sind, welche Aufgabenformate besonders guten Lernerfolg erzielen, welche Unterrichts- und Fördermaßnahmen wirklich sinnvoll sind und an welchen Stellen besser differenziert werden muss.

Darüberhinaus können Lehrkräfte, die Wissen über die Mathematik hinter Statistik, Datenanalyse und KI-Modellen haben, diese im Unterricht besser einordnen. In vielen Lebenssituationen werden schon Kinder damit konfrontiert, Entscheidungen auf der Basis davon zu treffen, ob etwas wahrscheinlich oder unwahrscheinlich ist. Dies geschieht bereits in der Grundschule bei vielen Gesellschaftsspielen. Wenn Lehrkräfte Wahrscheinlichkeiten verstehen, können die ein Grundverständnis kindgerecht vermitteln und beim Denken und Entscheiden fundiert unterstützen. Sie können die Schüler:innen besser auf den Umgang mit modernen digitalen Technologien vorbereiten und ihnen beibringen, selbst erhobene oder vorliegende Statistiken zu hinterfragen, sie kritisch zu interpretieren und Daten sinnvoll darzustellen. Schon in der Grundschule kann durch einfache Formen der Datenerhebung und -auswertung ein erstes Bewusstsein dafür geschaffen werden, wie Daten Informationen liefern und Entscheidungen beeinflussen. So können Lehrkräfte das kritische Reflektieren und einen verantwortungsvollen Umgang mit Informationen fördern.

Besonders wichtig ist aber das Verständis für Künstliche Intelligenz, die unseren Alltag immer mehr beeinflusst. Nur wenn Lehrer:innen verstehen, wie digitale Systeme und KI-basierte Modelle funktionieren, können sie Kindern vermitteln, dass Informationen aus KI-System nicht objektiv und fehlerfrei sind. Kinder sollten lernen, die Ausgaben aus KI-Systemen kritisch zu hinterfragen, die Ergebnisse kritisch zu reflektieren und vor allem KI-Modelle verantwortungsvoll zu nutzen – sie müssen dahingehend Medienkompetenz erlernen.

So trägt das Wissen, welches eine Lehrkraft im Bereich Datenanalyse und digitaler Technologien mitbringt, letztendlich dazu bei, Schüler:innen auf eine Zukunft vorzubereiten, in der ein kompetenter und reflektierter Umgang mit Daten und digitalen Technologien eine grundlegende Kompetenz ist. Kompetenz in Datenanaly-

se und künstlicher Intellizgenz hilft Lehrkräften dabei, Kinder gut auf die heutige Welt vorzubereiten. Sie unterstützt nicht nrt den Unterricht, sondern fördert auch das kritische Denken und einen bewussten Umgang mit Daten.

1.2 Überblick über die Vorlesung

Diese Vorlesung hat das Ziel, die Mathematik hinter KI-Modellen zur Datenanalyse verständlich zu machen. Damit fokussiert sie sich auf den zweiten Teil des vorherigen Abschnitts. Insbesondere sind konkrete Apps oder Plattformen zur Datenauswertung im Unterricht nicht teil dieser Vorlesung. Die Vorlesung konzentriert sich auf die mathematischen Grundlagen in Wahrscheinlichkeitstheorie und linearer Algebra, sowie die mathematische Formulierung von KI-Systemen. Damit soll die Vorlesung dazu beitragen, Lehrer:innen die grundlegende Funktionsweise moderner KI-Systeme verständlich zu machen, so dass sie diese Kompetenz wie zuvor beschrieben in den Unterricht mitnehmen können.

Es wird keine höhere Mathematik als Vorwissen vorausgesetzt. Die mathematischen und statistischen Grundlagen werden im ersten Kapitel diskutiert. Die weiteren Kapitel behandeln die Grundlagen der künstlichen Intelligenz, insbesondere künstliche neuronale Netze, und das sogenannte *Large Language Model*, auf dem z.B. ChatGPT oder Gemini basieren.

1.2.1 Jupyter Notebooks

Dieses Skript enthält verschiedene Übungsaufgaben. Einige davon sind Programmieraufgaben. Dazu werden beispielhaft 6 *Jupyter Notebooks* [5] bereit gestellt. Die Notebooks sind in der Programmiersprache `Julia` [8] geschrieben und behandeln die folgenden Themen:

(1) `Julia` Basics.
(2) Matrizen und Bilder.
(3) Modelle im maschinellen Lernen.
(4) Neuronale Netze zur Klassifikation.
(5) Einfache Sprachmodelle.
(6) Large Language Models.

Das Highlight in dieser Liste ist das letzte Notebook über Large Language Models (LLMs). Hier wird ein LLM von Grund auf implementiert und trainiert, so dass jeder Baustein eines LLMs nachvollzogen werden kann.

Es werden keine tiefen Programmier- oder `Julia`-Kenntnisse benötigt um die Notebooks ausführen. Hier ist eine Anleitung zur Installation:

(1) Folgen Sie der Anleitung auf der Seite https://julialang.org/downloads/, um `Julia` auf Ihrem System zu installieren.

(2) Installieren Sie danach das Paket `IJulia` in der Julia-Konsole mit dem Befehl:

```
using Pkg
Pkg.add("IJulia")
```

(3) Ein neues Notebook starten Sie in der Julia-Konsole mit dem Befehl:

```
using IJulia
notebook()
```

(4) Im Anschluss öffnet sich ein Browser-Fenster, welches die Dateien auf Ihrem System zeigt. Navigieren Sie zum ersten Notebook und öffnen es mit einem Doppelklick.

(5) Führen Sie die Jupyter Notebooks Zeile für Zeile aus. Versuchen Sie nachzuvollziehen, was in den einzelnen Zeilen passiert.

2 Mathematische Grundlagen

Dieses Kapitel führt in die mathematischen Grundlagen ein, die wir später brauchen werden, um die Methoden der künstlichen Intellizenz (KI) und des maschinellen Lernens (ML) zu beschreiben und zu verstehen. Für mehr Details wird auf [16] (für den wahrscheinlichkeitstheoretischen Teil) und [15] (für lineare Algebra) verwiesen.

2.1 Deskriptive Statistik

Das Ziel der deskriptiven Statistik ist es, Datenmengen zu *beschreiben*; dies geschieht hauptsächlich durch die Berechnung von Kennzahlen und die graphische Veranschaulichung. Diese Kennzahlen helfen uns, Muster und wesentliche Eigenschaften in den Daten zu erkennen.

Mathematisch lässt sich dies wie folgt modellieren: Gegeben ist eine Menge von *Daten*

$$D = \{e_1, \dots, e_N\} \subseteq \Omega,$$

wobei Ω eine Grundmenge darstellt. Die Elemente e_i für $1 \leq i \leq N$ sind die einzelnen Datenpunkte. Wir haben also insgesamt N Daten in D. Das Ziel ist es nun, *Informationen* aus D zu gewinnen und zusammenzufassen.

Definition 2.1.1 (Grundbegriffe)**.** Ω heißt statistische Grundgesamtheit oder Menge aller möglichen Ereignisse. D heißt Stichprobe oder Datensatz.

Beispiel 2.1.1.

(1) Wir werfen einen Würfel $N = 7$ mal. Dann ist $D = \{\text{Wurf } 1, \dots, \text{Wurf } 7\}$ die konkrete Sammlung unserer Beobachtungen. Ω ist die Menge aller theoretisch möglichen Würfe, inklusive aller Begleitumstände (Zeitpunkt, wer wirft,

Temperatur etc.). Diese Unterscheidung ist wichtig, da Ω alle potenziellen Informationen enthält, während D nur die tatsächlich beobachteten Daten repräsentiert.

(2) Wir beobachten die Haarfarben unserer Kommiliton:innen. Das Ergebnis könnte sein: $D = \{\text{blond}, \text{rot}, \text{schwarz}\}$. Ω ist dann die Menge aller möglichen Haarfarben, die grundsätzlich auftreten können.

Dieses Beispiel verdeutlicht, dass Daten nicht zwangsläufig durch Zahlen ausgedrückt werden müssen. Sie können auch kategorisch sein. Es kann auch verschiedene Möglichkeiten geben, die Grundmenge Ω zu definieren, je nachdem, welche Aspekte der Daten wir berücksichtigen möchten. Beispielsweise könnten wir Ω im zweiten Beispiel auf die Menge der Haarfarben aller Studierenden an der Universität erweitern. Dies würde die Analyse beeinflussen, da wir dann eine kleinere Grundgesamtheit betrachten.

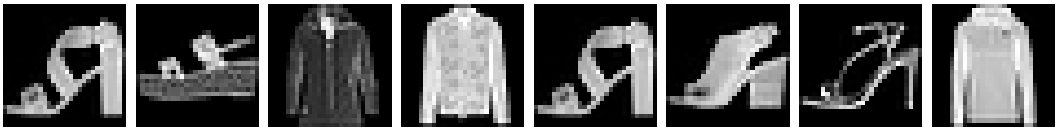
2.1.1 Merkmale

Um Daten besser zu verstehen zu können, betrachten wir verschiedene *Merkmale*. Mathematisch modellieren wir diese wie folgt:

Definition 2.1.2 (Merkmal)**.** Seien Ω und W Mengen und $X : \Omega \to W$ eine Funktion. Diese Funktion ordnet jedem Element aus der Grundmenge Ω einen Wert aus der Menge W zu. Wir nennen X Merkmal (oder Messwert) und W Wertebereich. Ist $D = \{e_1, \ldots, e_N\} \subset \Omega$ ein Datensatz, so nennen wir $x_i = X(e_i)$ Beobachtung.

Die Funktion X ist entscheidend, da sie bestimmt, *welche* Information aus den Daten extrahiert wird.

Beispiel 2.1.2. Ω ist die Menge aller Münzwürfe und $D = \{\text{Wurf } 1, \ldots, \text{Wurf N}\}$. Sei weiterhin $W = \{\text{Kopf}, \text{Zahl}\}$. Dann ordnet die Funktion $X : \Omega \to W$ einem Wurf Kopf oder Zahl zu. Dies ist ein einfaches Beispiel für ein Merkmal, das uns sagt, wie ein einzelner Datenpunkt (in diesem Fall ein einzelner Wurf) kategorisiert wird.



Beispiel 2.1.3. Der `FashionMNIST` [28] Datensatz enthält Bilder verschiedener Kleidungsstücke. Wir betrachten hier 8 Beispielbilder.

Wir bezeichnen das erste Bild als e_1, das zweite als e_2 usw. Dann ist der Datensatz

$$D = \{e_1, \ldots, e_8\}.$$

Das Merkmal, das uns interessiert, ist die Art des Kleidungsstücks, das auf dem Bild zu sehen ist. Beispielsweise ist

$$X(e_1) = \text{Sandale}, \qquad X(e_4) = \text{Jacke}.$$

Das Ziel im Abschnitt über maschinelles Lernen wird es sein, eine Funktion X zu lernen, die jedem Bild das korrekte Kleidungsstück zuordnet. Interessanterweise können Menschen diese Zuordnung oft automatisch und mühelos vornehmen – das ist ein Beispiel für Intelligenz, die wir mit maschinellem Lernen nachbilden wollen.

Warum ist die Unterscheidung zwischen D und W wichtig? Wozu brauchen wir D überhaupt? Reicht es nicht, nur mit Merkmalen zu arbeiten?

Wir benötigen die Unterscheidung zwischen D und W, um verschiedene Datenpunkte auch dann unterscheiden zu können, wenn sie denselben Wert im Wertebereich W haben. Betrachten wir erneut den Münzwurf: Wir möchten den ersten Wurf vom zweiten unterscheiden, auch wenn beide Kopf zeigen. Im Wertebereich W sind beide Würfe identisch, da sie beide "Kopf" sind. Um die Reihenfolge und die Individualität der Würfe zu berücksichtigen, benötigen wir den Datensatz D. Ähnliches gilt für die Bilder im FashionMNIST-Datensatz: Wenn wir nur W betrachten, verlieren wir die Information, dass es sich um unterschiedliche Bilder handelt, die durch 28×28 Pixel Grauwerte dargestellt werden.

Die Rolle von W ist es also, die für uns relevanten Informationen zu extrahieren und darzustellen, während D die vollständigen Daten, wie sie vorliegen, enthält.

Wir fassen verschiedene Arten von Merkmalen zusammen:

Definition 2.1.3 (Merkmaltypen)**.**

(1) Ein X ist ein *quantitatives Merkmal*, wenn $W \subseteq \mathbb{R}$. Das bedeutet, dass die Beobachtungen $x_i = X(e_i)$ durch messbare Größen definiert sind. Wir unterscheiden zwei Fälle:

- *diskret*:

  W besteht aus isolierten Punkten (z.B. Alter in Jahren: $W = \mathbb{N}$).

- *stetig*:

  W ist kontinuierlich (z.B. Temperatur in °C: $W = (-273, \infty)$).

(2) Ein X ist ein *nominales Merkmal*, wenn W eine endliche Menge von Bezeichnungen (Wörtern oder Buchstabenfolgen) ist (z.B. $W =$ Menge der möglichen Wohnorte).

(3) Ein X ist ein *ordinales Merkmal*, wenn es nominal ist und es zusätzlich eine natürliche Ordnung auf W gibt (z.B. $W = \{\text{groß}, \text{mittel}, \text{klein}\}$ mit der Ordnung klein $<$ mittel $<$ groß).

Definition 2.1.4. Sei $X = (X_1, \ldots, X_k)$ eine Liste (ein Vektor) von Merkmalen. Dann nennen wir X ein *multivariates Merkmal*.

Beispiel 2.1.4. $X = (X_1, X_2)$, wobei $X_1 =$ Name einer Stadt (nominales Merkmal) und $X_2 =$ Anzahl Einwohner (quantitativ diskret) sind.

2.1.2 Grafische Darstellung von Häufigkeitsverteilungen

Da wir nun die Begrifflichkeiten geklärt haben, wollen wir nun Methoden zur Beschreibung von Datensätzen verstehen. In diesem Abschnitt behandeln wir grafische Darstellungen, um Daten zu visualisieren, was uns dabei hilft Muster, Ausreißer und die zugrundeliegende Struktur der Daten zu erkennen.

Sei dazu Ω eine statistische Grundgesamtheit; $X : \Omega \to W$ ein Merkmal mit Wertebereich W und $D = \{e_1, \ldots, e_N\}$ ein Datensatz der Größe N. Wir treffen folgende Annahmen.

- X ist entweder nominal, ordinal oder quantitativ diskret.
- W ist endlich, wobei $M := \#W$ die Anzahl der möglichen Ausprägungen des Merkmals beschreibt.

Außerdem nehmen wir an, dass

$$W = \{w_1, \ldots, w_M\}$$

nummeriert ist. Diese Nummerierung ist wichtig, um die Ausprägungen des Merkmals quantitativ zu behandeln, auch wenn sie ursprünglich keine numerische Bedeutung haben.

Als erstes definieren wir die sogenannte *empirische Häufigkeitsverteilung* der Daten D. Diese Verteilung gibt uns einen Überblick darüber, wie oft jede Ausprägung des Merkmals im Datensatz vorkommt.

Definition 2.1.5 (Häufigkeitsverteilung)**.** Die Zahl

$$N_j := \#\{i \mid 1 \leq i \leq N, X(e_i) = w_j\} = \#X^{-1}(w_j)$$

ist die absolute Häufigkeit des Merkmals w_j. Die relative Häufigkeit von w_j ist

$$f_j := \frac{N_j}{N}$$

Weiterhin ist $(N_1, \ldots, N_M)$ die absolute Häufigkeitsverteilung des Merkmals X der Daten D und $(f_1, \ldots, f_M)$ ist die relative Häufigkeitsverteilung.

Es gilt immer:

(1) $\sum_{j=1}^{M} N_j = N$

(2) $\sum_{j=1}^{M} f_j = \sum_{j=1}^{M} \frac{N_j}{N} = \frac{1}{N} \sum_{j=1}^{M} N_j = \frac{N}{N} = 1.$

Die zweite Eigenschaft stellt sicher, dass die Summe der relativen Häufigkeiten immer 1 ergibt, was sie zu einer Wahrscheinlichkeitsverteilung macht (siehe Definition 2.3.1). Wahrscheinlichkeitsverteilungen sind ein grundlegendes Konzept im maschinellen Lernen. Sie spielen im Kapitel zu Large Language Models (Kapitel 4) eine zentrale Rolle.

Beispiel 2.1.5. $W = \{a, b, c\}$ (nominales Merkmal mit $M = 3$ Ausprägungen) und $D = \{e_1, \ldots, e_5\}$ ($N = 5$ Datenpunkte). Weiterhin sei

$$X(e_1) = a,\ X(e_2) = b,\ X(e_3) = a,\ X(e_4) = a,\ X(e_5) = a.$$

Sei etwa $w_1 = a, w_2 = b, w_3 = c$. Dann gilt:

$$(N_1, N_2, N_3) = (4, 1, 0), \quad (f_1, f_2, f_3) = (0.8, 0.2, 0)$$

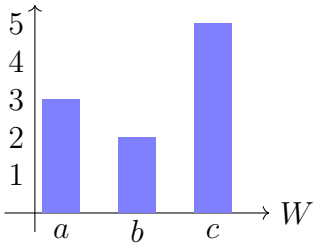
(4-mal a, 1-mal b und 0-mal c, bzw. 80% der Daten sind a, 20% sind b und 0% sind c).

Definition 2.1.6 (Balkendiagramm)**.** Ein Balkendiagramm stellt die Verteilung eines nominalen, ordinalen oder quantitativ diskreten Merkmals dar. Auf der x-Achse werden die Werte aus W aufgetragen. Über $w_j \in W$ wird ein Balken der Länge N_j (oder f_j) gezeichnet.

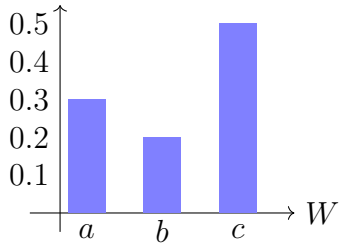
Hier ist ein simples Beispiel.

Beispiel 2.1.6. $W = \{a, b, c\}$, $(N_1, N_2, N_3) = (3, 2, 4)$, $a = w_1, b = w_2, c = w_3$

absolute Häufigkeit N_j



relative Häufigkeit f_j



Für quantitativ kontinuierliche Daten benutzen wir ein *Histogramm*. Die Idee ist es, einen kontinuierlichen Wertebereich $W \in \mathbb{R}$ in Einzelstücke zu zerlegen. Wir sagen auch *diskretisieren*.

Definition 2.1.7 (Diskretisierung)**.** Sei $W \subseteq \mathbb{R}$ ein kontinuierlicher Wertebereich. Seien weiterhin $I_j = [v_j, v_{j+1})$, $1 \leq j \leq k$, halboffene Intervalle, sodass

(1) $W \subseteq \bigcup_{j=1}^{k} I_j$

(2) $v_j < v_{j+1}$ für alle $1 \leq j \leq k-1$ (d.h. jeder Punkt in W liegt in genau einem Intervall I_j).

Dann nennen wir $(I_1, \ldots, I_k)$ eine Diskretisierung von W.

Eine sinnvolle Diskretisierung ist entscheidend für die Interpretation des Histogramms. Die Wahl der Intervalle (Bin-Breite) kann die Visualisierung erheblich beeinflussen.

Definition 2.1.8 (Häufigkeitsverteilung quantitativ stetiger Merkmale)**.** Gegeben sei ein Merkmal $X : \Omega \to W$ und ein Datensatz $D = \{e_1, \ldots, e_N\} \subset \Omega$. Sei $I = (I_1, \ldots, I_k)$ eine Diskretisierung von W. Wir definieren (wie zuvor)

$$N_j = \#\{i \mid 1 \leq i \leq N : X(e_i) \in I_j\} = \#X^{-1}(I_j), \quad f_j = \frac{N_j}{N}.$$

(1) $(N_1, \ldots, N_k)$ ist die absolute Häufigkeitsverteilung von X bzgl. der Diskretisierung.

(2) $(f_1, \ldots, f_k)$ ist die relative Häufigkeitsverteilung.

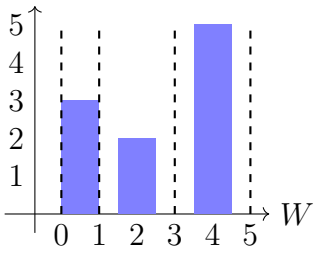
Definition 2.1.9. Ein Histogramm stellt die Häufigkeitsverteilung eines quantitativ stetigen Merkmals als Balkendiagramm nach Transformation in ein quantitativ diskretes Merkmal dar.

Beispiel 2.1.7. $W = [0, 5)$, $I = (I_1, I_2, I_3)$ mit

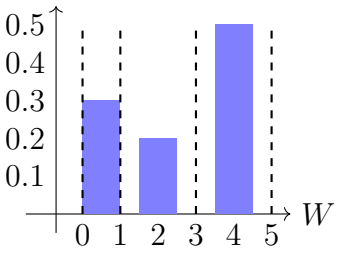
$$I_1 = [0, 1), \quad I_2 = [1, 3), \quad I_3 = [3, 5).$$

Angenommen $(N_1, N_2, N_3) = (3, 2, 4)$ ist die zu dieser Diskretisierung gehörende Verteilung. Dann ist das Histogramm wie folgt.

absolute Häufigkeit N_j



relative Häufigkeit f_j



Die gestrichelten Linien geben hierbei die Grenzen der einzelnen Intervalle in der Diskretisierung an (normalerweise werden diese in einem Histogramm aber nicht angezeigt).

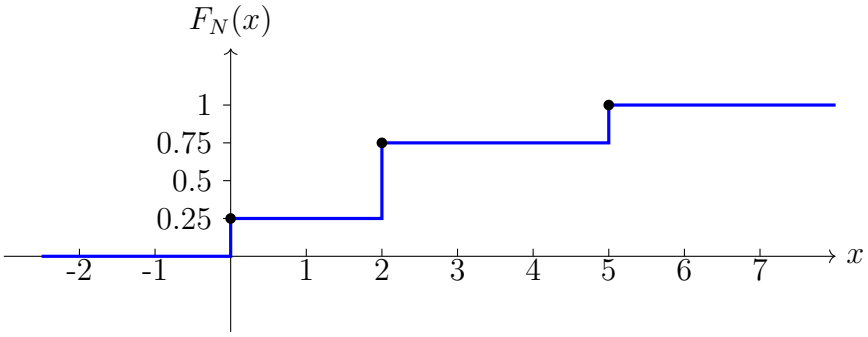
Eine weitere Möglichkeit die Häufigkeitsverteilung eines quantitativen Merkmals zu beschreiben ist die empirische Verteilungsfunktion. Die empirische Verteilungsfunktion gibt für jeden Wert x den Anteil der Beobachtungen an, die kleiner oder gleich x sind.

Definition 2.1.10. Sei X ein Merkmal mit Wertebereich $W \subseteq \mathbb{R}$. Angenommen wir haben Daten mit Beobachtungen $x_1 = X(e_1), \ldots, x_N = X(e_N)$. Die zugehörige empirische Verteilungsfunktion ist:

$$F_N(x) = \frac{1}{N} \#\{i \mid 1 \leq i \leq N : x_i \leq x\}.$$

Die empirische Verteilungsfunktion dient zur Schätzung der Verteilungsfunktion der Grundgesamtheit.

Beispiel 2.1.8. $N = 4, x_1 = 0, x_2 = 2, x_3 = 2, x_4 = 5$



2.1.3 Übungsaufgaben

Aufgabe 2.1.1. Wir beobachten ein multivariates Merkmal mit zwei Ausprägungen `Initialen` und `Note`:

Initialen	AB	CD	GH	NO	TU
Note	1	4	2	4	3

(1) Um welche Art von Merkmalen handelt es sich hier?

(2) Berechnen Sie Rangwerte, Mittelwert und Median des Merkmals `Note`. Wie ändern sich diese Werte wenn wir eine zusätzliche Beobachtung (`ML`, 4) machen?

Aufgabe 2.1.2. Laden sie das Package `DataFrames` (es muss eventuell zuerst installiert werden) in eine `Julia` Session:

```
using DataFrames
```

Definieren Sie in `Julia` zwei Vektoren `Initialen` und `Note` mit den Einträgen aus der Tabelle aus Aufgabe 1. Definieren Sie dann ein `DataFrame` mit dem Befehl.

```
D = DataFrame(Initialen=Initialen, Note=Note)
```

Erklären Sie die Ergebnisse folgender Zeilen:

```
D
D[1:2, :]
D[:, 1:2]
size(D)
sort(D, [:Note])
D[D.Note .> 2, :]
```

Aufgabe 2.1.3. Gegeben seien die drei Merkmale in der folgenden Tabelle.

Alter	Fernsehzeit in h/Woche	besitzt ein Smartphone
6	5	nein
8	10	ja
7	3	nein
10	15	ja
6	8	ja
7	9	nein
7	13	ja
9	8	ja

(1) Beurteilen Sie, um welche Art von Merkmalen es sich jeweils handelt.

(2) Speichern Sie die Daten in einem `DataFrame`.

(3) Berechnen Sie ein `DataFrame`, welches nur die Daten von Kindern enthält, die ein Smartphone besitzen.

(4) Berechnen Sie ein `DataFrame`, welches nur die Daten von Kindern enthält, die älter als 8 sind.

Aufgabe 2.1.4. Diese Aufgabe soll mit der grundlegenden Syntax von `Julia` vertraut machen.

(1) Definieren Sie in `Julia` folgende Vektoren:

```
x = [1, 2, 3, 4, 5, 6]
y = collect(1:6)
```

Erklären Sie die Ergebnisse der folgenden Zeilen:

```
x
x - y
x + y
x .* y
x.^2 .+ 2
length(x)
n = length(x + y)
sum(x+y)/n
```

(2) Definieren Sie in `Julia` folgende Vektoren:

```
x = collect(0:2:10)
y = collect(5:-1:1)
z = ["a", "b", "c"]
```

Erklären Sie die Ergebnisse der folgenden Zeilen:

```
y
x[1:3]
y[[1, 2, 4]]
z[2]
[x; z]
x .> 2
x[x .> 2]
x[x .> 2 .&& x .<= 8]
y[y .== 5 .|| y .< 2]
```

(3) Führen Sie folgenden Code aus. Was passiert?

```
for i in 1:10
  println(i)
end
```

2.2 Kennzahlen der deskriptiven Statistik

Im letzten Abschnitt haben wir Daten visuell beschrieben. Im Gegensatz dazu wollen wir in diesem Abschnitt Daten durch Kennzahlen beschreiben.

Wir fokussieren uns auf quantitative Daten mit Wertebereich $W \subseteq \mathbb{R}$. Die relevanten Informationen, die wir in diesen Abschnitt verstehen wollen sind (1) wo sich die Daten befinden (zentrale Tendenz) und (2) wie weit verstreut die Daten sind (Variabilität).

Beispiel 2.2.1. Verteilung von Temperaturdaten. (1) Wo: Wenn die Daten alle bei um die 20°C liegen, dann handelt es sich um einen warmen Ort. (2) Wie weit: In Osnabrück liegt die Temperatur das Jahr über ca. zwischen -10°C und $+30$°C. Temperaturdaten sind hier über das Intervall $[-10, 30]$ verstreut. Auf den kanarischen Inseln hingegen ist das ganze Jahr über eine Temperatur von ca. 25°C. Hier sind die Temperaturdaten weniger verstreut.

Für die relevanten Definitionen ordnen wir die Daten zunächst. Die Sortierung der Daten ist oft ein erster Schritt in der Datenanalyse. Wir zuvor bezeichnen wir mit $x_i = X(e_i)$ das Merkmal des Datenpunktes $e_i \in D$.

Definition 2.2.1 (Rangwerte)**.** Seien $x_1, \ldots, x_N \in \mathbb{R}$ Beobachtungen eines quantitativen Merkmals. Wir nummerieren um, so dass $x_{(1)} \leq x_{(2)} \leq x_{(3)} \leq \ldots \leq x_{(N)}$. Wir nennen $x_{(i)}$ den i-ten Rangwert, $x_{(1)}$ das Minimum und $x_{(N)}$ das Maximum.

Beispiel 2.2.2. Gegeben sind Daten $x_1 = 3, x_2 = 6, x_3 = 1, x_4 = 1, x_5 = 4$. Die Rangwerte sind dann: $x_{(1)} = 1, x_{(2)} = 1, x_{(3)} = 3, x_{(4)} = 4, x_{(5)} = 6$.

2.2.1 Lageparameter

Lageparameter beantworten die Frage "Wo liegen die Daten?". Sie geben uns ein Maß für die zentrale Tendenz der Daten.

Die erste Art von Lageparametern sind die sogenannten *Quantile*. Quantile teilen die Daten in gleich große Teile und geben uns Informationen über die Verteilung.

Definition 2.2.2 (Quantile)**.** Seien $x_1, \ldots, x_N \in \mathbb{R}$ Beobachtungen. Sei $0 < p < 1$. Das p-Quantil der Daten ist definiert als

$$\tilde{x}_p = \begin{cases} x_{(k)}, & \text{falls } pN \notin \mathbb{N} \text{ und } pN < k < pN + 1 \\ \frac{1}{2}(x_{(k)} + x_{(k+1)}), & \text{falls } pN = k \in \mathbb{N}. \end{cases}$$

Die Idee dieser Definition ist wie folgt: Für $0 < p < 1$ ist das p-Quantil $\tilde{x}_p$ ein Punkt, so dass $p \cdot 100\%$ der Daten kleiner als $\tilde{x}_p$ sind:

$$x_{(1)} \leq \ldots \leq x_{(k)} \leq \tilde{x}_p \leq x_{(k+1)} \leq \ldots \leq x_{(N)}, \quad \text{sodass } k \approx pN.$$

Quantile werden beispielsweise in der Datenvorverarbeitung verwendet, um Ausreißer zu identifizieren und zu behandeln.

Definition 2.2.3 (Median)**.** Das $\frac{1}{2}$-Quantil $\tilde{x}_{1/2}$ heißt Median.

Nach Definition liegen 50% der Daten über dem Median und 50% darunter.

Beispiel 2.2.3. Gegeben seien Daten $x_1 = 1, x_2 = 1, x_3 = 4, x_4 = 6$. Die Rangwerte sind dann $x_{(1)} = 1 \leq x_{(2)} = 1 \leq x_{(3)} = 4 \leq x_{(4)} = 6$ Dann ist der Median dieser Daten $\tilde{x}_{1/2} = 2.5$.

Jetzt haben zusätzlich $x_{(5)} = 10$. Dann wird der Median zu $\tilde{x}_{1/2} = x_{(3)} = 4$.

Definition 2.2.4 (Quartile)**.** Das $\frac{1}{4}$-Quantil heißt unteres Quartil. Das $\frac{3}{4}$-Quantil heißt oberes Quartil.

Der Median misst das "Zentrum" der Daten. Eine alternative Definition für eine Art Zentrum ist der *Mittelwert*. Der Mittelwert ist der intuitivste Lageparameter, aber er kann im Gegensatz zum Median eher durch Ausreißer beeinflusst werden.

Definition 2.2.5 (Mittelwert)**.** Seien $x_1, \ldots, x_N \in \mathbb{R}$ Beobachtungen. Der Mittelwert der Daten ist definiert als

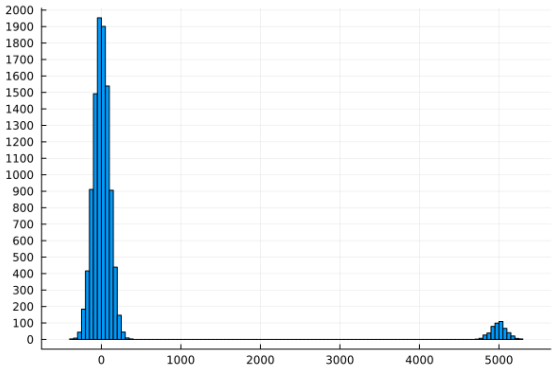
$$\bar{x} := \frac{1}{N}(x_1 + x_2 + \ldots + x_N).$$

Beispiel 2.2.4. Gegeben seien die Daten $x_1 = 3, x_2 = 10, x_3 = 1, x_4 = 4, x_5 = 5$. Dann ist $N = 5$. Median und Mittelwert sind dann

$$\tilde{x}_{1/2} = 3, \quad \bar{x} = \frac{1}{5}(3 + 10 + 1 + 4 + 5) = \frac{23}{5} = 4.6.$$

Der Median ist ein robuster Lageparameter, der weniger anfällig für Ausreißer ist als der Mittelwert. Das folgende Beispiel illustriert, wie Extremwerte den Mittelwert beeinflussen, während der Median demgegenüber robuster ist.

Beispiel 2.2.5. Angenommen wir haben einen Datensatz, den wir in einem Histogramm wie folgt darstellen.



Dann ist der Median ungefähr bei $x = 0$, weil der größte Teil der Daten sich um 0 zentriert. Andererseits ist der Mittelwert sicher größer als 0, da ein kleiner Teil der Daten bei $x = 5000$ liegt. In diesem Fall wäre der Median eine bessere Wahl als Lageparameter.

2.2.2 Streuungsparameter

Streuungsparameter beantworten die Frage "Wie weit sind die Daten verstreut?". Sie geben uns ein Maß für die Variabilität der Daten.

Im Folgenden seien wieder $x_1, \ldots, x_N \in \mathbb{R}$ Beobachtungen mit zugehörigen Rangwerten $x_{(1)} \leq \ldots \leq x_{(N)}$.

Definition 2.2.6 (Spannweite und Quartilsabstand)**.** Wir nennen $R = x_{(N)} - x_{(1)}$ und $Q = \tilde{x}_{3/4} - \tilde{x}_{1/4}$ die Spannweite und den Quartilsabstand der Daten.

Definition 2.2.7 (Standardabweichung und Varianz)**.** Die Standardabweichung der Daten ist definiert als

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}.$$

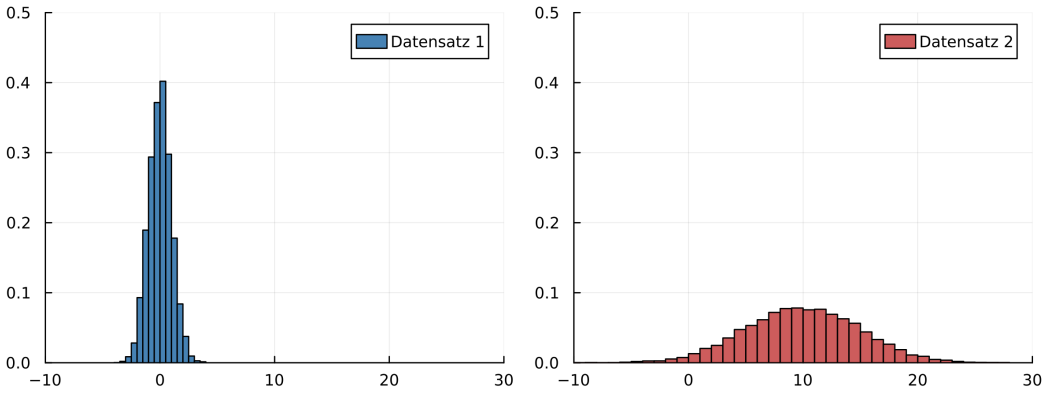
Die Varianz der Daten, oder auch Stichprobenvarianz, ist definiert als

$$s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2.$$

D.h. Varianz und Standardabweichung messen die durchschnittliche quadrierte Abweichung der Beobachtungen x_i zum Mittelwert $\bar{x}$.

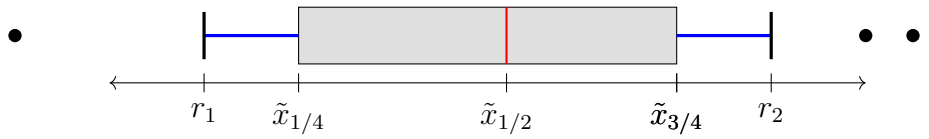
Der Nenner $N-1$ in dieser Definition ist kein Schreibfehler. Man normalisiert mit $\frac{1}{N-1}$ anstatt mit $\frac{1}{N}$, damit s ein sogenannter "unverzerrter Schätzer" wird. Die Mathematik hinter dieser Aussage geht allerdings über den mathematischen Inhalt dieser Vorlesung hinaus.

Beispiel 2.2.6. Angenommen wir haben zwei Datensätze, die wir in Histogrammen wie folgt darstellen.



Da die Daten auf der rechten Seite weiter verstreut sind, ist die Varianz dieses Datensatzes größer als auf der linken Seite. Eine höhere Varianz kann beispielsweise dazu führen, dass ein Modell weniger genau ist.

Alle Kennzahlen können in einem *Boxplot* zusammengefasst werden.



Hierbei ist r_1 die kleinste Beobachtung, die größer als $\tilde{x}_{1/4} - (3/2)Q$ ist, und r_2 die größte Beobachtung, die kleiner als $\tilde{x}_{3/4} + (3/2)Q$ ist (Erinnerung: $Q = \tilde{x}_{3/4} - \tilde{x}_{1/4}$ ist der Quartilsabstand). Wir nennen w_1 den *unteren Whisker* und w_2 den *oberen Whisker*. Das Boxplot ist ein nützliches Werkzeug, um Ausreißer zu identifizieren und die Verteilung der Daten visuell zu überprüfen.

2.2.3 Übungsaufgaben

Aufgabe 2.2.1. Während einer Geschwindigkeits- und Verkehrskontrolle werden in einer 70er Zone von 12 Autos die folgenden Daten aufgenommen.

Kennzeichen	Anzahl Personen im Auto	Geschwindigkeit in km/h
OS	4	75.5
OS	2	68.2
BI	1	65.3
GÖ	5	60.1
MS	1	80.9
BI	2	100.0
MS	3	87.0
OS	1	70.2
OS	2	72.5
HB	1	69.6
B	3	71.4
OS	5	87.1

Die folgenden Aufgaben können Sie per Hand oder mit Hilfe von `Julia` lösen (siehe auch Aufgabe 2.2.2).

(1) Um welche Art von Merkmalen handelt es sich hier?

(2) Erstellen Sie absolute und relative Häufigkeitsverteilungen für jede oben aufgeführte Merkmal. Verwenden Sie bei der Merkmal *Geschwindigkeit* eine Klassenbreite von 10.

(3) Erstellen Sie für die Häufigkeitstabellen aus Teil a) eine passende grafische Darstellung.

(4) Berechnen Sie Mittelwert, den Median und die Standardabweichung für das Merkmal *Geschwindigkeit*.

(5) Erstellen Sie die empirische Verteilungsfunktion für das Merkmal *Anzahl Personen im Auto*.

Aufgabe 2.2.2. Lesen Sie die `Julia` Dokumentationen für

(1) Histogramme:

`https://docs.juliaplots.org/latest/series_types/histogram/`

(2) Mittelwert:

`https://docs.julialang.org/en/v1/stdlib/Statistics/#Statistics.mean`

(3) Median:

`https://docs.julialang.org/en/v1/stdlib/Statistics/#Statistics.median`

(4) Standardabweichung:

`https://docs.julialang.org/en/v1/stdlib/Statistics/#Statistics.std`

Aufgabe 2.2.3. Diskutieren Sie die Antwort der Bundesregierung auf Frage 4. der Kleinen Anfrage, welche unter

`https://dip21.bundestag.de/dip21/btd/19/221/1922109.pdf`

verfügbar ist. Warum fällt die Antwort so aus?

Aufgabe 2.2.4. In einem Forschungsprojekt wurden zu zwei Zeitpunkten Messungen einer physikalischen Größe an einer Materialprobe durchgeführt. Für jeden Messzeitpunkt wurde ein Bild des Messbereichs aufgenommen und ein automatisches Erkennungs- bzw. Zählsystem ermittelte, wie viele Defekte (z. B. Mikrorisse) pro Bildausschnitt vorkamen.

Nach 9.000 Belastungszyklen wurden 54 Bildausschnitte ausgewertet, wobei die gefundenen Defektzahlen wie folgt sortiert vorliegen:

```
317 405 528 529 567 604 611 614 624 626 633 642 674 677
691 704 708 714 724 730 750 786 790 790 800 801 805 809
828 836 840 841 850 869 872 876 877 878 883 894 898 928
940 941 942 946 948 949 976 1003 1004 1010 1024 1028
```

Nach 12.000 Belastungszyklen wurden erneut 5 Bildausschnitte analysiert; die sortierten Defektzahlen lauten:

```
463 489 543 561 574 644 688 724 735 768 778 799 800 807
813 832 835 845 847 847 866 871 877 888 892 894 902 903
916 918 918 926 929 929 932 935 947 947 950 953 953 961
962 993 1002 1010 1012 1022 1033 1034 1036 1045 1063 1112
```

(1) Erstellen Sie in `Julia` ein `DataFrame`, welches die Daten enthält.

(2) Erstellen Sie jeweils ein Histogramm für die Daten. Verwenden Sie als Klassenbreite 100 und arbeiten Sie auf dem Intervall (300, 1200].

(3) Berechnen Sie Mittelwert und Standardabweichung der beiden Datensätze.

Aufgabe 2.2.5. Installieren Sie das Paket `RDatasets` und laden Sie es in die aktuelle `Julia` Session.

(1) Rufen Sie das `airquality` Dataset auf:

```
data = dataset("datasets", "airquality")
```

(2) Geben Sie das Objekt `data` im Terminal oder Jupyter-Notebook aus. Was sehen Sie?

(3) Lesen Sie die Dokumentation des Datensatzes.

(4) Erklären Sie, was die folgenden Befehle bewirken.

```
names(data)
propertynames(data)
describe(data)
```

(5) Visualisieren Sie die Verteilung des Merkmals `Wind`.

(6) Berechnen Sie Mittelwert, Median, Standardabweichung sowie Quartile des Merkmals `Wind`.

2.3 Theorie des Wahrscheinlichkeitsraums

Im ersten Abschnitt haben wir die statistische Grundgesamtheit Ω definiert. Der Datensatz D war eine Teilmenge von Ω. Es ist wichtig, sich daran zu erinnern, dass die statistische Grundgesamtheit alle möglichen Datenpunkte umfasst, während ein Datensatz lediglich eine Stichprobe aus dieser Grundgesamtheit darstellt.

In diesem Abschnitt wollen wir die Wahrscheinlichkeiten von Daten abstrakt definieren. Dazu nennen wir Ω den *Ereignisraum*. Diese Formalisierung erlaubt uns, über Wahrscheinlichkeiten unabhängig von spezifischen Experimenten oder konkreten Datensätzen zu sprechen. Man stelle sich den Ereignisraum als die Menge aller denkbaren Ergebnisse vor. Teilmengen $A \subseteq \Omega$ nennen wir *Ereignisse*. Ein Ereignis repräsentiert also eine bestimmte Teilmenge der möglichen Ergebnisse. Ein Beispiel ist der Würfelwurf, bei dem Ω alle sechs möglichen Augenzahlen umfasst; ein Ereignis könnte dann das Würfeln einer geraden Zahl sein.

Definition 2.3.1 (Wahrscheinlichkeitsraum)**.** Sei Ω ein Ereignisraum und $\mathcal{A}$ eine Menge von Ereignissen. Ein Wahrscheinlichkeitsmaß ist eine Abbildung

$$P : \mathcal{A} \to [0, 1], \quad A \mapsto P(A).$$

mit folgenden Eigenschaften:

(1) $P(\emptyset) = 0$ und $P(\Omega) = 1$.

(2) $P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$, falls $A_i, A_2, \ldots$ paarweise disjunkt.

$P(A)$ heißt dann die Wahrscheinlichkeit von $A \in \mathcal{A}$. Das Tripel $(\Omega, \mathcal{A}, P)$ heißt Wahrscheinlichkeitsraum.

(Bemerkung: Nicht jede Menge $\mathcal{A}$ von Ereignissen kann in der Definition gewählt werden. $\mathcal{A}$ muss ein sogenannte σ-Algebra sein.)

Diese Definition kann zunächst abstrakt wirken, ist aber das Fundament der Wahrscheinlichkeitstheorie und somit auch des maschinellen Lernens. Ω definiert, was überhaupt passieren kann, $\mathcal{A}$ legt fest, welche Ereignisse wir überhaupt messen können, und P quantifiziert, wie wahrscheinlich ein bestimmtes Ereignis $A \in \mathcal{A}$ ist. Die beiden Eigenschaften des Wahrscheinlichkeitsmaßes gewährleisten eine konsistente und logische Berechnung von Wahrscheinlichkeiten. Die erste Eigenschaft besagt, dass kein Ergebnis die Wahrscheinlichkeit 0 besitzt und das alle Ereignisse zusammen die Wahrscheinlichkeit von 1 hat. Die zweite Eigenschaft, die Additivität, erlaubt uns, die Wahrscheinlichkeit komplexer Ereignisse aus den Wahrscheinlichkeiten einfacher, disjunkter Ereignisse zu berechnen.

Das nächste Beispiel ist sehr einfach, aber es veranschaulicht deutlich das Konzept eines Wahrscheinlichkeitsraums.

Beispiel 2.3.1. Sei Ω = Menge aller Münzwürfe und $X : \Omega \to \{\text{Kopf}, \text{Zahl}\}$. Wir betrachten das Ereignis $A = \{w \in \Omega \mid X(w) = \text{Kopf}\}$. Dann ist $P(A)$ die Wahrscheinlichkeit Kopf zu werfen.

Im Kontext des maschinellen Lernens verwenden wir Wahrscheinlichkeitsräume, um Unsicherheit in Daten und Modellen zu modellieren. Wahrscheinlichkeitsräume bilden auch die Grundlage für generative Modelle wie *Large Language Models* (LLM), die die Verteilung der Trainingsdaten erlernen, um neue Daten zu generieren.

Wir geben nun noch dre Beispiele spezifischer Wahrscheinlichkeitsräume

Beispiel 2.3.2 (Endlicher/diskreter Wahrscheinlichkeitsraum)**.** Sei $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum. Falls Ω endlich/diskret ist, nennen wir $(\Omega, \mathcal{A}, P)$ endlichen/diskreten Wahrscheinlichkeitsraum.

Beispiel 2.3.3 (Gleichverteilung)**.** Sei Ω endlich. Dann heißt das Wahrscheinlichkeitsmaß $P(A) = \frac{\#A}{\#\Omega}$ für $A \subseteq \Omega$. ($\#A$ = Anzahl Elemente in A) das Maß der Gleichverteilung auf Ω. Insbesondere gilt für alle $w \in \Omega$: $P(\{w\}) = \frac{1}{\#\Omega}$.

Beispiel 2.3.4 (Das Urnenmodell)**.** Die Theorie der Wahrscheinlichkeitsräume kann auf viele verschiedene Arten veranschaulicht werden. Ein klassisches Beispiel ist das *Urnenmodell*.

In einer Urne befinden sich n Kugeln. Das Zufallsexperiment im Urnenmodell besteht darin, zufällig k Kugeln aus der Urne zu ziehen. Dabei nehmen wir an, dass jede Kugel die gleiche Wahrscheinlichkeit hat, gezogen zu werden. Dieses Modell ist ein grundlegendes Werkzeug in der Wahrscheinlichkeitstheorie und dient als anschauliches Beispiel. Es findet auch Anwendung in verschiedenen Bereichen des maschinellen Lernens, beispielsweise bei der Bewertung von Stichprobenverfahren und der Analyse von Algorithmen, die auf zufälliger Auswahl basieren.

Wenn $k = 1$ Kugel gezogen wird, dann gilt $P(\text{"Kugel } i \text{ wird gezogen"}) = \frac{1}{n}$. Diese einfache Wahrscheinlichkeit ergibt sich direkt aus der Annahme, dass jede Kugel gleich wahrscheinlich gezogen wird. Was passiert aber, wenn wir $k > 1$ Kugeln ziehen? Hier ergeben sich verschiedene Möglichkeiten, die auf unterschiedlichen Annahmen über das Zufallsexperiment basieren.

Zwei grundlegende Fragen stellen sich dabei: Ziehen wir die Kugeln mit oder ohne Zurücklegen? Und ist die Reihenfolge der gezogenen Kugeln von Bedeutung oder nicht? Diese Fragen führen zu vier verschiedenen Ereignisräumen, die wir im Folgenden näher betrachten werden. Die Unterscheidung zwischen diesen Fällen ist wichtig, da sie sich direkt auf die Berechnung der Wahrscheinlichkeiten auswirkt.

Wir benennen die entsprechenden Ereignisräume wie folgt:

	mit Reihenfolge	ohne Reihenfolge
mit Zurücklegen	$\Omega_{mZ,mR}$	$\Omega_{mZ,oR}$
ohne Zurücklegen	$\Omega_{oZ,mR}$	$\Omega_{oZ,k,oR}$

Um die Gleichverteilung auf diesen Ereignisräumen zu berechnen, genügt es, die Anzahl der Elemente in den jeweiligen Räumen zu bestimmen, da alle Ergebnisse gleich wahrscheinlich sind.

Mit Zurücklegen, mit Beachtung der Reihenfolge Hier ziehen wir k Kugeln aus der Urne, wobei wir nach jedem Zug die Kugel wieder zurücklegen. Das bedeutet, dass bei jedem Zug alle n Kugeln zur Auswahl stehen. Der Ereignisraum kann wie

folgt modelliert werden.

$$\Omega_{mZ,mR} = \{w = (w_1, \ldots, w_k) \mid 1 \leq w_i \leq n \text{ für alle } i\}.$$

Da bei jedem Zug alle n Kugeln gezogen werden können, gilt

$$\#\Omega_{mZ,mR} = n \cdot n \cdots n = n^k.$$

Z.B. ist $\frac{1}{6^4} = \frac{1}{1296} \approx 0.0008$ die Wahrscheinlichkeit, 4 Sechsen hintereinander zu werfen. Dies verdeutlicht, dass die Wahrscheinlichkeit, 4 Sechsen hintereinander zu werfen, ziemlich klein ist.

Ohne Zurücklegen, mit Beachtung der Reihenfolge In diesem Modell können wir, wenn wir eine bestimmte Kugel gezogen haben, sie im nächsten Zug nicht mehr ziehen. Ein Modell für den Ereignisraum ist also

$$\Omega_{oZ,mR} = \{w = (w_1, \ldots, w_k) \mid 1 \leq w_i \leq n \text{ für alle } i \text{ und } w_i \neq w_j \text{ für } i \neq j\}.$$

Für die Wahl von w_1 gibt es n Möglichkeiten, für die Wahl von w_2 gibt es $n-1$ Möglichkeiten (da $w_2 \neq w_1$) usw. Insgesamt erhalten wir:

$$\#\Omega_{oZ,mR} = n \cdot (n-1) \cdot (n-2) \cdots (n-k+1) = \frac{n!}{(n-k)!}.$$

Z.B. berechnet sich die Anzahl aller möglicher Bundesligatabellen durch 18 aus 18 Vereinen ohne Zurücklegen zu ziehen, also $\frac{18!}{(18-18)!} = 6.402.373.705.728.000$.

Ohne Zurücklegen, ohne Beachtung der Reihenfolge Ohne Beachtung der Reihenfolge heißt, dass z.B. $w_1 = (1, 2)$ und $w_2 = (2, 1)$ als gleich betrachtet werden sollen. Ein mathematisches Konstrukt, in dem die Reihenfolge irrelevant ist, ist die Menge (im Beispiel also $\{1, 2\}$). Wir erhalten folgendes Modell für den Ereignisraum.

$$\Omega_{oZ,oR} = \{w = \{w_1, \ldots, w_k\} \mid 1 \leq w_i \leq n \text{ für alle } i\}.$$

Wir zählen als Nächstes die Elemente in dieser Menge.

Wir zählen die Anzahl der Möglichkeiten $w_1, \ldots, w_n$ zu ordnen ($n!$) und teilen diese Anzahl dann durch die Anzahl Möglichkeiten die ersten k zu ordnen ($k!$) und die letzten $n-k$ zu ordnen ($(n-k)!$). Insgesamt:

$$\#\Omega_{oZ,oR} = \frac{n!}{(n-k)! \cdot k!} = \binom{n}{k}.$$

Anders gesagt: $\binom{n}{k}$ ist die Anzahl der k-elementigen Teilmengen in einer Menge mit n Elementen.

Z.B. ist die Wahrscheinlichkeit das richtige Los bei "Lottot 6 aus 49" zu ziehen gleich $\binom{49}{6} = 13.983.816$.

Mit Zurücklegen, ohne Beachtung der Reihenfolge In diesem Modell können wir Kugeln mehrfach ziehen (da mit Zurücklegen), aber die Reihenfolge soll nicht beachtet werden. Wir haben folgendes Modell für den Ereignisraum.

$$\Omega_{mZ,oR} = \{w = (w_1, \ldots, w_k) \mid 1 \leq w_i \leq n \text{ für alle } i \text{ und } w_1 + \cdots + w_n = k\}.$$

Dabei misst w_i wie oft wir Kugel i gezogen haben.

Um die Anzahl der Elemente in $\Omega_{mZ,oR}$ zu zählen, können wir jedem $w \in \Omega_{mZ,oR}$ genau ein Diagramm zuordnen. für $w = (w_1, \ldots, w_k)$ erstellen wir ein Diagramm der Form

$$\cdots\cdots|\cdots|\cdots\cdots\cdots|\cdots,$$

wobei im i-ten Abschnitt genau w_i Punkte zu sehen sein sollen. Z.B. ist das Diagramm, das wir $w = (2, 4, 1)$ zuordnen, wie folgt: $\cdot\cdot|\cdot\cdot\cdot\cdot|\cdot$. Die Anzahl der Elemente in $\Omega_{mZ,oR}$ ist also die Anzahl solcher Diagramme. Jedes Diagramm besteht aus $n+k-1$ Symbolen (k Punkte, und $n-1$ Striche). D.h. die Anzahl der Diagramm ist gleich der Anzahl Möglichkeiten aus $n+k-1$ Symbolen $n-1$ Striche zu ziehen. Dies ist gleich der Anzahl der $(n-1)$-elementigen Teilmengen in einer $n+k-1$ elementigen Menge. Daher:

$$\#\Omega_{mZ,oR} = \binom{n+k-1}{n-1}.$$

2.3.1 Frequentistischer vs. Bayes'scher Wahrscheinlichkeitsbegriff

Es gibt nun zwei grundlegend unterschiedliche Perspektiven darauf, wie die Definition eines Wahrscheinlichkeitsraumes die Verteilung reeller Daten modelliert. Diese beiden Perspektiven prägen unser Verständnis von Wahrscheinlichkeit und haben weitreichende Konsequenzen für die Art und Weise, wie wir statistische Schlüsse ziehen und Modelle erstellen.

- Sei f_n die relative Häufigkeit des Ereignisses A in n unabhängigen Zufallsexperimenten. Dann ist $P(A) = \lim_{n\to\infty} f_n$, vorausgesetzt dieser Grenzwert existiert. Das bedeutet, dass die Wahrscheinlichkeit eines Ereignisses definiert wird als der Wert, den die relative Häufigkeit des Ereignisses annimmt, wenn das Experiment unendlich oft wiederholt wird. Diese Perspektive wird als *frequentistischer Wahrscheinlichkeitsbegriff* bezeichnet. Ein zentrales Element ist hier die Annahme, dass es eine "wahre" Wahrscheinlichkeit gibt, die durch die langfristige Wiederholung des Experiments approximiert werden kann.
- Der *Bayes'sche Wahrscheinlichkeitsbegriff* definiert $P(A)$ als Erfahrungswert, nicht als Grenzwert einer Häufigkeit. Insbesondere ist es mit dem Bayes'schen Wahrscheinlichkeitsbegriff möglich, unvollständige Information über deterministische Prozesse auf Wahrscheinlichkeitsraum zu modellieren.

Der frequentistische und der Bayes'sche Wahrscheinlichkeitsbegriff stellen also zwei unterschiedliche Denkweisen dar, wie wir mit Unsicherheit umgehen. Der frequentistische Ansatz versteht Wahrscheinlichkeit als die langfristige relative Häufigkeit eines Ereignisses, also als objektives Merkmal einer wiederholbaren Situation. Es geht darum, wie oft etwas in unendlich vielen Versuchen passieren würde, wenn die Bedingungen konstant bleiben. Diese Perspektive ist eng mit der Idee verbunden, dass Wahrscheinlichkeiten Eigenschaften der Welt sind, die unabhängig von unserem Wissen existieren. Im Gegensatz dazu betrachtet der Bayes'sche Ansatz Wahrscheinlichkeit als ein Maß für unsere persönliche Erwartung an das Eintreten eines Ereignisses, basierend auf Vorwissen. Hier ist Wahrscheinlichkeit subjektiv und wird durch den Beobachter festgelegt. Es ist wichtig zu betonen, dass "subjektiv" hier nicht willkürlich bedeutet; der Bayes'sche Ansatz erfordert eine Aktualisierung der Erwartung angesichts neuer Daten. Kurz gesagt: Der frequen-

tistische Ansatz fragt "Wie oft würde das passieren?", der Bayes'sche Ansatz fragt "Wie wahrscheinlich halte ich das?". Die beiden Fragen zielen auf unterschiedliche Aspekte der Unsicherheit ab.

Im Kontext des maschinellen Lernens ist die Bayes'sche Perspektive häufig die natürlichere. Viele Probleme im maschinellen Lernen beinhalten Unsicherheit, sei es aufgrund unvollständiger Daten oder inhärenter Zufälligkeit. Hier werden Wahrscheinlichkeiten verwendet, um Vorhersagen zu treffen und Unsicherheit zu quantifizieren. Zum Beispiel könnte ein maschinell gelerntes Modell als Antwort auf die Frage, was auf dem ersten Bild in Beispiel 2.1.3 zu sehen ist, antworten: "Zu 80% ist das eine Sandale." Diese 80% geben die Wahrscheinlichkeit an, basierend auf den gelernten Daten, dass das Bild tatsächlich eine Sandale zeigt. Diese Wahrscheinlichkeit ist nicht als die Häufigkeit zu interpretieren, mit der das Modell Sandalen in Trainingsdaten gesehen hat, sondern als ein Ausdruck der Erwartung des Modells, basierend auf seinem gelernten Wissen.

Ein weiterer wichtiger Unterschied liegt in der Art und Weise, wie mit neuen Informationen umgegangen wird. Wie erwähnt, können Wahrscheinlichkeiten im Bayes'sche Ansatz im Kontext sich aktualisierender Datenlage aktualisiert werden. Diese Fähigkeit ist besonders wertvoll in Szenarien, in denen die Daten begrenzt oder unvollständig sind.

2.3.2 Venn-Diagramme

Wir fahren fort mit Eigenschaften, die Wahrscheinlichkeiten erfüllen.

Satz 2.3.1 (Eigenschaften von Wahrscheinlichkeitsräumen)**.** *Sei* $(\Omega, \mathcal{A}, P)$ *ein Wahrscheinlichkeitsraum und* A, B, C, D *Ereignisse.*

(1) $P(\Omega \setminus A) = 1 - P(A)$

(2) $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ *und* $P(A \cap B) = P(A) + P(B) - P(A \cup B)$

(3) $A \subseteq B \Rightarrow P(A) \leq P(B)$

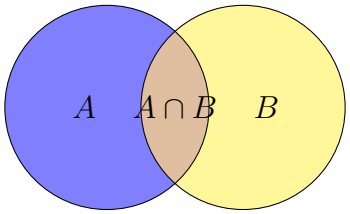
(4) Siebformel:

$$P(A \cup B \cup C) = P(A) + P(B) + P(C) \\ - P(A \cap B) - P(A \cap C) - P(B \cap C) + P(A \cap B \cap C).$$

Die Eigenschaften, die in diesem Satz aufgelistet werden, sind grundlegend für das Verständnis von Wahrscheinlichkeiten und werden in vielen Bereichen der Mathematik und Statistik angewendet. Beispielsweise hilft die erste Eigenschaft, die Wahrscheinlichkeit des Komplements eines Ereignisses zu berechnen, also die Wahrscheinlichkeit, dass ein Ereignis nicht eintritt. Die zweite Eigenschaft, die Additionsregel für Wahrscheinlichkeiten, berechnet die Wahrscheinlichkeit der Vereinigung zweier Ereignisse. Die dritte Eigenschaft drückt aus, dass die Wahrscheinlichkeit eines Teilereignisses niemals größer sein kann als die Wahrscheinlichkeit des Gesamtereignisses. Die Siebformel erweitert die Additionsregel auf drei Ereignisse und verdeutlicht die zunehmende Komplexität bei der Betrachtung mehrerer Ereignisse gleichzeitig.

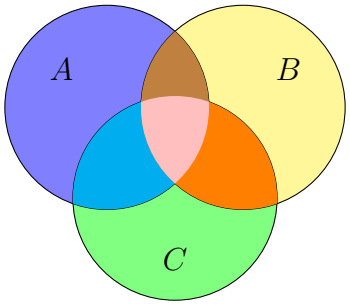
Die Aussagen des Satzes lassen sich mit Hilfe von *Venn-Diagrammen* nachvollziehen. Hierbei wird jedes Ereignis durch eine Kreisscheibe dargestellt und Überschneidungen visualisieren die Schnittmengen.

Das Venn-Diagramm für $P(A \cup B)$ illustriert, wie die Vereinigung der Ereignisse A und B berechnet wird: Man addiert die Inhalte von A und B und subtrahiert den Schnittbereich $A \cap B$, um eine doppelte Zählung zu vermeiden:



$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

Die Siebformel lässt sich ebenfalls durch ein Venn-Diagramm visualisieren.



$$\begin{aligned} P(A \cup B \cup C) = {} & P(A) + P(B) + P(C) - P(A \cap B) \\ & - P(A \cap C) - P(B \cap C) + P(A \cap B \cap C) \end{aligned}$$

2.3.3 Übungsaufgaben

Aufgabe 2.3.1. Bei einer Befragung unter den Studierenden einer Universität bezeichne A das Ereignis, dass ein:e zufällig ausgewählte:r Studierende:r eine bestimmte Wahlpflichtfach-Gruppe wählt, und B sei das Ereignis, dass ein:e zufällig ausgewählte:r Studierende:r einen Sprachkurs belegt.

(1) Beschreiben Sie die folgenden Ereignisse mit Worten:

$$A \cup B, \quad A \cap B, \quad A^c, \quad B \setminus A, \quad A \cup B^c, \quad A^c \cup B^c, \quad A^c \cap B^c$$

Hinweis: Hierbei bezeichnet A^c das Komplement von A; d.h., $A^c = \{x \in \Omega \mid x \notin A\}$.

(2) Es seien $P(A) = 0.3$, $P(B) = 0.5$ und $P(A \cap B) = 0.1$ bekannt. Berechnen Sie die Wahrscheinlichkeiten der oben angegebenen Ereignisse.

Aufgabe 2.3.2. In einem Klasse befinden sich 18 Schüler:innen. Davon haben 7 an einem bestimmten Projekttag teilgenommen. Es wird nun zufällig eine Gruppe von 4 Schüler:innen ausgewählt.

(1) Berechnen Sie die Anzahl aller möglichen Gruppen.

(2) Berechnen Sie für jedes $k \in \{0, \ldots, 4\}$ die Wahrscheinlichkeit, dass in der Gruppe genau k Schüler:innen am Projekttag teilgenommen haben.

Aufgabe 2.3.3. Wie hoch ist die Wahrscheinlichkeit, dass in einer Gruppe von k Personen zwei Personen am gleichen Tag Geburtstag haben? Diese Fragestellung heißt *Geburtstagsparadoxon*, da es eine unerwartete Antwort hat. Wir nehmen an, dass die Wahrscheinlichkeiten, dass Person i an einen bestimmten Tag Geburtstag hat, gleichverteilt sind. D.h.

$$P(\text{ Person } i \text{ hat am Tag } x \text{ Geburtstag}) = \frac{1}{365}.$$

Wie hoch ist die Wahrscheinlichkeit für $k = 23$?

Aufgabe 2.3.4. Aus einem Kartenspiel mit 5 weißen, 7 schwarzen und 3 roten Karten werden 3 Karten gezogen. Berechnen Sie die Wahrscheinlichkeit dafür, dass mindestens 2 weiße Karten gezogen wurden, wenn ohne Zurücklegen gezogen wurde.

Aufgabe 2.3.5. In einem Interview auf einen Youtube-News Kanal (Minuten 5:00–5:14) wird berichtet, dass das RKI im März 2020 günstigstenfalls 300.000 Tote, aber bis zu 1.5 Millionen Tote durch die Covid-19 Pandemie prognostizierte. Diese Aussage bezieht sich auf Abbildung 8 im RKI Bericht zur Modellierung von Beispielszenarien der SARS-CoV-2-Epidemie 2020 in Deutschland. Vergleichen Sie die Aussage mit den Diagrammen in Abbildung 8.

Aufgabe 2.3.6. Definieren Sie in `Julia` den Vektor

```
x = [1 2 3 4]
```

(1) Was ist der Unterschied zwischen `x` und `vec(x)`?

(2) Erklären Sie die Bedeutung der folgenden Befehle.

```
y = vec(x)
reshape(y, 1, 4)
reshape(y, 2, 2)
[y_i^2 for y_i in y]
map(sqrt, y)
sqrt.(y)
```

(3) Was macht der folgende Code?

```
z = 0
for y_i in y
  z = z + y_i
end
z
```

(4) Vergleichen Sie das Ergebnis aus (c) mit `sum(y)`.

(5) Wir definieren folgende Funktion:

```
function f(n::Int)
    if n <= 0
        @warn "Input $n muss positiv sein."
        return nothing
    elseif n == 1
        return n
    else
        return n * f(n-1)
    end
end
```

Was berechnet `f`? Welche Rolle spielt die Deklaration "`n::Int`"?

2.4 Zufallsvariablen

Zufallsvariablen sind die Entsprechung der Merkmale (Definition 2.1.2) in der Wahrscheinlichkeitstheorie. Sie ermöglichen es uns, Information über Ereignisse als Zahlen darzustellen und somit mathematisch zu analysieren. Im Kontext des maschinellen Lernens repräsentieren Zufallsvariablen die Merkmale, deren Werte von Daten beeinflusst werden und somit Unsicherheit beinhalten.

Definition 2.4.1. Sei $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum. Eine Zufallsvariable ist eine Abbildung

$$X : \Omega \to W,$$

wobei $W \subseteq \mathbb{R}^n$ der Wertebereich/Stichprobenraum ist. Die Zufallsvariable muss $\{\omega \in \Omega \mid X(\omega)_i \leq w_i, i = 1, \ldots, n\} \in \mathcal{A}$ für alle $w \in \mathbb{R}^n$ erfüllen.

Der letzte Teil der Definition stellt sicher, dass die Abbildung X messbar ist, d.h. dass wir Wahrscheinlichkeiten für Ereignisse der Form $X \leq w$ berechnen können und ist essentiell für eine korrekte mathematische Behandlung.

Insbesondere umfasst Definition 2.4.1 auch *multivariate Merkmale*, wie wir sie in Definition 2.1.4 beschrieben haben. Die meisten der folgenden Beispiele sind univariate Zufallsvariablen ($n = 1$). Allerdings wird der Fall $n > 1$ im folgenden Kapitel über das maschinelle Lernen wichtig.

Beispiel 2.4.1. $\Omega = \{(w_1, w_2) \mid 1 \leq w_1, w_2 \leq 6\}$, der Ereignisraum für den zweifachen Würfelwurf. Die Summe der gewürfelten Zahlen wird durch $X : \Omega \to \mathbb{R}$ mit $X(w_1, w_2) = w_1 + w_2$ angegeben.

Dieses Beispiel verdeutlicht, wie eine Zufallsvariable einem zufälligen Ereignis (die Summe der gewürfelten Zahlen) eine numerische Darstellung zuordnet. Die Summe der Augenzahlen ist eine Zufallsvariable, die Werte zwischen 2 und 12 annehmen kann.

Im Folgenden schreiben wir

$$P(X \in A) := P(\{\omega \in \Omega \mid X(\omega) \in A\}).$$

Die Abbildung, die $A \subset W$ auf $P(X \in A)$ abbildet, nennen wir *Verteilungsfunktion* der Zufallsvariable X. Sie beschreibt, wie die Wahrscheinlichkeit über die möglichen Werte der Zufallsvariable verteilt ist. Ist $a \in W$, so schreiben wir auch

$$P(X = a) := P(\{\omega \in \Omega \mid X(\omega) = a\}).$$

Eine weitere Möglichkeit die Verteilung anzugeben ist es, die kumulative Verteilungsfunktion

$$F(x) := P(\{\omega \in \Omega \mid X(\omega)_i \leq x_i, i = 1, \ldots, n\})$$

zu berechnen.

Beispiel 2.4.2. Für den zweifachen (fairen) Münzwurf sei X die Anzahl der geworfenen Zahlen. Die Verteilung von X ist

$$\begin{aligned} P(X = 1) &= P(\{(K, Z), (Z, K)\}) = \frac{1}{2} \quad \text{und} \\ P(X = 2) &= P(\{(Z, Z)\}) = \frac{1}{4}, \\ P(X = 0) &= P(\{(K, K)\}) = \frac{1}{4} \end{aligned}$$

Dieses Beispiel zeigt die Wahrscheinlichkeitsverteilung einer Zufallsvariablen, die endlich viele Werte annehmen kann. Eine solche Zufallsvariable nennen wir daher auch *endliche Zufallsvariable*.

Im Folgenden beschreiben wir wichtige *diskrete* Zufallsvariablen.

Definition 2.4.2. (Endlich/diskret Zufallsvariable) Eine Zufallsvariable X heißt endlich/diskret, wenn $X(\Omega)$ endlich/diskret ist.

Gleichverteilung Für den Wertebereich $W = \{1, \ldots, n\}$ ist die Gleichverteilung gegeben durch

$$P(X = i) = \frac{1}{n}.$$

Notation: $X \sim \text{Unif}(\{1, \ldots, n\})$.

Binomialverteilung Für den Wertebereich $W = \{1, \ldots, n\}$ ist die Binomialverteilung mit Parameter $p \in [0, 1]$ gegeben durch

$$P(X = k) = \binom{n}{k} \cdot p^k \cdot (1 - p)^{n-k}.$$

$P(X = k)$ ist die Wahrscheinlichkeit aus n unabhängigen (siehe Abschnitt 2.5.1) Bernoulli-Experimenten k Erfolge zu erzielen. Wir schreiben $X \sim \text{Bin}(n, p)$

Geometrische Verteilung Für den Wertebereich $W = \{1, 2, 3, \ldots\}$ ist die geometrische Verteilung mit Parameter $p \in [0, 1]$ gegeben durch

$$P(X = k) = (1 - p)^{k-1} \cdot p.$$

Hierbei ist X die Anzahl der Versuche von unabhängigen (siehe Abschnitt 2.5.1) Bernoulli-Experimenten mit Erfolgswahrscheinlichkeit p, bis zum ersten Mal ein Erfolg eintritt. Wir schreiben $X \sim \text{Geom}(p)$.

Als nächstes geben wir wichtige *stetige* Zufallsvariablen an.

Definition 2.4.3. Sei X eine Zufallsvariable. Dann nennen wir X stetig, falls $X(\Omega) \subseteq \mathbb{R}^n$ einen kontinuierlichen (stetigen) Bereich enthält.

Die folgende Definition beinhaltet Integrale in $\mathbb{R}^n$. Die Theorie hinter solchen multivariaten Integralen ist nicht einfach und führt über den Inhalt dieses Abschnitts hinaus. Es reicht aus, sich vorzustellen, dass multivariate Integrale eine Art "Volumen" in höhreren Dimensionen berechnen.

Definition 2.4.4. (Dichte) Sei $X \in \mathbb{R}^n$ eine stetige Zufallsvariable und sei weiterhin $f : \mathbb{R}^n \to [0, \infty)$ eine integrierbare Funktion, sodass $\int_{\mathbb{R}^n} f(x)\, \mathrm{d}x = 1$. Dann nennen wir f eine Dichte von X, falls

$$P(X \in A) = \int_A f(x)\, \mathrm{d}x.$$

Bemerkung: Nicht alle stetigen Zufallsvariablen haben Dichten. Im Folgenden beschränken wir uns auf Zufallsvariablen mit Dichten. Stetige Zufallsvariablen können nur kontinuierlichen Bereichen positive Wahrscheinlichkeiten zuordnen, nicht aber einzelnen Punkten.

Gleichverteilung Eine Zufallsvariable X hat die Gleichverteilung auf $[a, b]$, wenn X die Dichte

$$f(x) = \begin{cases} \frac{1}{b-a} & \text{falls } x \in [a, b] \\ 0 & \text{sonst} \end{cases}$$

hat. Wir schreiben: $X \sim \mathrm{Unif}([a, b])$.

Normalverteilung Eine Zufallsvariable X hat die Normalverteilung auf mit Parametern $\mu \in \mathbb{R}, \sigma^2 > 0$, wenn X die folgende Dichte hat:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

Wir schreiben $X \sim N(\mu, \sigma^2)$. Falls $X \sim N(0, 1)$, heißt X standardnormalverteilt.

2.4.1 Erwartungswert und Varianz

Wie die Lage- und Streuungsparameter im ersten Abschnitt geben Erwartungswert und Varianz einer Zufallsvariable an, wo die Zufallsvariable zu erwarten ist und wie weit sie streut. Der Erwartungswert kann als der Durchschnittswert der Zufallsvariablen interpretiert werden, während die Varianz ein Maß für die Streuung der Werte um den Erwartungswert darstellt.

Im Folgenden sei $X \in \mathbb{R}$ eine univariate Zufallsvariable.

Definition 2.4.5. Sei X eine endliche oder diskrete Zufallsvariable. Der Erwartungswert von X ist

$$E(X) := \sum_{a \in W} a \cdot P(X = a).$$

Ist X eine stetige Zufallsvariable mit Dichte $f(x)$, so ist der Erwartungswert

$$E(X) := \int_{-\infty}^{\infty} x \cdot f(x) \, \mathrm{d}x.$$

Der Erwartungswert ist ein Maß für die zentrale Tendenz einer Zufallsvariablen. Er gibt an, welcher Wert im Durchschnitt zu erwarten ist, wenn man das Experiment wiederholt. Es ist wichtig zu betonen, dass der Erwartungswert nicht unbedingt ein Wert sein muss, der tatsächlich von der Zufallsvariablen angenommen werden kann.

Das folgende Beispiel zeigt, wie der Erwartungswert für eine geometrische Verteilung berechnet werden kann (die geometrische Verteilung beschreibt die Anzahl der Versuche, die benötigt werden, bis ein Erfolg eintritt, und der Erwartungswert gibt an, wie viele Versuche im Durchschnitt benötigt werden).

Beispiel 2.4.3. Sei $X \sim \mathrm{Geo}(p)$. Dann ist

$$E(X) = \sum_{k=1}^{\infty} k \cdot P(X = k) = \sum_{k=1}^{\infty} k \cdot (1-p)^{k-1} \cdot p = p \cdot \frac{\mathrm{d}}{\mathrm{d}p}\Big(- \sum_{k=0}^{\infty} (1-p)^k \Big) = \frac{1}{p}.$$

Dabei haben wir die Formel $\sum_{k=0}^{\infty}(1-p)^k = 1/(1-(1-p)) = 1/p$ verwendet.

Als nächstes listen wir einige Eigenschaften des Erwartungswertes. Diese Eigenschaften sind grundlegend für die Berechnung von Erwartungswerten.

Satz 2.4.1. *(Linearität von Erwartungswerten) Seien X, Y Zufallsvariablen. Dann:*

$$E(aX + bY + c) = a \cdot E(X) + b \cdot E(Y) + c, \quad a, b, c \in \mathbb{R}.$$

Wir illustrieren im nächsten Beispiel wie die Additivitätseigenschaft des Erwartungswertes ($E(X+Y) = E(X) + E(Y)$) verwendet werden kann, um den Erwartungswert einer komplexen Zufallsvariable zu berechnen.

Beispiel 2.4.4. (Pokémon - Sammelkarten) Wie viele Pokémon-Karten muss ein:e Sammler:in im Mittel kaufen, um eine Serie von $n = 150$ Karten zu erhalten? Um diese Frage zu beantworten treffen wir zwei (nicht realistische) Annahmen: (1) Die Karten werden einzeln gekauft; (2) Jede Karte hat die gleiche Wahrscheinlichkeit gezogen zu werden. Die allgemeine Form dieses Problems wurde bereits 1930 von Pólya beschrieben. [21]. In unserem Fall ist der Ereignisraum $\Omega = \{(w_1, w_2, w_3, \ldots) \mid 1 \leq w_i \leq n\}$, wobei w_n die Karte angibt, die im n-ten Schritt erworben wurde.

Wir definieren

$X_i :=$ Anzahl Karten die gekauft werden müssen, um eine neue Karte zu erhalten, nachdem bereits $(i-1)$ verschiedene Karten gezogen wurden.

Mit Hilfe des Urnenmodells erhalten wir

$$P(X_i = k) = \left(\frac{i-1}{n}\right)^{k-1} \cdot \frac{n-(i-1)}{n};$$

d.h. X ist geometrisch verteilt mit Parameter $p = \frac{n-(i-1)}{n}$.

Die Zufallsvariable $X := X_1 + X_2 + \cdots + X_n$ gibt dann die Anzahl der Karten an, die gezogen werden müssen, um alle n Karten zu bekommen. Es gilt daher

$$\begin{aligned} E(X) = E(X_1 + \cdots + X_n) &= E(X_1) + \cdots + E(X_n) \\ &= \sum_{i=1}^{n} \frac{n}{n-i+1}, \qquad \text{(nach Beispiel 2.4.3)} \\ &= \frac{n}{1} + \frac{n}{2} + \cdots + \frac{n}{n} = n \cdot \sum_{i=1}^{n} \frac{1}{i} =: n \cdot H_n. \end{aligned}$$

Für $n = 150$ haben wir $H_n = 5.6$. D.h. man muss 5.6 mal mehr Karten kaufen, als es Karten insgesamt gibt, um alle $n = 150$ zu sammeln.

Als nächstes definieren wir die Varianz einer Zufallsvariablen. Die Varianz ist ein Maß für die Streuung der Werte einer Zufallsvariablen um ihren Erwartungswert. Eine hohe Varianz deutet auf eine große Streuung hin, während eine niedrige Varianz auf eine geringe Streuung hinweist.

Definition 2.4.6. Die Varianz einer Zufallsvariable X ist definiert als

$$\mathrm{Var}(X) := E([X - E(X)]^2).$$

Die Standardabweichung von X ist $\sqrt{\mathrm{Var}(X)}$.

Beispiel 2.4.5. Sei $X \sim N(\mu, \sigma^2)$. Dann ist $E(X) = \mu$, $\mathrm{Var}(X) = \sigma^2$ und die Standardabweichung von X ist σ.

2.4.2 Übungsaufgaben

Aufgabe 2.4.1. Sei X eine diskrete Zufallsvariable mit Werten $\mathbb{N}$. Die Verteilungsfunktion F der Zufallsvariablen X sei folgendermaßen gegeben:

$$F(x) = \begin{cases} 0, & \text{für } x < 1 \\ 1 - \frac{1}{k}, & \text{für } x \in [k, k+1),\ k \in \mathbb{N},\ k \geq 1. \end{cases}$$

(1) Bestimmen Sie folgende Wahrscheinlichkeiten

$$P(X < 2), \quad P(X \leq 2), \quad P(X \leq 3), \quad P(X > 3)$$

(2) Geben Sie die Wahrscheinlichkeiten $P(X = k)$, $k \in \mathbb{N}$ an.

Aufgabe 2.4.2. Eine Klausur hat 4 Aufgaben. Die Zufallsvariable X gibt an, wieviele Aufgaben ein:e zufällig ausgewählte:r Schüler:in richtig gelöst hat. X kann also die Werte 0, 1, 2, 3 oder 4 annehmen. Wir nehmen dabei an, dass jeder Wert mit gleicher Wahrscheinlichkeit angenommen werden kann.

(1) Bestimmen Sie die Wahrscheinlichkeitsfunktion $P(X = x)$ und die Verteilungsfunktion $F(x)$ von X.

(2) Berechnen Sie den Erwartungswert von X.

(3) Berechnen Sie die Varianz von X mittels der Formeln:

(a) $\mathrm{Var}(X) = \mathbb{E}\big((X - \mathbb{E}(X))^2\big)$; (b) $\mathrm{Var}(X) = \mathbb{E}(X^2) - (\mathbb{E}(X))^2$.

Aufgabe 2.4.3. Für diese Aufgaben sollten Sie in `Julia` die Pakete `DataFrames`, `Plots` und `RDatasets` installiert haben.

Rufen Sie den `survey` Datensatz auf:

```
survey = dataset("MASS", "survey")
```

Dieser Datensatz beinhaltet die Antworten von 237 Statistik Studierenden der University of Adelaide zu einigen Fragen. Die Antworten sind, u.A., unter den folgenden Abkürzungen eingetragen.

`Wr.Hnd`: Spannweite der Schreibhand in cm.

`Pulse`: Puls der Studierenden.

`Smoke`: Ob die Studierenden rauchen (Heavy, Regul (regularly), Occas (occasionally), Never).

`Height`: Größe der Studierenden in cm.

(1) Erklären Sie, was die folgenden Befehle bewirken.

```
names(survey)
describe(survey)
```

Insbesondere, was beschreibt die letzte Spalte von `describe(survey)`?

(2) Veranschaulichen Sie in einer Grafik gleichzeitig (1) die Verteilung der Körpergröße der Studierenden und (2) die Verteilung der Körpergröße gegeben `Smoke`.

(3) Veranschaulichen Sie grafisch das multivariate Merkmal `(Wr.Hnd, Pulse)`.

(4) Die empirische Korrelation der Daten $(x_1, y_1), \ldots, (x_N, y_N)$ zweier Merkmale X und Y ist gegeben durch

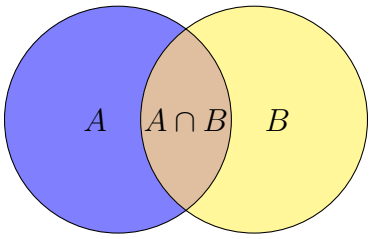
$$\mathrm{Cor} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{(\sum_{i=1}^{N}(x_i - \bar{x})^2) \cdot (\sum_{i=1}^{N}(y_i - \bar{y})^2)}}.$$

Die Korrelation ist ein reeller Wert zwischen -1 und 1. Ist die Korrelation positiv, dann gehen kleine Werte der einen Variable überwiegend einher mit kleinen Werten der anderen Variable und gleichfalls für große Werte. Für eine negative Korrelation ist das genau umgekehrt.

Berechnen Sie die Korrelation der Merkmale aus Aufgabenteil (c) und beurteilen Sie das Ergebnis.

2.5 Der Satz von Bayes

Im maschinellen Lernen wollen wir bei sich ändernder Datenlage verstehen, wie sich Wahrscheinlichkeiten verändern. Die zentrale Frage in diesem Abschnitt ist: Angenommen, wir haben zwei Ereignisse A und B. Wenn wir wissen, dass B eingetreten ist oder eintreten wird, wie ändert sich die Wahrscheinlichkeit, dass A eintritt? Wir nennen dies auch die *Wahrscheinlichkeit von A gegeben B* und schreiben dafür $P(A \mid B)$. Die Wahrscheinlichkeit heißt auch *bedingte Wahrscheinlichkeit*. Dieses Konzept ist grundlegend, um Schlussfolgerungen aus Daten zu ziehen und Modelle an neue Informationen anzupassen. Hier ist ein Venn-Diagramm der Situation:



Definition 2.5.1. Sei $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum und seien $A, B \in \mathcal{A}$ mit $P(B) > 0$. Dann ist

$$P(A \mid B) := \frac{P(A \cap B)}{P(B)}$$

die bedingte Wahrscheinlichkeit von A gegeben B.

Die bedingte Wahrscheinlichkeit kann unsere Einschätzung der Wahrscheinlichkeit eines Ereignisses auf der Grundlage neuer Informationen ändern. Die Bedingung $P(B) > 0$ ist wichtig, um eine Division durch Null zu vermeiden. Wir geben auch eine Definition von bedingten Dichten.

Definition 2.5.2. Es sei $(X, Y) \in \mathbb{R}^n \times \mathbb{R}^m$ eine Zufallsvariable mit Dichte $f_{(X,Y)}$ und $x \in \mathbb{R}^n$. Angenommen X hat Dichte f_X. Die bedingte Dichte von Y gegeben $X = x$ ist

$$f_{Y|X=x}(y) = \frac{f_{(X,Y)}(x, y)}{f_X(x)}.$$

Wir schreiben $(Y \mid X = x)$ für die Zufallsvariable mit dieser Dichte und nennen sie Y gegeben $X = x$.

Beispiel 2.5.1. Würfelwurf: $\Omega = \{1, 2, 3, 4, 5, 6\}$, $A = \{2\}$ (= ein Zwei wird gewürfelt) und $B = \{2, 4, 6\}$ (= eine gerade Zahl wurde gewürfelt). Dann gilt:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} = \frac{1/6}{3/6} = \frac{1}{3}$$

und $P(A) = 1/6$. D.h. falls wir wissen, dass eine gerade Zahl gewürfelt wird, wissen wir, dass wir eher eine 2 werfen, als wenn wir diese Information nicht hätten.

Dieser einfache Würfelwurf verdeutlicht, wie die bedingte Wahrscheinlichkeit unsere ursprüngliche Einschätzung der Wahrscheinlichkeit verändert, wenn wir zusätzliche Informationen erhalten.

Der Bayes'sche Wahrscheinlichkeitsbegriff hat seinen Namen aus folgendem Satz.

Satz 2.5.1. *(Satz von Bayes') Sei $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum und $A, B \in \mathcal{A}$. Dann gilt:*

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}.$$

Der Satz von Bayes stellt eine fundamentale Beziehung zwischen der Wahrscheinlichkeit von A gegeben B und der Wahrscheinlichkeit von B gegeben A her. In Anwendungen des maschinellen Lernens ist oft B das Auftreten neuer Daten und A sind die Parameter eines Modells. Um zu beurteilen, ob die Parameter im Rahmen der neuen Datenlage gut gewählt sind, können wir mit Hilfe des Satzes von Bayes die Rollen von A und B vertauschen und die Wahrscheinlichkeit der Daten B gegeben die Parameter A optimieren. Dies ist die Grundlage vieler Bayes'scher Lernverfahren.

Weiterhin erhalten wir

$$\frac{P(A \mid B)}{P(A)} = \frac{P(B \mid A)}{P(B)}.$$

D.h., falls B das Auftreten von A wahrscheinlicher macht ($\frac{P(A|B)}{P(A)} > 1$), dann macht auch A das Auftreten von B wahrscheinlicher. Dieses Verhältnis wird auch als Likelihood-Ratio bezeichnet.

Es gibt auch einen Satz von Bayes für Zufallsvariablen mit Dichten.

Definition 2.5.3. (Satz von Bayes' für Dichten) Seien $X \in \mathbb{R}^n$ und $Y \in \mathbb{R}^m$ Zufallsvariablen mit Dichten f_X und f_Y. Zudem bezeichne $f_{Y|X=x}$ die Dichte von Y gegeben $X = x$. Dann gilt

$$f_{Y|X=x}(y) = f_{X|Y=y}(x) \cdot \frac{f_Y(y)}{f_X(x)}.$$

2.5.1 Stochastische Unabhängigkeit

Als Nächstes diskutieren wir das Konzept der stochastischen Unabhängigkeit. Zwei Ereignisse A, B sollen unabhängig sein, wenn das Ereignis B die Wahrscheinlichkeit von A nicht beeinflusst; d.h. $P(A \mid B) = P(A)$.

Definition 2.5.4. Sei $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum und $A, B \in \mathcal{A}$. Dann nennen wir A und B stochastisch unabhängig, wenn

$$P(A \mid B) = P(A).$$

Da $P(A \mid B) = \frac{P(A \cap B)}{P(B)}$, gilt stochastische Unabhängigkeit von A und B genau dann, wenn

$$P(A \cap B) = P(A) \cdot P(B).$$

Wenn Ereignisse unabhängig sind, können wir die Wahrscheinlichkeit ihres gemeinsamen Eintretens einfach als das Produkt ihrer individuellen Wahrscheinlichkeiten berechnen.

Beispiel 2.5.2. 2-facher Würfelwurf: $\Omega = \{(w_1, w_2) \mid 1 \leq w_1, w_2 \leq 6\}$ und $A = \{\text{im ersten Wurf eine 2}\}, B = \{\text{im zweiten Wurf eine 2}\}$ Falls die Würfe unabhängig sind, dann sind A und B stochastisch unabhängig. Dann gilt

$$\begin{aligned} P\{\text{es werden 2 Zweien geworfen}\} &= P(A \cap B) \\ &= P(A) \cdot P(B) \\ &= \frac{1}{6} \cdot \frac{1}{6} = \frac{1}{36}. \end{aligned}$$

Wir erweitern die Definition auf mehr als zwei Ereignisse.

Definition 2.5.5. Sei $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum und $A_1, \ldots, A_n \in \mathcal{A}$ Ereignisse. Dann heißen die A_i

(1) paarweise (stochastisch) unabhängig, falls $P(A_i \cap A_j) = P(A_i) \cdot P(A_j)$ für alle $i \neq j$ gilt.

(2) gemeinsam (stochastisch) unabhängig, falls

$$P(A_1 \cap \ldots \cap A_k) = \prod_{i=1}^{k} P(A_i)$$

für alle Wahlen von Indizes $1 \leq i_1 < \cdots < i_k \leq n$.

(Es gilt: (2) impliziert (1), aber im Allgemeinen gilt die Rückrichtung nicht).

Beispiel 2.5.3. Ein zentrales Beispiel im Kontext von Unabhängigkeit ist die *Binomialverteilung*. Hierbei modellieren wir n unabhängige Zufallsexperimente $w_1, \ldots, w_n$ mit Ausgang in $\{0, 1\}$. D.h., $w_i = 0$ oder $w_i = 1$. Für die Binomialverteilung nimmt man an, dass $P(\{w_i = 0\}) = p$ und $P(\{w_i = 1\}) = 1 - p$, dass also die Wahrscheinlichkeit, dass 0 eintritt, für alle Experimente gleich ist. Wegen der stochastischen Unabhängigkeit gilt dann

$$P((w_1, \ldots, w_n)) = p^k \cdot (1-p)^{n-k}, \qquad k = \#\{i \mid w_i = 0\}.$$

Insbesondere gilt:

$$P(\{\text{genau k der } w_i \text{ sind } 0\}) = \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k},$$

da $\binom{n}{k}$ die Anzahl der k-elementigen Teilmengen in einer Menge mit n Elementen angibt. Diese Verteilung nennen wir die Binomialverteilung.

Die Definition von Unabhängigkeit überträgt sich direkt auf Zufallsvariablen.

Definition 2.5.6. Die Zufallsvariablen $X_1, \ldots, X_n$ heißen (stochastisch) unabhängig, falls für alle Intervalle $A_1, \ldots, A_n \subseteq W$ gilt:

$$P(X_1 \in A_1, X_2 \in A_2, \ldots, X_n \in A_n) = \prod_{i=1}^{n} P(X_i \in A_i).$$

2.5.2 Übungsaufgaben

Aufgabe 2.5.1. Ein E-Mail-Anbieter möchte zum Schutz seiner Kund:innen einen Spam-Filter anbieten. Es gibt zwei Merkmale (Merkmal 1 und Merkmal 2), mit denen Spam-Mail identifiziert werden. Damit können die Mails in drei Gruppen eingeteilt werden:

Gruppe 1: Mails mit Merkmal 1
Gruppe 2: Mails mit Merkmal 2 und ohne Merkmal 1
Gruppe 3: Mails ohne die Merkmale 1 und 2

(d.h. in Gruppe 1 sind sowohl die Mails mit Merkmal 1 und nicht 2, als auch die mit 1 und 2).

Die Anteile der drei Gruppen am Gesamtmailaufkommen und die Spam-Mail-Quoten (Anteil der Spam-Mails in der jeweiligen Gruppe) sind in der folgenden Tabelle zu finden:

Gruppe	Anteil an den Mails	Spam-Mail-Quote
1	5%	95%
2	15%	75%
3	80%	20%

(1) Übersetzen Sie die sechs Prozentzahlen der Tabelle in Wahrscheinlichkeiten von Ereignissen.

(2) Wie groß ist die Wahrscheinlichkeit, dass eine Mail eine Spam-Mail ist?

(3) Gegeben Sei eine Spam-Mail. Wie groß ist die Wahrscheinlichkeit, dass sie identifiziert wird?

Aufgabe 2.5.2. Ein Unternehmen sammelt Kundendaten aus drei verschiedenen Quellen: Online-Formulare, mobile Apps und Social Media. 50% der Daten stammen aus Online-Formularen, und jeweils 25% aus mobilen Apps und Social Media. Die Daten aus jeder Quelle haben unterschiedliche Fehlerraten: 1% der Daten aus Online-Formularen sind fehlerhaft, 2% aus mobilen Apps und 4% aus Social Media.

(1) Übersetzen Sie die gegebenen Prozentsätze in (bedingte) Wahrscheinlichkeiten für geeignet gewählte Ereignisse.

(2) Bestimmen Sie die Wahrscheinlichkeit, dass ein zufällig ausgewähltes Datenelement fehlerhaft ist.

Hinweis: Verwenden Sie den Satz von der totalen Wahrscheinlichkeit für drei Ereignisse: Sind A_1, A_2, A_3 drei Ereignisse, so dass

$$A_1 \cap A_2 = A_1 \cap A_3 = A_2 \cap A_3 = \emptyset \quad \text{and} \quad B \subset (A_1 \cup A_2 \cup A_3),$$

dann gilt

$$P(B) = P(B \mid A_1) \cdot P(A_1) + P(B \mid A_2) \cdot P(A_2) + P(B \mid A_3) \cdot P(A_3).$$

(3) Wir wählen zufällig ein Datenelement aus und stellen fest, dass es fehlerhaft ist. Bestimmen Sie die Wahrscheinlichkeit, dass dieses Datenelement aus der mobilen App stammt. Verwenden Sie den Satz von Bayes.

2.6 Lineare Algebra

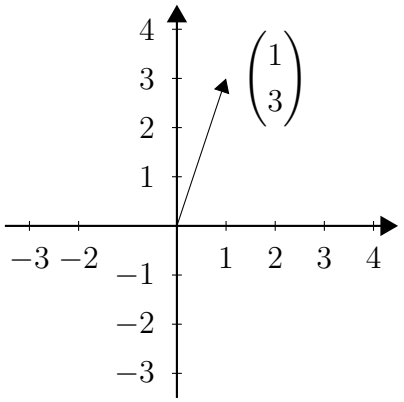
Die Algorithmen des maschinellen Lernens und der KI basieren im Kern auf Methoden der linearen Algebra. Stellen wir uns Daten als große Tabellen vor, in denen jede Zeile einen einzelnen Datenpunkt repräsentiert und jede Spalte ein Merkmal dieses Datenpunkts beschreibt, so lassen sich diese Tabellen als Matrizen – ein zentrales Objekt der linearen Algebra – verstehen. Die Interpretation von Daten als Matrix eröffnet uns Möglichkeiten wie Matrixmultiplikation, um Daten zu manipulieren. Algorithmen wie z.B. neuronale Netze verwenden Matrizenmultiplikation, um diese Daten zu transformieren und Merkmale zu extrahieren, die für die Datenanalyse relevant sind. Dies ist die Motivation, uns in diesem Kapitel mit den grundlegenden Konzepten der linearen Algebra vertraut zu machen. Die lineare Algebra ist somit nicht nur ein abstraktes mathematisches Gebiet, sondern bildet die Grundlage für die Methoden des maschinellen Lernens.

2.6.1 Vektoren

Vektoren sind Listen von reellen Zahlen. Z.B. ist

$$\begin{pmatrix} 1 \\ 3 \end{pmatrix}$$

ein Vektor mit den zwei Einträgen 1 und 3. Wir sagen der Vektor hat die Länge 2, weil er zwei Einträge hat. Geometrisch kann man sich einen Vektor als einen Pfeil im Raum vorstellen, dessen Länge und Richtung durch die Zahlen in der Liste bestimmt werden. Das nächste Bild zeigt den Vektor in der Kartesischen Ebene.



Die Menge aller Vektoren der Länge n bezeichnen wir mit

$$\mathbb{R}^n = \left\{ \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \;\middle|\; u_1, \ldots, u_n \in \mathbb{R} \right\}.$$

Dies ist ein sogenannter Vektorraum. Wir können Vektoren addieren und sie mit einer skalaren Zahl $a \in \mathbb{R}$ multiplizieren, d.h.,

$$u = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix}, \; v = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}; \quad u + v := \begin{pmatrix} u_1 + v_1 \\ u_2 + v_2 \\ \vdots \\ u_n + v_n \end{pmatrix}, \quad a \cdot u := \begin{pmatrix} au_1 \\ au_2 \\ \vdots \\ au_n \end{pmatrix}.$$

Vektoraddition und Skalarmultiplikation ist also komponentenweise definiert.

Beispiel 2.6.1. Es ist

$$\begin{pmatrix} 5 \\ 6 \\ -1 \end{pmatrix} + \begin{pmatrix} 2 \\ -2 \\ 4 \end{pmatrix} = \begin{pmatrix} 5+2 \\ 6-2 \\ -1+4 \end{pmatrix} = \begin{pmatrix} 7 \\ 4 \\ 3 \end{pmatrix}.$$

Die Addition von Vektoren ist assoziativ und kommutativ. Für $u, v \in \mathbb{R}^n$ gilt:

$$u + v = v + u, \quad (u + v) + w = u + (v + w).$$

Außerdem hat Vektoraddition ein neutrales Element und inverse Elemente:

$$o = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}, \quad -u = \begin{pmatrix} -u_1 \\ \vdots \\ -u_n \end{pmatrix}.$$

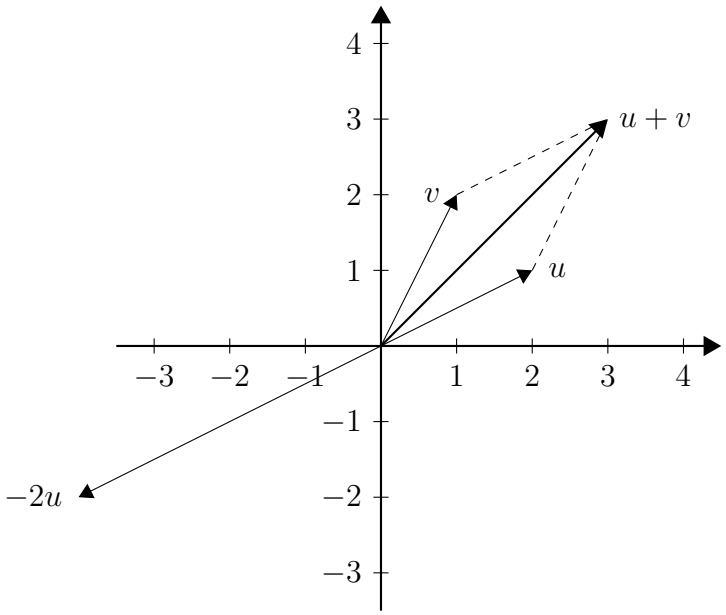
Das bedeutet

$$u + o = o + u = u$$

und

$$u + (-u) = o.$$

Vektoraddition und Skalarmultiplikation kann geometrisch interpretiert werden:



Vektoraddition legt die einzelnen Vektoren hintereinander und verbindet sie, während Skalarmultiplikation Vektoren streckt oder staucht.

Vektorarithmetik – wie Arithmetik in den reellen Zahlen – erfüllt gewisse Regeln.

Satz 2.6.1. *Für alle $a, b \in \mathbb{R}$ und $u, v \in \mathbb{R}^n$ erfüllt die skalare Multiplikation*

(1) $(a + b) \cdot v = a \cdot v + b \cdot v$,
(2) $a \cdot (u + v) = a \cdot u + a \cdot v$,
(3) $(a \cdot b) \cdot v = a \cdot (b \cdot v)$,
(4) $1 \cdot v = v$.

Definition 2.6.1. Für Vektoren $u \in \mathbb{R}^n$ mit Einträgen u_i und $v \in \mathbb{R}^n$ mit Einträgen v_i definieren wir das *Skalarprodukt* als

$$\langle u,\ v\rangle := \sum_{i=1}^{n} u_i \cdot v_i.$$

Wir sagen, dass u orthogonal zu v steht, falls $\langle u, v\rangle = 0$. Die Norm ist

$$\|u\| := \sqrt{\langle u, u\rangle}.$$

Beispiel 2.6.2. Seien

$$u = \begin{pmatrix} 5 \\ 6 \\ -1 \end{pmatrix}, \quad v = \begin{pmatrix} 2 \\ -2 \\ 4 \end{pmatrix}.$$

Dann ist $\langle u, v\rangle = 5 \cdot 2 + 6 \cdot (-2) + (-1) \cdot 4 = -6$. Die Normen der zwei Vektoren sind

$$\|u\| = \sqrt{\langle u, u\rangle} = \sqrt{5^2 + 6^2 + (-1)^2} = \sqrt{62}, \qquad \|v\| = \sqrt{2^2 + (-2)^2 + 4^2} = \sqrt{24}.$$

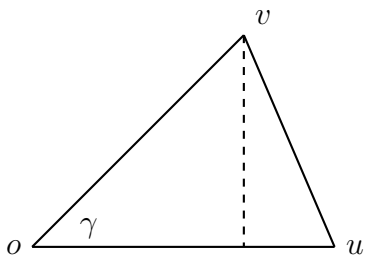
Es gilt für $a, b, u, v \in \mathbb{R}^n$ und $s, t \in \mathbb{R}$:

$$\langle a + tb, u + sv\rangle = \langle a, u\rangle + t\langle b, u\rangle + s\langle a, v\rangle + st\langle b, v\rangle.$$

Die Norm $\|u\|$ von $u \in \mathbb{R}^n$ misst die Länge eines Vektors. Die Bedeutung des Skalarprodukts zwischen u und v ist

$$\cos(\text{Winkel zwischen } u \text{ und } v) = \frac{\langle u, v\rangle}{\|u\| \cdot \|v\|}. \tag{2.1}$$

Die Gleichung in (2.1) können wir wie folgt beweisen: Sei γ der Winkel zwischen u und v:



Der Abstand von u nach v erfüllt, laut Kosinussatz, die Gleichung

$$\|u - v\|^2 = \|u\|^2 + \|v\|^2 - 2\|u\| \cdot \|v\| \cdot \cos(\gamma).$$

Andererseits gilt

$$\|u - v\|^2 = \langle u - v, u - v\rangle = \langle u, u\rangle - 2\langle u, v\rangle + \langle v, v\rangle = \|u\|^2 + \|v\|^2 - 2\langle u, v\rangle.$$

Vergleichen wir diese zwei Gleichungen erhalten wir (2.1).

Im maschinellen Lernen wird der Ausdruck $\langle u, v\rangle/(\|u\| \cdot \|v\|)$ auch *cosine similarity* genannt. Die Ähnlichkeit zweier (Daten-)Vektoren wird durch den Winkel zwischen ihnen gemessen. Dementsprechend heißt

$$d(u, v) = 1 - \frac{\langle u, v\rangle}{\|u\| \cdot \|v\|}$$

Kosinus-Abstand zwischen u und v.

2.6.2 Matrizen

Matrizen werden in vielen Bereichen des maschinellen Lernens verwendet, um Daten darzustellen und zu manipulieren. Beispielsweise können Bilder als Matrizen von Pixelwerten dargestellt werden, und Algorithmen für die Bilderkennung verwenden oft Matrizenoperationen.

Eine $m \times n$-Matrix ist eine rechteckige Anordnung oder Tabelle reeller Zahlen a_{ij}:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}.$$

Wir schreiben

$$A = (a_{ij}) \in \mathbb{R}^{m \times n}$$

und nennen a_{ij} die Einträge, m die Anzahl der Zeilen und n die Anzahl der Spalten der Matrix. Wenn $m = n$, wird die Matrix als quadratisch bezeichnet. Eine $(1, n)$-Matrix ist ein Zeilenvektor, eine $(m, 1)$-Matrix ist ein Spaltenvektor. Zwei Matrizen sind gleich, wenn alle Einträge gleich sind.

Die Matrix A hat n *Spaltenvektoren*

$$a_j := \begin{pmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{pmatrix}, \quad 1 \leq j \leq n$$

und sie hat m *Zeilenvektoren*

$$\alpha_i := \begin{pmatrix} a_{i1} \\ a_{i2} \\ \vdots \\ a_{in} \end{pmatrix}, \quad 1 \leq i \leq m$$

Wir schreiben auch

$$\begin{aligned} A &= \begin{pmatrix} | & & | \\ a_1 & \dots & a_n \\ | & & | \end{pmatrix} \\ &= \begin{pmatrix} - & \alpha_1 & - \\ & \vdots & \\ - & \alpha_m & - \end{pmatrix} \end{aligned} \tag{2.2}$$

Ähnlich wie bei Vektoren definieren wir die Addition für Matrizen $A = (a_{ij})$ und $B = (b_{ij}) \in \mathbb{R}^{m\times n}$ durch

$$A + B = \begin{pmatrix} a_{11} + b_{11} & \dots & a_{1n} + b_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & \dots & a_{mn} + b_{mn} \end{pmatrix} = \begin{pmatrix} | & & | \\ a_1 + b_1 & \dots & a_n + b_n \\ | & & | \end{pmatrix}.$$

Ebenso die skalare Multiplikation mir $c \in \mathbb{R}$:

$$cA = \begin{pmatrix} ca_{11} & \dots & ca_{1n} \\ \vdots & \ddots & \vdots \\ ca_{m1} & \dots & ca_{mn} \end{pmatrix} = \begin{pmatrix} | & & | \\ ca_1 & \dots & ca_n \\ | & & | \end{pmatrix},$$

wobei a_i und b_j die Spaltenvektoren von A und B sind. Diese Operationen sind assoziativ, kommutativ und distributiv, sodass der Raum der reellen $m \times n$ Matrizen auch ein Vektorraum ist.

2.6.3 Matrix-Vektor und Matrix-Matrix Produkt

Im letzten Abschnitt haben wir die Addition von Matrizen definiert. Jetzt definieren wir das Produkt zweier Matrizen. Beachte, dass nicht alle Matrizen miteinander multipliziert werden können; die Seitenlängen müssen kompatibel sein.

Beispielsweise basieren neuronale Netze im Kern auf sequentiellen Matrixmultiplikationen. Dies werden wir im nächsten Kapitel im Detail ausführen. In diese Abschnitt lernen wir zunächst die abstrakt mathematische Definition kennen. Es ist dabei aber wichtig zu betonen, dass diese abstrakte Definition vielen konkreten Anwendungen zu Grunde liegt.

Definition 2.6.2. Sei $A = (a_{ij}) \in \mathbb{R}^{m\times r}$ und $B = (b_{k\ell}) \in \mathbb{R}^{r\times n}$. Dann definieren wir das Matrixprodukt $C = (c_{i\ell}) := AB \in \mathbb{R}^{m\times n}$ durch

$$c_{i\ell} = \sum_{s=0}^{r} a_{is} \cdot b_{s\ell}, \quad 1 \le i \le m,\ 1 \le j \le n.$$

Dies ist die Definition der Matrixmultiplikation auf der Ebene der einzelnen Einträge. Wir entwickeln im weiteren Verlauf dieses Abschnitts einige Konzepte, die

uns helfen werden, die Multiplikation einfacher und übersichtlicher darzustellen. Zunächst aber ein Beispiel.

Beispiel 2.6.3. $\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 4 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 2 & -1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 5 & 2 \\ 4 & 5 \end{pmatrix}.$

Definition 2.6.3. Sei $A = (a_{ij}) \in \mathbb{R}^{m \times n}$ eine Matrix, dann ändert die *transponierte Matrix* $A^\top$ Zeilen mit Spalten, d.h.,

$$A^\top = \begin{pmatrix} a_{11} & a_{21} & \dots & a_{n1} \\ a_{12} & a_{22} & \dots & a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1m} & a_{2m} & \dots & a_{nm} \end{pmatrix} \in \mathbb{R}^{n \times m}.$$

Es gilt dabei

$$(A^\top)^\top = A \quad \text{und} \quad (A+B)^\top = A^\top + B^\top.$$

In Zeilen- und Spaltenvektornotation können wir die transponierte Matrix wie folgt schreiben:

$$\begin{pmatrix} | & & | \\ a_1 & \dots & a_n \\ | & & | \end{pmatrix}^\top = \begin{pmatrix} - & a_1 & - \\ & \vdots & \\ - & a_n & - \end{pmatrix}.$$

Das heißt, das Transponieren einer Matrix verwandelt Spalten in Zeilenvektoren und umgekehrt.

Für Matrizen

$$S = \begin{pmatrix} - & s_1 & - \\ & \vdots & \\ - & s_m & - \end{pmatrix} \in \mathbb{R}^{m \times r}, \quad T = \begin{pmatrix} | & & | \\ t_1 & \dots & t_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{r \times n}$$

kann das Matrixprodukt geschrieben werden als

$$ST = \begin{pmatrix} \langle s_1,\, t_1 \rangle & \dots & \langle s_1,\, t_n \rangle \\ \vdots & \ddots & \vdots \\ \langle s_m,\, t_1 \rangle & \dots & \langle s_m,\, t_n \rangle \end{pmatrix} \in \mathbb{R}^{m \times n}. \tag{2.3}$$

Satz 2.6.2. *Seien A, B, C Matrizen, so dass die folgenden Formel definiert sind. Dann gilt*

(1) $(AB)C = A(BC)$ *(Assoziativität),*

(2) $(A + B)C = AC + BC$ *und* $A(B + C) = AB + AC$ *(Distributivität),*

(3) $(aA)B = a(AB) = A(aB)$ *(Kompatibilität mit der Skalarmultiplikation).*

Die transponierte Matrix eines Produkts ist das Produkt der transponierten Matrizen, aber die Reihenfolge kehrt sich um (beachte, dass das Matrix Produkt nicht kommutativ ist, die Reihenfolge zählt!): $(AB)^\top = B^\top A^\top$.

Die *Identitätsmatrix* ist

$$I := \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix} \in \mathbb{R}^{n \times n}$$

Wir haben für alle $A \in \mathbb{R}^{m \times n}$ und $B \in \mathbb{R}^{n \times k}$:

$$AI = A, \qquad IB = B.$$

Ein wichtiger Spezialfall ist Matrix-Vektor Multiplikation. Sind

$$A = \begin{pmatrix} - & a_1 & - \\ & \vdots & \\ - & a_m & - \end{pmatrix} \in \mathbb{R}^{m \times n} \quad \text{und} \quad u = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \in \mathbb{R}^n,$$

so ist das Produkt Au definiert durch

$$Au = \begin{pmatrix} \langle a_1, u \rangle \\ \vdots \\ \langle a_m, u \rangle \end{pmatrix} \in \mathbb{R}^m.$$

Dies bedeutet, dass das Ergebnis der Multiplikation einer $m \times n$-Matrix mit einem n-Vektor ein Vektor im $\mathbb{R}^m$ ist, dessen i-te Komponente das Skalarprodukt des i-ten Zeilenvektors von A mit dem Vektor u ist. Z.B. gilt immer

$$Iu = u.$$

Beispiel 2.6.4. Das Produkt einer 2×3-Matrix mit einem Vektor der Länge 3 ist ein Vektor in $\mathbb{R}^2$:

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 4 \end{pmatrix} \begin{pmatrix} -2 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ -3 \end{pmatrix}.$$

Die erste Komponente des Ergebnisvektors ist

$$1 \cdot (-2) + 2 \cdot 1 + 3 \cdot 0 = 0$$

und die zweite Komponente ist

$$2 \cdot (-2) + 1 \cdot 1 + 4 \cdot 0 = -3.$$

Jede Matrix $A \in \mathbb{R}^{m \times n}$ definiert somit eine Abbildung

$$\phi_A : \mathbb{R}^n \to \mathbb{R}^m, \quad u \mapsto Au. \tag{2.4}$$

Diese Art von Abbildungen nennen wir *lineare Abbildungen*. Lineare Abbildungen spielen eine zentrale Rolle in der Mathematik, da sie wichtige Eigenschaften wie die Erhaltung von Vektoraddition und Skalarmultiplikation besitzen. Dies bedeutet, dass für alle Vektoren $u, v \in \mathbb{R}^n$ und alle Skalare $t \in \mathbb{R}$ gilt:

$$\phi_A(u + tv) = \phi_A(u) + t\phi_A(v), \tag{2.5}$$

oder äquivalent,

$$A(u + tv) = Au + tAv.$$

Diese Eigenschaft ist essentiell z.B. in der Sprachverarbeitung, wo Wörter oder Sätze als Vektoren dargestellt werden und lineare Abbildungen verwendet werden, da dadurch semantische Beziehungen erhalten bleiben. Weitere Anwendungen finden sich in der Bildverarbeitung, der Datenkompression und vielen anderen Bereichen des maschinellen Lernens. Das Verständnis linearer Abbildungen ist daher ein Schlüssel zum Verständnis vieler Algorithmen und Techniken im Bereich der künstlichen Intelligenz.

2.6.4 Übungsaufgaben

Aufgabe 2.6.1. Berechnen Sie die folgenden Vektor- und Matrixoperationen (wenn möglich):

(1) $2\begin{pmatrix}1\\2\\4\end{pmatrix}+5\begin{pmatrix}-2\\3\\0\end{pmatrix}$;

(2) $2\begin{pmatrix}0\\5\\-6\end{pmatrix}+3\begin{pmatrix}7\\-1\\3\end{pmatrix}-4\begin{pmatrix}-5\\-3\\9\end{pmatrix}$;

(3) $\begin{pmatrix}2&3&15\\12&-3&8\\-9&-26&6\end{pmatrix}+2\begin{pmatrix}-11&13&2\\7&5&-21\\16&2&-24\end{pmatrix}$;

(4) $3\begin{pmatrix}1&2\\5&7\\-11&6\end{pmatrix}+2\begin{pmatrix}-5&11&13\\-8&17&-27\\-2&12&4\end{pmatrix}$.

Aufgabe 2.6.2. Seien

$$A=\begin{pmatrix}0&1&-2\\4&-1&2\\3&0&-7\end{pmatrix},B=\begin{pmatrix}0&-3&4\\0&6&-8\\2&5&6\end{pmatrix},C=\begin{pmatrix}1&1&1\\2&4&-1\\3&5&0\end{pmatrix},I=\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}.$$

Berechnen Sie X.

(1) $X=A+B$.

(2) $X=A-B$.

(3) $X=3B-2C$.

(4) $X=AB$.

(5) $X=AB-BA$.

(6) $C=A+B-X$

(7) $X=A^\top B+I$.

(8) $A=AB-X^\top$.

(9) $X=BCB^\top$.

(10) $X=A^\top A-B^\top B$.

Aufgabe 2.6.3. Berechnen Sie die folgenden Matrixprodukte:

(1) $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \cdot \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$;

(2) $\begin{pmatrix} 5 & 3 & -2 \\ 1 & 4 & 1 \end{pmatrix} \cdot \begin{pmatrix} -2 & 1 \\ 5 & 0 \\ 3 & 3 \end{pmatrix}$;

(3) $\begin{pmatrix} 2 \\ 1 \\ -7 \end{pmatrix} \cdot \begin{pmatrix} 1 & -1 & 3 \end{pmatrix}$;

(4) $\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 2 & 1 \end{pmatrix} \cdot \begin{pmatrix} -1 & 2 & -2 \\ 2 & -3 & 1 \\ 0 & 1 & -1 \end{pmatrix}$.

Aufgabe 2.6.4. Seien $a = (1, -4, 0, 6)$, $b = (0, 3, 7, -2)$ und $c = (5, 0, -3, 2)$. Berechnen Sie x.

(1) $x = 2a + 5b$.

(2) $x = a - 2b + c$.

(3) $x = a^\top - b^\top + 5c^\top$.

(4) $x = ab^\top + 2$.

(5) $a + c = 2b - x$.

Aufgabe 2.6.5. Gegeben sei die folgende Tabelle mit den Noten von 8 Studierenden in einem Kurs.

Initialen	Note
AB	1
CD	4
EF	3
GH	2
KL	2
NO	4
RS	1
TU	3

(1) Um welche Art von Merkmalen handelt es sich hier?

(2) Stellen Sie die Daten durch eine reelle Matrix $A \in \mathbb{R}^{n \times m}$ dar. Wie lauten n und m?

3 Grundlagen der Künstlichen Intelligenz (KI) und des Maschinellen Lernens (ML)

3.1 Begriffsklärung

In diesem Abschnitt werden die grundlegenden Konzepte der Künstlichen Intelligenz (KI), des Maschinellen Lernens (ML) und des Deep Learning (DL) erläutert und ihre Zusammenhänge aufgezeigt. Jeder Begriff wird detailliert betrachtet und anhand von Beispielen veranschaulicht. Im Kern lässt sich feststellen, dass

$$\text{KI} \supset \text{ML} \supset \text{DL}.$$

Dies bedeutet, dass Deep Learning eine spezielle Form des Maschinellen Lernens darstellt, und Maschinelles Lernen wiederum eine spezielle Form der Künstlichen Intelligenz. KI ist das umfassendste Gebiet, ML ist darin enthalten, und DL ist ein Teil des ML. Jeder Bereich baut auf den Erkenntnissen und Techniken des vorherigen auf, verfolgt aber unterschiedliche Ansätze, um das Ziel intelligenter Maschinen zu erreichen.

3.1.1 Künstliche Intelligenz (KI)

Stellen Sie sich vor, Sie versuchen, ein komplexes Spiel zu meistern, einen Text in eine andere Sprache zu übersetzen oder ein Fahrzeug zu steuern. All diese Aufgaben erfordern Intelligenz. Die KI ist der Versuch, diese Intelligenz in Maschinen abzubilden.

Rich [22] definierte künstliche Intelligenz wie folgt:

> "Artificial Intelligence is the study of how to make computers do things at which, at the moment, people are better."

Künstliche Intelligenz (KI) zielt darauf ab, Maschinen zu entwickeln, die Aufgaben bewältigen können, die menschliche Intelligenz erfordern. Dazu gehören Fähigkeiten wie Problemlösung, logisches Denken, Entscheidungsfindung, Spracherkennung, visuelle Wahrnehmung (das Erkennen von Objekten in Bildern), das Verstehen und Erzeugen von Sprache sowie das Lernen aus Erfahrungen. Das Bestreben, Intelligenz zu schaffen, ist nicht neu, hat aber mit dem Aufkommen moderner, leistungsstarker Computer in den letzten Jahrzehnten erheblich an Dynamik gewonnen.

Frühe Ansätze in der Informatik versuchten dies durch direkte Programmierung zu erreichen. Dabei gaben Programmierer der Maschine eine umfangreiche Sammlung von Regeln vor, die sie befolgen sollte. Das Problem bestand darin, dass diese Regeln oft unflexibel waren und sich nur schwer auf komplexe, reale Probleme anwenden ließen. Ein klassisches Beispiel ist ein Schachprogramm. Um gut zu spielen, mussten frühe Schachprogramme mit Hunderttausenden von Regeln programmiert werden, die alle denkbaren Spielsituationen abdecken sollten. Dies war sehr aufwendig und konnte nie vollständig sein. Ein weiterer Nachteil war die Schwierigkeit, solche Systeme zu warten und zu erweitern, da jede neue Regel unerwünschte Nebeneffekte haben und das System destabilisieren konnte. Eine bessere Strategie ist es, dem System selbst zu ermöglichen, aus Daten zu lernen und sich so flexibel an neue Situationen anzupassen.

Der Kern moderner KI-Systeme ist daher Software, die große Datenmengen verarbeitet, um Lösungen zu generieren. Dabei werden Muster in den Daten gesucht, um Vorhersagen für komplexe Situationen zu treffen. Dies erfordert die Anwendung verschiedener mathematischer Methoden aus der linearen Algebra, der Wahrscheinlichkeitstheorie und der Statistik. Dazu müssen die Daten in Form von Zahlen vorliegen; sie müssen also *digitalisiert* sein. Viele Daten, wie z.B. Bilder, Töne oder Wörter, lassen sich leicht digitalisieren. Bei anderen Daten hingegen, z.B. Gefühle oder Sinneswahrnehmungen, ist nicht unbedingt klar, wie sie zu digitalisieren sind.

Daher übersteigt die menschliche Intelligenz die Fähigkeiten aktueller KI-Systeme,

die hauptsächlich auf Mustererkennung basieren. So umfasst menschliche Intelligenz auch die körperlich-kinästhetische Intelligenz (die Fähigkeit, den eigenen Körper zu steuern und einzusetzen), die interpersonelle Intelligenz (die Fähigkeit, Stimmungen, Absichten und Motivationen anderer Menschen zu erkennen und zu verstehen), sowie die existenzielle Intelligenz (ermöglicht, über Fragen der eigenen Existenz und des Sinns des Lebens zu reflektieren) [13]. Diese Aspekte von Intelligenz sind für aktuelle KI-Systeme schwer zu erfassen.

3.1.2 Maschinelles Lernen (ML)

Maschinelles Lernen (ML) ist der Teilbereich der KI, der den Ansatz verfolgt, Maschinen aus Daten *lernen* zu lassen, anstatt sie mit expliziten Regeln auszustatten. Das bedeutet, dass dem System eine große Menge an Daten präsentiert wird, und es selbstständig Muster in diesen Daten erkennt. Diese Mustererkennung dient als Grundlage für das Treffen von Vorhersagen. Ein grundlegender Vorteil dieses Ansatzes ist, dass das System seine Leistung im Laufe der Zeit durch die Verarbeitung zusätzlicher Daten verbessern kann.

Stellen Sie sich zum Beispiel vor, Sie möchten einen Algorithmus entwickeln, der E-Mails automatisch als Spam oder Nicht-Spam einordnet. Anstatt manuell Regeln zu definieren (z.B.: E-Mails mit den Wörtern 'Angebot' oder 'Gewinn' sind Spam), geben Sie dem Algorithmus einen Datensatz mit Tausenden von E-Mails, die bereits als Spam oder Nicht-Spam markiert sind. Der Algorithmus analysiert diese Daten, erkennt die Muster und lernt, welche Merkmale einer E-Mail typisch für Spam sind (z.B. bestimmte Wörter, der Absender, die Häufigkeit bestimmter Zeichen). Je mehr E-Mails der Algorithmus analysiert, desto besser wird er in der Spam-Erkennung. Dabei ist dem/der Programmierer:in oft nicht klar, welche spezifischen Regeln das System verwendet, um Spam zu erkennen. Das System hat seine eigenen Regeln aus den Daten abgeleitet.

ML ist somit ein datenbasierter Ansatz für KI. Er ist besonders nützlich für Probleme, bei denen es schwierig oder unmöglich ist, explizite Regeln zu definieren. Eine ausführliche Abhandlung der Mathematik hinter dem maschinellen Lernen ist [11].

3.1.3 Deep Learning (DL)

Deep Learning (DL) ist ein Teilbereich des Maschinellen Lernens, der sich auf eine bestimmte Art von System zur Mustererkennung konzentriert, nämlich auf sogenannte "tiefe neuronale Netze" (deep neural networks). Die Struktur dieser künstlichen neuronalen Netze ist dabei der Struktur des menschlichen Gehirns nachempfunden. Das Netzwerk besteht aus mehreren Schichten, von denen jede unterschiedliche Merkmale aus den Daten extrahiert. Die Kombination dieser Merkmale ermöglicht es dem Netzwerk, komplexe Zusammenhänge zu erkennen. Wir werden künstliche neuronale Netze im Detail in Abschnitt 3.3 behandeln.

Deep Learning hat in den letzten Jahren enorme Fortschritte gemacht, insbesondere in Bereichen wie Bilderkennung, Spracherkennung und natürliche Sprachverarbeitung. Dies liegt daran, dass DL-Algorithmen besonders gut darin sind, komplexe Zusammenhänge in Daten zu erfassen, die für traditionelle ML-Algorithmen schwer zu entdecken sind. Denken Sie an die Gesichtserkennung auf Ihrem Smartphone, die automatische Übersetzung von Texten oder die Fähigkeit von Sprachassistenten wie Siri oder Alexa, Fragen zu beantworten. Diese Anwendungen basieren in der Regel auf Deep Learning.

Eine ausführliche Referenz für Deep Learning ist [29].

3.2 Was bedeutet Lernen?

Im vorherigen Abschnitt haben wir maschinelles Lernen als einen Ansatz der künstlichen Intelligenz beschrieben, der es Maschinen ermöglicht, aus Daten zu lernen. Was bedeutet Lernen aber in diesem Kontext? Menschen können auf viele verschiedene Arten lernen. Es ist ein komplexer Prozess mit unterschiedlichen Lernstilen, unter anderem durch Kombinationen von Vester's [26] Lerntypen Hören, Sehen, Lesen und Fühlen/Tasten. Im maschinellen Lernen bezeichnet Lernen jedoch eine spezifische Methode, um Informationen aus Daten zu extrahieren. In diesem Abschnitt wollen wir mathematisch beschreiben, was dieses Lernen bedeutet. Dazu müssen wir zunächst klären, wie wir Information, beziehungsweise Muster in Daten, mathematisch formalisieren können. Buckley [10] beschreibt Information als Beziehung zwischen Datensätzen:

> "Information is [...] a relationship between sets or ensembles of structured variety."

Daraus folgt, dass wir Information mathematisch als eine Teilmenge beschreiben können:

$$I \subset E \times A, \tag{3.1}$$

wobei E eine Menge von Eingaben und A eine Menge möglicher Ausgaben ist. Dann bedeutet $(x, y) \in I$, dass die Eingabe $x \in E$ und die Ausgabe $y \in A$ in Beziehung zueinander stehen; y ist eine mögliche Ausgabe für x. Beispielsweise könnte x ein Bild einer Katze darstellen, während y das Wort "Katze" selbst ist. In diesem Beispiel bedeutet $(x, y) \in I$, dass x tatsächlich ein Bild mit Beschreibung y ist. Wenn y' hingegen das Wort 'Hund' ist, dann gilt $(x, y') \notin I$. Andere Beispiele sind Sensordaten (E) und zugehörige Zustände (A) oder Text (E) und zugehörige Kategorien (A). Das Ziel des maschinellen Lernens ist es, eine mathematische oder statistische Beschreibung von I zu finden.

3.2.1 Modelle

Der folgende Abschnitt basiert lose auf [9] und [11].

Wir wollen nun mathematische Modelle entwickeln, um die Menge I in (3.1) beschreiben zu können. Wir werden dazu zwei Ansätze kennen lernen: das *deterministische Modell* und das *statistische Modell*. Dazu werden wir im Folgenden annehmen, dass

$$E = \mathbb{R}^d, \quad \text{und} \quad A = \mathbb{R}^n.$$

Das heißt, die Eingabedaten sind durch d reelle Zahlen und die Ausgabedaten durch n reelle Zahlen bestimmt. Die Ein- und Ausgabedaten liegen somit in digitalisierter Form vor. Dies bedeutet, dass wir Daten, wie beispielsweise Bilder oder Temperaturen, in Zahlen umwandeln, die ein Computer verarbeiten kann.

Das deterministische Modell beschreibt die Menge I als den Graphen einer Abbildung $f : \mathbb{R}^d \to \mathbb{R}^n$; d.h. $I = \{(x, y) \mid y = f(x)\}$. Gesucht wird dann die Abbildung f. Da die Menge aller Abbildungen jedoch zu groß ist, um damit effektiv zu rechnen, schränkt man sich bei der Modellwahl auf Abbildungen f_θ ein, die von endlich vielen Parametern θ abhängen.

Beispiel 3.2.1. Sei $d = n = 1$ und $f_\theta(x) = ax + b$. Dies ist das sogenannte *lineare Modell*. f_θ hängt von den zwei Parametern $\theta = (a, b)$ ab. Hierbei sind a und b reelle Zahlen, die das Verhalten der Funktion bestimmen. Durch Anpassung dieser Parameter können wir die Funktion an unsere Daten anpassen.

Das deterministische Modell beschreibt eine feste Beziehung zwischen Ein- und Ausgabe. Das statistische Modell hingegen beschreibt den Zusammenhang zwischen Ein- und Ausgabe mittels Zufallsvariablen $X \in \mathbb{R}^d$ und $Y \in \mathbb{R}^n$ und einer Wahrscheinlichkeitsfunktion oder –dichte $P_\theta(y \mid x)$ für die Zufallsvariable Y gegeben $X = x$ (siehe Definition 2.5.2). Die Dichte soll wieder von endlich vielen Parametern θ abhängen. Sie beschreibt, wie wahrscheinlich eine bestimmte Ausgabe y ist, wenn die Eingabe x gegeben ist. In diesem Fall wäre $I = \{(x, y) \mid P_\theta(y \mid x) > 0\}$ die Menge aller Paare, die eine positive Dichte aufweisen.

Beispiel 3.2.2. (1) Sei $d = n = 1$ und $P_\theta(y \mid x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y-x)^2}$. Dann ist y eine normalverteilte Zufallsvariable mit Mittelwert x (siehe Abschnitt 2.4). Der Parameter ist hier die Varianz $\theta = \sigma^2$. Diese Verteilung beschreibt, dass die Ausgabe y wahrscheinlich in der Nähe des Wertes x liegt, wobei σ^2 die Streuung der Werte angibt.

(2) Sei $d = 3, n = 1$. D.h., $x = (x_1, x_2, x_3) \in \mathbb{R}^3, y \in \mathbb{R}$. Wir nehmen an, dass y nur 3 Zustände annehmen kann, etwa $y \in \{1, 2, 3\}$. Dies ist der Fall in Kategorisierungsproblemen, wenn wir dem Inputdatum x eine Kategorie y zuordnen wollen. $P_\theta(i \mid x) = \exp(x_i)/(\exp(x_1) + \exp(x_2) + \exp(x_3))$ ist ein Beispiel für eine Wahrscheinlichkeitsverteilung auf $\{1, 2, 3\}$. Die Wahrscheinlichkeit $P_\theta(i \mid x)$ wird auch *SoftMax* genannt; siehe Definition **??**. SoftMax wird insbesondere im Kapitel über Large Language Models (Kapitel 4) eine zentrale Rolle spielen.

Wir fassen zusammen.

Definition 3.2.1. Sei $\mathbb{R}^d$ die Menge der Eingaben und $\mathbb{R}^n$ die Menge der Ausgaben. Die Anzahl der Parameter sei p.

(1) Ein deterministisches Modell ist eine Funktion $f_\theta : \mathbb{R}^d \to \mathbb{R}^n$, die von Parametern $\theta \in \mathbb{R}^p$ abhängt.

(2) Ein statistisches Modell ist eine Wahrscheinlichkeitsverteilung für Y gegeben $X = x$ mit Verteilung $P_\theta(y \mid x)$, die von Parametern $\theta \in \mathbb{R}^p$ abhängt.

3.2.2 Training

Wir nehmen nun an, dass wir ein Modell (Definition 3.2.1) mit Parametern $\theta \in \mathbb{R}^p$ ausgewählt haben und dass wir N Datenpaare

$$D = \{(x_1, y_1), \ldots, (x_N, y_N)\} \subset \mathbb{R}^d \times \mathbb{R}^n$$

gegeben haben. Die einzelnen Einträge der Daten bezeichnen wir in diesem Abschnitt mit $x_i^{(j)}$ bzw. $y_i^{(j)}$; d.h., $x_i^{(j)}$ ist der j Eintrag von x_i und genauso für y_i.

Beispiel 3.2.3. Stellen wir uns folgende $N = 4$ Daten $(x, y) \in \mathbb{R}^3 \times \mathbb{R}$ vor:

$x^{(1)}$ = Abschluss	$x^{(2)}$ = Wohnort	$x^{(3)}$ = Alter	y = Jahreseinkommen
`MSc`	`Osnabrück`	36	60.145
`PhD`	`Osnabrück`	24	72.541
`BSc`	`Hannover`	31	58.901
`MSc`	`Bremen`	29	61.005

Diese Daten sind noch nicht digitalisiert. Wir können sie z.B. digitalisieren, indem wir dem Abschluss einen numerischen Wert zuordnen (Bsc = 1, Msc = 2, PhD = 3) und die Stadt durch ihre geographischen Koordinaten (Breiten- und Längengrad) ersetzen.

Im Kontext des maschinellen Lernens bezeichnen wir:

- x_i als Eingabedaten oder Attribute.
- y_i als Labels, Ausgabevariablen oder Responsevariablen.

Variablen, die jeden Wert innerhalb eines Bereichs annehmen können, werden kontinuierliche Variablen genannt. Variablen, die nur bestimmte Werte annehmen können, werden diskrete Variablen oder kategorische Variablen genannt.

Training bezeichnet den Prozess, mit Hilfe der Daten einen Parameter θ zu finden, so dass das resultierende Modell die Daten gut beschreibt. Diesen Prozess nennt man das Lernen des Parameters θ. Damit haben wir die Frage zu Beginn dieses Abschnitts beantwortet: Im maschinellen Lernen bedeutet Lernen das Finden eines Parameters anhand von Daten.

Um einen Parameter zu finden, der die Muster in den Daten gut beschreibt, müssen wir definieren, was "gut" in diesem Fall bedeutet. Angenommen wir haben uns für ein deterministisches Modell f_θ entschieden. Im optimalen Fall haben wir dann einen Parameter gefunden, so dass $f(x) = y$ für *alle* neuen Datenpunkte (x, y). In der Praxis sind Daten oft ungenau oder verrauscht. Daher ist es unrealistisch zu erwarten, dass $f(x)$ genau y ergibt. Stattdessen verwenden wir Annäherungen, um flexibler zu sein. Was "nahe an" genau bedeutet, hängt vom Problem ab und wird üblicherweise mithilfe einer *Verlustfunktion*

$$\ell : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$$

gemessen. Dabei misst $\ell(y, \hat{y})$ den Fehler zwischen einem tatsächlichen Wert y und der Vorhersage $\hat{y}$, die unser Modell für y nach Eingabe von x getroffen hat. Eine übliche Wahl ist z.B. die Norm $\ell(y, \hat{y}) = \|y - \hat{y}\|$ (siehe Definition 2.6.1) oder die quadrierte Norm

$$\ell(y, \hat{y}) = \|y - \hat{y}\|^2.$$

Je kleiner der Wert der Verlustfunktion, desto besser passt das Modell zu den Daten.

Im vorherigen Abschnitt haben wir nun zum ersten mal einen wichtigen konzeptuellen Schritt gemacht:

> Wir interpretieren Datenpunkte als Vektoren im $\mathbb{R}^n$ und somit als geometrische Objekte! Dies eröffnet uns die Möglichkeit, Daten mittels geometrischer Methoden zu manipulieren und somit Informationen aus ihnen zu erhalten.

Idealerweise wählen wir ein Modell und lernen einen Parameter, der auch für unbekannte Daten gute Vorhersagen liefert; d.h., dass für jeden neuen Datenpunkt (x, y) und Vorhersage $\hat{y}$ für y der Verlust $\ell(y, \hat{y})$ klein ist. Deshalb teilen wir die Daten in Trainingsdaten und Testdaten auf. Die Trainingsdaten verwenden wir, um den Parameter zu lernen, und die Testdaten, um zu prüfen, wie gut unser Modell auf unbekannten Daten funktioniert. Um die Daten in Trainings- und Testdaten aufzuteilen, können wir jedem Datenpunkt zufällig ein Label zuweisen. Üblicherweise verwenden wir zwischen 50% und 90% der Daten für das Training. Die zufällige Aufteilung in Trainings- und Testdaten sollte unabhängig von der Modellauswahl sein.

Nachdem wir ein Modell ausgewählt und die Daten vorbereitet haben, lernen wir die Parameter. Beim deterministischen Modell verwenden wir die *Empirische Risikominimierung (ERM)*. Das bedeutet, wir suchen einen Parameter θ^*, der das empirische Risiko

$$R(\theta) := \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, f_\theta(x_i)).$$

minimiert. Man beachte, dass das empirische Risiko der Mittelwert (Definition 2.2.5) der einzelnen Verluste ist.

Beim statistischen Modell können wir Maximum-Likelihood-Schätzung (MLE) verwenden. Dies entspricht der Maximierung der *Likelihood-Funktion*

$$L(\theta) := \prod_{i=1}^{N} P_\theta(y_i \mid x_i). \tag{3.2}$$

Die Motivation für die Likelihood-Funktion ist die Annahme, dass die Datenpunkte (x_i, y_i) (stochastisch) unabhängig (Definition 2.5.6) gezogen werden. Dann ist $L(\theta)$ die Dichte der multivariablen Zufallsvariable $(y_1|x_1, \ldots, y_N|x_N)$. Alternativ können wir auch die (negative und mit $\frac{1}{N}$ skalierte) *Log-Likelihood-Funktion*

$$l(\theta) := -\frac{1}{N} \sum_{i=1}^{N} \log P_\theta(y_i \mid x_i) \tag{3.3}$$

minimieren, was einfacher ist, da Summen einfacher abzuleiten sind als Produkte.

Zusammenfassend besteht das Training im maschinellen Lernen also aus drei Schritten:

(1) Daten in Trainings- und Testdaten aufteilen

(2) Parameter lernen

(3) Validierung.

Hierbei bedeutet Validierung, dass wir überprüfen, wie gut unser Modell auf den Testdaten funktioniert. Dabei wird üblicherweise das empirische Risiko ausgewertet und geprüft, ob das Risiko der Trainingsdaten ungefähr dem Risiko der Testdaten entspricht. Wenn die Qualität des Modells auf den Trainingsdaten besser ist als auf den Testdaten, spricht man von Überanpassung (oder Overfitting). Das

bedeutet, dass das Modell die Trainingsdaten sehr gut gelernt hat, aber nicht in der Lage ist, allgemeine Vorhersagen zu treffen.

Der Ansatz, den wir in diesem Abschnitt diskutiert haben, versucht, einen bestimmten Parameter θ zu bestimmen. Ein anderer Ansatz ist es, θ selbst als Zufallsvariable zu interpretieren, so dass unser Modell Fluktuationen in θ berücksichtigt. Beispielsweise könnte θ einer Wahrscheinlichkeitsverteilung folgen, die einen Mittelwert hat, den wir bereits beobachtet haben. Die Wahrscheinlichkeitsverteilung für θ nennt man Prior-Verteilung. Die Responsevariable hat dementsprechend eine bedingte Verteilung $(y \mid x, \theta)$. Der Satz von Bayes' erlaubt es uns dann, die Verteilung von θ bei Erhalt neuer Daten zu aktualisieren. Dieser Ansatz wird daher unter dem Namen *Bayesian machine learning* zusammengefasst.

3.2.3 Lernparadigmen

Das im vorherigen Abschnitt vorgestellte Konzept nennt man auch *überwachtes Lernen*, weil jeder Datenpunkt x_i mit einem Label y_i versehen ist. Das Wort "überwacht" bedeutet in diesem Fall so viel wie, dass eine externe Quelle (wie z.B. der/die Datenwissenschaftler:in) für Richtigkeit der Zugehörigkeit von x_i und y_i zuständig ist, sie also *überwacht*. Dabei unterscheidet man oft zwei Haupttypen beim überwachten Lernen:

- Regression: Das Label hat einen kontinuierlichen Wertebereich (z.B. Temperatur).
- Klassifikation: Das Label ist eine Kategorie (z.B. Katze/Hund).

Beispiel 3.2.4. Ein Beispiel für ein Klassifikationsproblem ist Beispiel 2.1.3: Gegeben ein Bild, welches Fashion-Item ist darauf zu sehen?

Angenommen wir haben ein Klassifikationsproblem und k Klassen $\{1, \dots, k\}$. Dann können wir die i-te Klasse, mit dem i-ten Basisvektor e_i identifizieren; d.h., $e_i = (0, \dots, 0, 1, 0, \dots, 0) \in \mathbb{R}^k$ und 1 steht an der i-ten Stelle. Sei nun P_θ ein statistisches Modell, (x, y) ein Datenpunkt und $\hat{y} \in \mathbb{R}^k$ der Vektor dessen i-ter Eintrag $\hat{y}^{(i)}$ gleich $P_\theta(y = e_i \mid x)$ ist. Der Vektor $\hat{y}$ gibt also die Wahrscheinlichkeitsverteilung der k Klassen gegeben x an. Dann gilt $\log \hat{y}^{(i)} = \sum_{i=1}^{k} y^{(i)} \log \hat{y}^{(i)}$,

da ja $y = e_i$ nur einen Eintrag hat, der nicht Null ist, nämlich $y^{(i)}$. Wir setzen nun in die Log-Likelihood Funktion (3.3) ein und erhalten

$$l(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, \hat{y}_i), \qquad \text{wobei } \ell(y, \hat{y}) = -\sum_{i=1}^{k} y^{(i)} \log \hat{y}^{(i)}.$$

Wir geben der Verlustfunktion in dieser Gleichung einen Namen.

Definition 3.2.2. Es seien $y, \hat{y} \in \mathbb{R}^k$ Vektoren, deren Einträge die Wahrscheinlichkeiten von k Klassen angeben. Die Verlustfunktion

$$\ell(y, \hat{y}) = -\sum_{i=1}^{k} y^{(i)} \log \hat{y}^{(i)}$$

heißt *Cross-Entropy*.

Die Definition der Cross-Entropy hat folgenden Hintergrund. Wir betrachten eine Zufallsvariable $\hat{Y}$ auf den k Klassen mit der von unserem Modell berechneten Wahrscheinlichkeitsverteilung $\hat{y} \in \mathbb{R}^k$. In der Informationstheorie wird die Unsicherheit des Ereignisses $\hat{Y} = i$ wird mit $\log(1/(\hat{y}^{(i)})) = -\log(\hat{y}^{(i)})$ modelliert. Das bedeutet, dass je geringer die Wahrscheinlichkeit $\hat{y}^{(i)}$ für das Ereignis $\hat{Y} = i$ ist, desto mehr Unsicherheit enthält es! Wenn der i-te Eintrag von $\hat{y}$ groß ist, ist die Unsicherheit klein, weil wir bereits relativ viel darüber wissen, was passieren wird, da die Eintrittswahrscheinlichkeit ja großist. Umgekehrt, wenn ein Ereignis mit einer niedrigen Wahrscheinlichkeit eintritt, können wir es nur selten beobachten und erhalten daher viel mehr Information, wenn es tatsächlich passiert. Ist nun y die echte Verteilung, die von $\hat{y}$ approximiert wird, so ist die Cross-Entropy $\ell(y, \hat{y}) = \sum_{i=1}^{k} y^{(i)} (-\log \hat{y}^{(i)})$ der Erwartungswert (Definition 2.4.5) der Unsicherheit von $\hat{y}$. Minimierung der Cross-Entropy bedeutet also, die Unsicherheit unseres Modells zu minimieren. In der Informationstheorie wird das Fachwort *Entropie* für Unsicherheit verwendet.

Im Unterschied zum überwachten Lernen gibt es auch das *unüberwachte Lernen*. In diesem Fall haben wir nur die Eingabedaten $x_1, \ldots, x_N \in \mathbb{R}^d$, aber nicht die Labels $y_1, \ldots, y_N \in \mathbb{R}^n$ zur Verfügung, bzw. es ist uns nicht möglich die Labels in angemessener Zeit zu generieren.

Beispiel 3.2.5. Als Beispiel kann man sich einen großen Textdatensatz vorstellen, beispielsweise sämtliche YouTube Kommentare. Es ist unmöglich in einer überwachten Art und Weise allen Kommentaren das Label "Like" oder "Dislike" zuzufügen, da es einfach zu viele dieser Kommentare gibt.

Das Ziel beim unüberwachten Lernen es daher, Muster und Strukturen in den (Eingabe-)Daten ohhne Zugriff auf die Labels zu finden. In obigen Beispiel könnte ein unüberwachter Lernalgorithmus selbstständig anhand der Daten Klassen von YouTube-Kommentaren generieren.

Einige gängige Techniken sind:

- Clustering: Gruppierung ähnlicher Datenpunkte (z.B. YouTube-Kommentare klassifizieren).
- Dimensionsreduktion: Reduzierung der Anzahl der Variablen, ohne wichtige Informationen zu verlieren (z.B. Visualisierung hochdimensionaler Daten).
- Assoziationsanalyse: Finden von Beziehungen zwischen Variablen (z.B. Kunden, die A kaufen, kaufen auch B).

Zuletzt gibt es noch das Konzept des *verstärkenden Lernen* Beim verstärkenden Lernen lernt ein Agent, eine optimale Strategie zu entwickeln, um eine bestimmte Aufgabe zu erfüllen, indem er in einer Umgebung interagiert und dabei lernt (d.h. eine Qualitätsfunktion optimiert). Der Agent lernt durch Versuch und Irrtum. Er probiert verschiedene Aktionen aus und beobachtet die Resultate. Dadurch lernt er im Laufe der Zeit eine optimale Strategie.

Beispiel 3.2.6. Ein klassisches Beispiel ist das Training eines Roboters, der ein Labyrinth navigieren soll. Eine Qualitätsfunktion belohnt direktere Wege und bestraft Fehler wie z.B. gegen die Wand fahren. Der Roboter probiert verschiedene Strategien und optimiert dadurch die Qualitätsfunktion und somit sein Verhalten.

Die hier diskutierte Einordnung verschiedener Lernparadigmen ist aber letztendlich nur eine Orientierung. In der Praxis verschwimmen oft die Grenzen zwischen den drei Paradigmen, oder sie werden miteinander verschaltet. Manchmal ist auch vom *semi-überwachten Lernen* oder *selbst-überwachten Lernen* die Rede.

3.2.4 Übungsaufgaben

Aufgabe 3.2.1. Gegeben seien Eingabedaten $x_1, x_2, x_3 \in \mathbb{R}$ mit Ausgabedaten $y_1, y_2, y_3 \in \mathbb{R}$:

$$(x_1, y_1) = (1, 2),\ (x_2, y_2) = (2, 0),\ (x_3, y_3) = (0, \tfrac{1}{2}).$$

(1) Beschreiben Sie das lineare Modell $f_\theta : \mathbb{R} \to \mathbb{R}$.

(2) Wieviele Parameter werden für das lineare Modell benötigt? Begründen Sie Ihre Antwort.

(3) Beschreiben Sie im linearen Modell das empirische Risiko $R(\theta)$ der Daten, wenn wir als Verlustfunktion den quadratischen Abstand $\ell(y, \hat{y}) = (y - \hat{y})^2$ wählen.

Aufgabe 3.2.2. Gegeben seien Eingabedaten $x_1, x_2, x_3 \in \mathbb{R}$ mit Ausgabedaten $y_1, y_2, y_3 \in \mathbb{R}^2$:

$$x_1 = 1, y_1 = (\tfrac{1}{2}, \tfrac{1}{2}),\ x_2 = 2, y_2 = (\tfrac{1}{4}, \tfrac{3}{4}),\ x_3 = 0, y_3 = (\tfrac{2}{3}, \tfrac{1}{3}).$$

und das Modell

$$f_\theta(x) = \begin{pmatrix} \theta_1 x + \theta_2 \\ \theta_3 x + \theta_4 \end{pmatrix}.$$

(1) Was sind die Parameter in diesem Modell? Begründen Sie Ihre Antwort.

(2) Beschreiben Sie das empirische Risiko $R(\theta)$ der Daten, wenn wir als Verlustfunktion den quadratischen Abstand $\ell(y, \hat{y}) = \|y - \hat{y}\|^2$ wählen.

(3) Beschreiben Sie das empirische Risiko $R(\theta)$ der Daten, wenn wir als Verlustfunktion Cross-Entropy

$$\ell(y, \hat{y}) = y^{(1)} \cdot \log \hat{y}^{(1)} + y^{(2)} \cdot \log \hat{y}^{(2)},$$

wobei $y = (y^{(1)}, y^{(2)})$ und $\hat{y} = (\hat{y}^{(1)}, \hat{y}^{(2)})$, wählen.

(4) Wie klein kann das empirische Risiko in (3) werden? Begründen Sie Ihre Antwort.

Aufgabe 3.2.3. Gegeben seien Eingabedaten $x_1, \ldots, x_4 \in \mathbb{R}^3$ mit Ausgaben $y_1, \ldots, y_4 \in \mathbb{R}$:

$$x_1 = (1, 2, 4), y_1 = 2, \qquad x_2 = (0, 1, 1), y_2 = 0,$$
$$x_3 = (-2, 1, 0), y_3 = \tfrac{1}{2}, \quad x_4 = (3, 2, 0), y_4 = 1.$$

(1) Beschreiben Sie das lineare Modell $f_\theta : \mathbb{R}^3 \to \mathbb{R}$.

(2) Wieviele Parameter werden für das lineare Modell benötigt? Begründen Sie Ihre Antwort.

(3) Beschreiben Sie im linearen Modell das empirische Risiko $R(\theta)$ der Daten, wenn wir als Verlustfunktion den quadratischen Abstand $\ell(y, \hat{y}) = (y - \hat{y})^2$ wählen.

Aufgabe 3.2.4. Es sei wieder $f_\theta : \mathbb{R} \to \mathbb{R}, f_\theta(x) = ax + b, \theta = (a, b)$, das (deterministische) lineare Modell. Wir definieren ein statistisches Modell, indem wir $\sigma^2 > 0$ wählen und

$$y \sim N(f_\theta(x), \sigma^2)$$

setzen. y gegeben x ist also eine normalverteilte Zufallsvariable mit Erwartungswert $f_\theta(x)$ und Varianz σ^2; y streut also zufällig um den Mittelpunkt $f_\theta(x)$. Die Wahrscheinlichkeitsdichte von $y \mid x$ ist dann

$$P_\theta(y \mid x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{1}{2\sigma^2}(y - f_\theta(x))^2}.$$

Gegeben seien wieder die Daten $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ aus Aufgabe 1.

(1) Berechnen Sie die Likelihood-Funktion

$$L(a, b) = L(\theta) = P_\theta(y_1 \mid x_1) \cdot P_\theta(y_2 \mid x_3) \cdot P_\theta(y_3 \mid x_3).$$

(2) Berechnen Sie die Log-Likelihood-Funktion $l(a, b) = \log L(a, b)$.

(3) Finden Sie einen optimalen Parameter θ, indem Sie die partiellen Ableitungen gleich Null setzen: $\frac{\partial}{\partial a} l(a, b) = \frac{\partial}{\partial b} l(a, b) = 0$, und dann nach a und b auflösen. Vergleichen Sie mit Aufgabe 1.

Aufgabe 3.2.5. Es sei $f_\theta : \mathbb{R} \to \mathbb{R}, f_\theta(x) = ax + b, \theta = (a, b)$, das (deterministische) lineare Modell. Gegeben seien die drei Datenpunkte

$$(x_1, y_1) = (0, 0),\ (x_2, y_2) = (1, 1),\ (x_3, y_3) = (-1, 2).$$

Als Verlustfunktion haben wir wieder den quadratischen Abstand $\ell(y, \hat{y}) = (y - \hat{y})^2$.

(1) Berechnen Sie das empirische Risiko $R(\theta) = R(a, b)$ bzgl. der Daten.

(2) Finden Sie einen optimalen Parameter θ, indem Sie die partiellen Ableitungen gleich Null setzen: $\frac{\partial}{\partial a} R(a, b) = \frac{\partial}{\partial b} R(a, b) = 0$, und dann nach a und b auflösen.

Wir wollen nun im Rest der Aufgabe geometrisch beschreiben, was in (b) passiert.

(3) Zeigen Sie, dass

$$R(a, b) = \frac{1}{3} \|X\theta - y\|^2,$$

wobei $\|\cdot\|$ die Norm von Vektoren aus Definition 2.6.1 ist und

$$X = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \end{pmatrix} \in \mathbb{R}^{3\times 2}, \quad y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \in \mathbb{R}^3, \quad \theta = \begin{pmatrix} a \\ b \end{pmatrix} \in \mathbb{R}^2.$$

(X heißt in diesem Kontext *Datenmatrix*).

(4) Beachte, dass $X\theta \in \mathbb{R}^3$ ein Punkt im drei-dimensionalen ist. Was ist die Menge aller dieser Punkte

$$H = \{X\theta \mid \theta \in \mathbb{R}^2\} \subset \mathbb{R}^3$$

für ein geometrisches Objekt? Zeichnen Sie es im Raum. Zeichnen Sie auch y ein.

(5) Überlegen Sie, dass $R(a, b)$ minimiert wird, wenn der Abstand von y zu H minimal wird.

(6) Berechnen Sie den Punkt $p \in H$, der am nächsten an y liegt, der also den Abstand $\|p - y\|$ minimiert. (*Hinweis:* Lot fällen!)

(7) Lösen Sie das Gleichungssystem $X\theta = p$ und vergleichen Sie mit (b).

(8) Was passiert, wenn wir $N \geq 4$ Datenpunkte haben?

Aufgabe 3.2.6. Laden Sie das dritte Jupyter-Notebook herunter und führen es aus.

(1) Ersetzen Sie den `cars` Datensatz, durch den `GAGurine` Datensatz:

```
data_GAG = dataset("MASS", "GAGUrine");
```

(2) Lesen Sie die Dokumentation, um zu verstehen, was die zwei Merkmale `Age` und `GAG` im Datensatz bedeuten.

Es seien nun $(x_1, y_1), \ldots, (x_N, y_N)$ die Datenpunkte, wobei x `Age` angibt und y `GAG` angibt.

(3) Wie gut beschreiben die drei Modelle im Notebook die Daten?

(4) Entwickeln Sie ein neues Modell, das die Daten besser beschreiben kann. Es ist dazu hilfreich, die Daten zu visualisieren, um zu schauen, welcher Funktionsgraph passen könnte. Visualieren Sie auch den transformierten Datensatz, den wir erhalten, wenn wir die y_i logarithmieren: $\{(x_i, \log(y_i)) \mid 1 \leq i \leq N\}$.

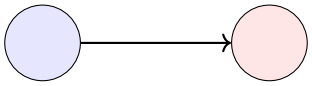
3.3 Neuronale Netze

Eines der wichtigsten, wenn nicht *das* wichtigste Modell im maschinellen Lernen ist das künstliche neuronale Netz. Im Abschnitt 3.2.1 haben wir ein mathematisches Konzept für Modelle entwickelt und neuronale Netze sind ein Spezialfall davon. Künstliche neuronale Netze sind der Funktionsweise biologischer Gehirne nachempfunden. Daher ist es sinnvoll sich kurz mit der Biologie neuronaler Aktivität zu befassen, bevor wir neuronale Netze mathematisch beschreiben werden.

3.3.1 Wie funktionieren biologische Neuronen?

Ein menschliches Gehirn umfasst schätzungsweise 10 bis 100 Milliarden Nervenzellen, auch Neuronen genannt. Diese Anzahl an Einheiten bildet die Grundlage für die Komplexität unseres Denkens und Handelns. Jedes dieser Neuronen ist ein hochspezialisierter Zelltyp, der in der Lage ist, Informationen zu empfangen, zu verarbeiten und weiterzuleiten.

Ein Neuron besteht im Wesentlichen aus drei Hauptteilen: einem Zellkörper, einem Axon und den Dendriten. Der Zellkörper kann vereinfacht als eine Art "Akkumulator" betrachtet werden, der elektrische Spannungen speichert. Diese Spannung wird durch eingehende Impulse anderer Neuronen aufgeladen. Dabei werden diese Impulse über die Dendriten empfagen. Je mehr Impulse ankommen, desto höher wird die Spannung im Zellkörper. Überschreitet die Spannung einen bestimmten Schwellenwert, wird ein elektrisches Signal ausgelöst und über das Axon weitergeleitet. Die Verbindung zwischen dem Axon eines Neurons und dem Dendriten eines anderen Neurons wird als Synapse bezeichnet. Neuronen kommunizieren also nicht direkt miteinander, sondern über diese spezialisierten Kontaktstellen. Wir können dies abstrakt visuell wie folgt darstellen:



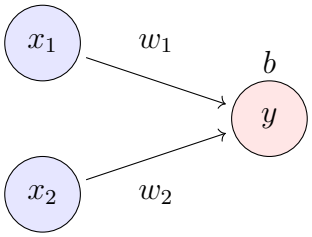
Die beiden Kreise stellen die Neuronen dar, und der Pfeil symbolisiert die Richtung der Informationsübertragung von einem Neuron zum anderen.

Synapsen bestehen aus einem kleinen Spalt, der von Neurotransmittern gefüllt ist. Neurotransmitter sind chemische Botenstoffe, die das Signal von einem Neuron zum nächsten übertragen. Die Stärke einer Synapse – ihre Leitfähigkeit oder Wirksamkeit – ist nicht konstant, sondern dynamisch und verändert sich abhängig von der Aktivität. Wird eine Synapse häufig genutzt, indem regelmäßig Signale übertragen werden, wird sie gestärkt. Umgekehrt schwächt sich eine Synapse ab, wenn sie wenig bis gar nicht genutzt wird. Diese Veränderung der synaptischen Stärke ist die biologische Grundlage für Lernen und Gedächtnisbildung. Durch die Verstärkung bestimmter synaptischer Verbindungen und die Abschwächung anderer kann das Gehirn seine Struktur und Funktion an neue Erfahrungen anpassen.

3.3.2 Das McCulloch-Pitts Neuron

Die Idee, dass Nervenzellen für Wahrnehmung, Denken und Lernen verantwortlich sind, setzte sich Anfang des 20. Jahrhunderts durch und führte 1943 zu ersten mathematischen Modellen des Neurons durch McCulloch und Pitts [18]. Ein Neuron wird dabei durch zwei reelle Werte y und b dargestellt; y gibt die elektrische Spannung und b den Schwellenwert an, ab dem das Neuron aktiv wird. Ein einfaches

Netzwerk mit drei Neuronen, von denen zwei Information zum Dritten leiten, stellt sich visuell wie folgt dar.



Hierbei bezeichnen x_1 und x_2 die Spannungen der Neuronen auf der linken Seite und w_1 und w_2 bezeichnet die Stärke der neuronalen Verbindungen. Das Signal, welches zum Neuron auf der rechten Seite geleitet wird ist dann $w_1x_1 + w_2x_2$. Das Neuron auf der rechten Seite wird jedoch nur aktiviert, wenn ein Schwellenwert b überschritten wird. Im McCulloch-Pitts Neuron stellt sich dies dann wie folgt dar:

$$y \text{ ist } \begin{cases} \text{aktiv}, & \text{falls } w_1x_1 + w_2x_2 - b > 0 \\ \text{inaktiv}, & \text{falls } w_1x_1 + w_2x_2 - b \leq 0 \end{cases}.$$

Wir können dies in kompakter Art und Weise mit Hilfe einer sogenannten *Aktivierungsfunktion* σ darstellen:

$$y = \sigma(w_1x_1 + w_2x_2 - b).$$

Z.B. ist die ReLU (Rectified Linear Unit) Aktivierungsfunktion gegeben durch

$$\sigma(z) = \max\{0, z\}.$$

Hierbei bedeutet $y = 0$ eben, dass das Neuron auf der rechten Seite inaktiv bleibt. Insgesamt erhalten wir ein *Modell*

$$f_\theta : \mathbb{R}^2 \to \mathbb{R},\ (x_1, x_2) \mapsto y$$

mit Parametern $\theta = (w_1, w_2, b)$.

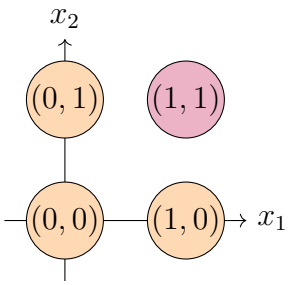
Wir diskutieren weitere Wahlen von Aktivierungsfunktionen in Abschnitt 3.3.4. Für den Moment bleiben wir bei der ReLU Aktivierungsfunktion. Mit dieser Wahl

ist das McCulloch-Pitts Neuron bereits relativ ausdrucksstark, wie das folgende Beispiel zeigt.

Beispiel 3.3.1. Wir verwenden das McCulloch-Pitts Neuron zur Klassifizierung von Daten in $\mathbb{R}^2$. Dabei klassifizieren wir Daten anhand der Zugehörigkeit zu einer der zwei Gruppen

$$\text{Gruppe } 1 = \{(x_1, x_2) \in \mathbb{R}^2 \mid f_\theta(x_1, x_2) > 0\},$$
$$\text{Gruppe } 2 = \{(x_1, x_2) \in \mathbb{R}^2 \mid f_\theta(x_1, x_2) = 0\}.$$

Angenommen haben die vier Datenpunkte $D = \{(0,0),\ (1,0),\ (0,1),\ (1,1)\} \subset \mathbb{R}^2$. Wir wollen die Punkte $(0,0)$, $(1,0)$ und $(0,1)$ der Gruppe 1 (orange) und den Punkt $(1,1)$ der Gruppe 2 (lila) zuordnen:

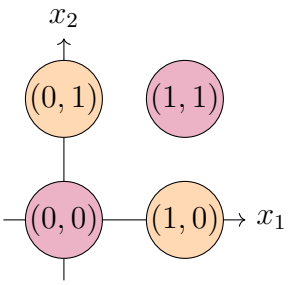


Wir können dies mit dem McCulloch-Pitts Neuron mit Parametern $\theta = (2, 2, 3)$ erreichen. Dann ist

$$f_\theta(x_1, x_2) = \sigma(2 \cdot x_1 + 2 \cdot x_2 - 3) = \max\{0,\ 2(x_1 + x_2) - 3\}.$$

Also $f_\theta(0,0) = -3$, $f_\theta(1,0) = f_\theta(0,1) = -1$ und $f_\theta(1,1) = 1$.

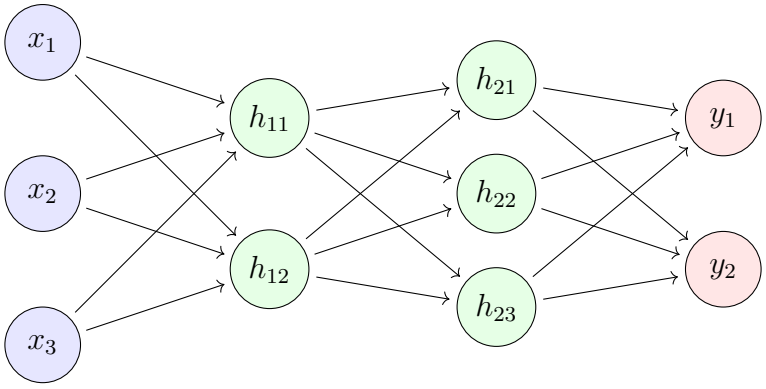
Hier ist eine wichtige Beobachtung: Wir können einen Parameter θ finden, der $(0,0)$, $(1,0)$ und $(0,1)$ einer Gruppe und den Punkt $(1,1)$ einer anderen Gruppe zuordnet, weil sich die Punkte $(0,0)$, $(1,0)$ und $(0,1)$ durch eine Gerade von dem Punkt $(1,1)$ trennen lassen. Die *Entscheidungsgrenze* ist der Ort, an dem sich die zwei Gruppen treffen, also dort, wo $x_1 + x_2 = \frac{3}{2}$ gilt. Dies ist die Gleichung einer Gerade. Daher kann das McCulloch-Pitts Neuron nur Daten klassifizieren, die durch eine Gerade trennbar sind. Z.B. lässt sich folgende Klassifizierung eben nicht durch ein McCulloch-Pitts Neuron realisieren:



Das Beispiel zeigt, dass die Ausdrucksstärke des McCulloch-Pitts Neurons begrenzt ist. Wir können nur Daten klassifizieren, die durch eine Gerade trennbar sind. Dies motiviert die Entwicklung komplexerer Modelle, die in der Lage sind, auch nicht-linear trennbare Daten zu klassifizieren. Eine Möglichkeit, dies zu erreichen, ist die Verwendung von *mehrschichtigen neuronalen Netzen*, die im Grunde eine Verschaltung von McCulloch-Pitts Neuronen sind und die im nächsten Abschnitt vorgestellt werden.

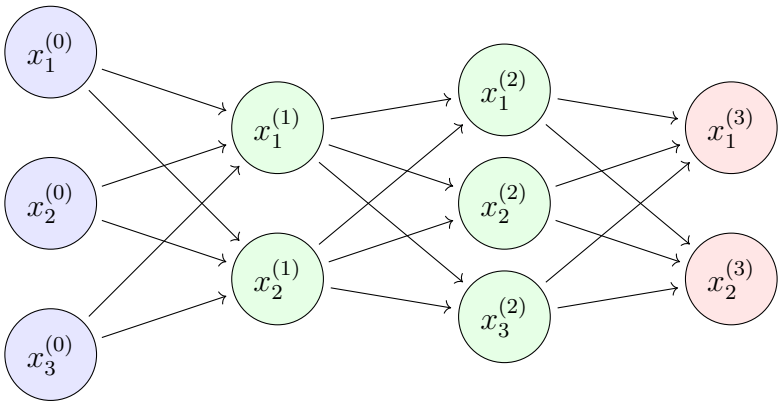
3.3.3 Struktur eines künstlichen neuronalen Netzes

Ein neuronales Netz ist ein Rechenmodell, das von der Struktur des menschlichen Gehirns inspiriert ist. Es besteht aus miteinander verschalteten McCulloch-Pitts Neuronen, die in *Schichten* organisiert sind. Hier ist ein Beispiel für ein einfaches neuronales Netz mit vier Schichten:



Dabei heißt die linke Schicht *Eingabeschicht*, die rechte Schicht *Ausgabeschicht* und die dazwischen liegenden Schichten *versteckte Schichten* (*hidden layers*). Jede Schicht besteht aus mehreren Neuronen, die miteinander verbunden sind. Wir

bennen die Neuronen nun um, da wir bei der folgenden Herleitungen unnötige Fallunterscheidungen vermeiden wollen.



Wie im McCulloch-Pitts Neuron haben die Verbindungen zwischen den Neuronen haben unterschiedliche Stärken, die durch *Gewichte* dargestellt werden. Außerdem hat jedes Neuron auch einen Bias-Term $b_j^{(l)}$, der den Schwellenwert des Neurons in der l-ten Schicht angibt. In der obigen Abbildung sind die Gewichte und Schwellenwerte nicht explizit dargestellt, aber jeder Pfeil erhält ein Gewicht und jedes Neuron einen Schwellenwert. Wir nummerieren diese wie folgt:

- $w_{ij}^{(l)}$ ist das Gewicht der Verbindung vom j-ten Neuron der $(l-1)$-ten Schicht zum i-ten Neuron der l-ten Schicht.
- $b_i^{(l)}$ ist der Bias-Term des i-ten Neurons in der l-ten Schicht.

Z.B. ist $w_{21}^{(1)}$ das Gewicht von $x_1^{(0)}$ zu $x_2^{(1)}$ und $w_{12}^{(2)}$ das Gewicht von $x_2^{(1)}$ zu $x_1^{(2)}$. Mit dieser Notation ist das von Neuron $x_1^{(2)}$ empfangene Signal

$$x_1^{(2)} = \sigma\Big(w_{11}^{(2)} x_1^{(1)} + w_{12}^{(2)} x_2^{(1)} - b_1^{(2)}\Big).$$

Allgemein ist das von Neuron i in Schicht l empfangene Signal

$$x_i^{(l)} = \sigma\bigg(\sum_{j=1}^{n_{l-1}} w_{ij}^{(l)} x_j^{(l-1)} - b_i^{(l)}\bigg), \tag{3.4}$$

wobei

$$n_l := \text{ Anzahl Neuronen in Schicht } l.$$

Um die Übersicht zu bewahren, werden die Gewichte oft in einer sogenannten *Gewichtsmatrix* zusammengefasst. Für die obige Abbildung sind die Gewichtsmatrizen

$$W^{(1)} = \begin{pmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{pmatrix}, \; W^{(2)} = \begin{pmatrix} w_{11}^{(2)} & w_{12}^{(2)} \\ w_{21}^{(2)} & w_{22}^{(2)} \\ w_{31}^{(2)} & w_{32}^{(2)} \end{pmatrix}, \; W^{(3)} = \begin{pmatrix} w_{11}^{(3)} & w_{12}^{(3)} & w_{13}^{(3)} \end{pmatrix}.$$

Beachte, dass $W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$. Nun kommt eine entscheidende Beobachtung. Wir können die Gleichungen (3.4) nun in kompakter Art und Weise mit Hilfe von Vektoren und Matrizen schreiben. Definieren wir die Vektoren

$$x^{(l)} = \begin{pmatrix} x_1^{(l)} \\ x_2^{(l)} \\ \vdots \\ x_{n_l}^{(l)} \end{pmatrix} \in \mathbb{R}^{n_l}, \quad b^{(l)} = \begin{pmatrix} b_1^{(l)} \\ b_2^{(l)} \\ \vdots \\ b_{n_l}^{(l)} \end{pmatrix} \in \mathbb{R}^{n_l},$$

so gilt

$$x^{(l)} = \sigma\Big(W^{(l)} x^{(l-1)} - b^{(l)}\Big),$$

wobei σ hier komponentenweise auf den Vektor angewendet wird.

Mit Hilfe der Matrix-Vektor Notation können wir nun neuronale Netze mit beliebig vielen Schichten beschreiben. Insgesamt erhalten wir das folgende Modell:

Definition 3.3.1. Ein *mehrschichtiges künstliches neuronales Netz* mit L Schichten und Parametern

$$\theta = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}, \ldots, W^{(L)}, b^{(L)}),$$

wobei

$$b^{(l)} \in \mathbb{R}^{n_l} \quad \text{und} \quad W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}},$$

ist das Modell

$$f_\theta : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}, \quad f_\theta(x) = \big((\sigma \circ f_L) \circ (\sigma \circ f_{L-1}) \circ \cdots \circ (\sigma \circ f_1)\big)(x),$$

mit

$$f_l(x) = W^{(l)}x - b^{(l)}$$

für $l = 1, \ldots, L$.

Das Modell in Definition (3.3.1) wird auch als *Feedforward-Netzwerk* bezeichnet, weil die Information nur in eine Richtung fließt, nämlich von der Eingabeschicht zur Ausgabeschicht. Es gibt auch sogenannte *rekurrente neuronale Netze*, bei denen die Information in beide Richtungen fließt darf.

3.3.4 Aktivierungsfunktionen

Im vorherigen Abschnitt haben wir die Aktivierungsfunktion $\sigma = \max\{0, z\}$ verwendet. Die Motivation dafür war die Diskussion über die biologische Funktion echter Neuronen und dass sie erst ab einem gewissen Schwellenwert aktiv werden. Das mathematische Modell eines neuronalen Netzes in Definition 3.3.1 ist jedoch derart allgemein, dass wir alternative Aktivierungsfunktion wählen können, auch solche die keinen biologischen Ursprung haben. Darüberhinaus können wir für verschiedene Schichten auch unterschiedliche Aktivierungsfunktionen wählen. Die Wahl der Aktivierungsfunktionen kann einen großen Einfluss auf die Leistung des neuronalen Netzes haben. Im Folgenden listen wir einige gängige Aktivierungsfunktionen auf und diskutieren ihre Eigenschaften.

- *ReLU (Rectified Linear Unit):* Diese Aktivierungsfunktion haben wir bereits diskutiert. Sie ist (komponentenweise) definiert als

  $$\sigma(z) = \max\{0, z\}.$$

  Sie ist einfach zu berechnen und hat sich in vielen Anwendungen als effektiv erwiesen.

- *Sigmoid-Funktion:* Die Sigmoid-Aktivierungsfunktion ist (komponentenweise) definiert als

  $$\sigma(z) = \frac{1}{1 + e^{-z}}.$$

  Sie bildet alle reellen Zahlen auf den Bereich $(0, 1)$ ab und wird häufig in Ausgabeschichten für binäre Klassifikationsprobleme verwendet.

- *Tanh (Hyperbolische Tangens):* Die Tangens-Hyperbolicus-Funktion ist (komponentenweise) definiert als

$$\sigma(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}.$$

  Sie bildet reelle Zahlen auf den Bereich $(-1, 1)$ ab und ist wie die Sigmoid-Funktion glatt und differenzierbar, aber zentriert um 0.

Zuletzt definieren wir die *SoftMax-Funktion*.

Definition 3.3.2. Die SoftMax-Aktivierungsfunktion ist die Aktivierungsfunktion

$$\sigma(z) = \mathrm{SoftMax}_t(z) = \frac{1}{\sum_{j=1}^{k} e^{z_j/t}} \begin{pmatrix} e^{z_1/t} \\ e^{z_2/t} \\ \vdots \\ e^{z_k/t} \end{pmatrix}, \quad \text{wobei } z = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_k \end{pmatrix} \in \mathbb{R}^k.$$

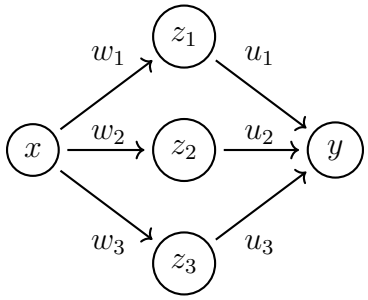
Hierbei ist t eine Konstante, die *Temperatur* genannt wird. Dies kommt daher, dass die Einträge von $\mathrm{SoftMax}_t(z)$ sich mehr und mehr der Gleichverteilung (siehe Beispiel 2.3.3) annähern, je größer t ist – genau wie sich Moleküle zufällig bewegen, wenn die Temperatur steigt. Wenn die Temperatur von uns vorausgesetzt wird oder wenn sie nicht wichtig für die Diskussion ist, schreiben wir SoftMax auch ohne das Subskript t:

$$\mathrm{SoftMax}(z) = \mathrm{SoftMax}_t(z).$$

Die SoftMax-Aktivierungsfunktion nimmt einer Sonderrolle unter den vier hier erwähnten Aktivierungsfunktionen ein. Zunächst ist sie keine komponentenweise Aktivierungsfunktion. Die Ausgabe der SoftMax-Funktion ist ein Vektor, dessen Komponenten alle positiv sind und deren Summe 1 ergibt. D.h. $\sigma(z)$ gibt eine Wahrscheinlichkeitsverteilung über k Klassen an. Die Wahl der SoftMax-Funktion am Ende eines neuronalen Netzes in der Ausgabeschicht definiert somit ein statistisches Modell im Sinne von Definition 3.2.1. Durch das Training der Modellparameter lernt ein neuronales Modell mit SoftMax-Ausgabeschicht, die Wahrscheinlichkeitsverteilung der Klassen für gegebene Eingabedaten zu approximieren. Dabei sorgt die Exponentialfunktion in der SoftMax-Formel dafür, dass größere Eingabewerte exponentiell stärker gewichtet werden, was zu einer klareren Unterscheidung zwischen den Klassen führt.

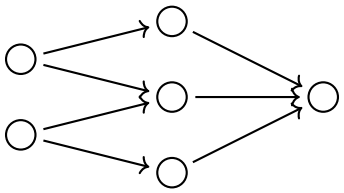
3.3.5 Übungsaufgaben

Aufgabe 3.3.1. Gegeben Sei ein neuronales Netz, beschrieben als Graph wie folgt.



Der Biaswert von z_i ist $b_i, 1 \leq i \leq 3$, und der Biaswert von y ist b_0. Im ersten Layer verwenden wir die nichtlineare Aktivierungsfunktionen σ_1 und im Outputlayer die Aktivierungsfunktionen σ_2. Übersetzen Sie den Graphen in ein Modell $f_\theta : \mathbb{R} \to \mathbb{R}$, indem Sie $f_\theta(x)$ angeben.

Aufgabe 3.3.2. Gegeben Sei ein neuronales Netz, beschrieben als Graph wie folgt.

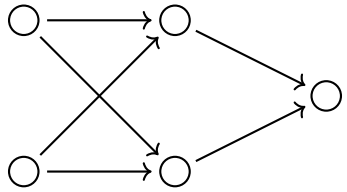


Im ersten Layer verwenden wir die Aktivierungsfunktion σ_1 und im Outputlayer die Aktivierungsfunktion σ_2.

(1) Beschriften Sie die Gewichte der Verbindungen und die Bias-Werte der Neuronen mit Ihrer eigenen Notation.

(2) Übersetzen Sie den Graph in ein Modell $f_\theta : \mathbb{R}^2 \to \mathbb{R}$.

Aufgabe 3.3.3. Gegeben Sei ein neuronales Netz, beschrieben als Graph wie folgt.

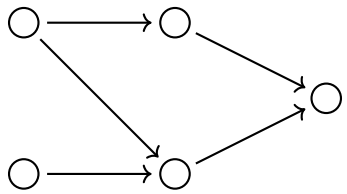


Im ersten Layer verwenden wir die nichtlineare Aktivierungsfunktionen σ_1 und im Outputlayer die Aktivierungsfunktionen σ_2.

(1) Beschriften Sie die Gewichte der Verbindungen und die Bias-Werte der Neuronen mit Ihrer eigenen Notation.

(2) Übersetzen Sie den Graph in ein Modell $f_\theta : \mathbb{R}^2 \to \mathbb{R}$.

Aufgabe 3.3.4. Gegeben Sei ein neuronales Netz, beschrieben als Graph wie folgt.



Im ersten Layer verwenden wir die nichtlineare Aktivierungsfunktionen σ_1 und im Outputlayer die Aktivierungsfunktionen σ_2.

(1) Beschriften Sie die Gewichte der Verbindungen und die Bias-Werte der Neuronen mit Ihrer eigenen Notation.

(2) Übersetzen Sie den Graph in ein Modell $f_\theta : \mathbb{R}^2 \to \mathbb{R}$.

Aufgabe 3.3.5. Gegeben Sei ein neuronales Netz, beschrieben als Graph wie folgt.

Der Biaswert von z ist b_1, und der Biaswert von y ist b_2. Sowohl im ersten Layer als auch im Outputlayer verwenden wir die Identität $\sigma(x) = x$ als Aktivierungsfunktion. Zeigen Sie, dass das neuronale Netz in der Form $f_\theta(x) = a \cdot x + b$ geschrieben werden kann. Was sind die Parameter θ?

Aufgabe 3.3.6. Seien

$$f_1(x_1, x_2) = \begin{pmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \quad \text{und} \quad f_2(z_1, z_2) = \begin{pmatrix} u_1 & u_2 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} - c,$$

und σ_1, σ_2 nichtlineare Aktivierungsfunktionen.

(1) Beschreiben Sie das neuronale Netz $(\sigma_2 \circ f_2 \circ \sigma_1 \circ f_1) : \mathbb{R}^2 \to \mathbb{R}$ als Graphen.

(2) Was sind die Parameter? Wieviele Parameter has das neuronale Netz insgesamt.

Aufgabe 3.3.7. Seien

$$f_1(x_1, x_2) = \begin{pmatrix} w_1 & 0 \\ 0 & w_2 \\ w_3 & w_4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \quad \text{und}$$

$$f_2(z_1, z_2, z_3) = \begin{pmatrix} u_1 & u_2 & u_3 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} - c,$$

und σ_1, σ_2 nichtlineare Aktivierungsfunktionen.

(1) Beschreiben Sie das neuronale Netz $(\sigma_2 \circ f_2 \circ \sigma_1 \circ f_1) : \mathbb{R}^2 \to \mathbb{R}$ als Graphen.

(2) Was sind die Parameter? Wieviele Parameter has das neuronale Netz insgesamt.

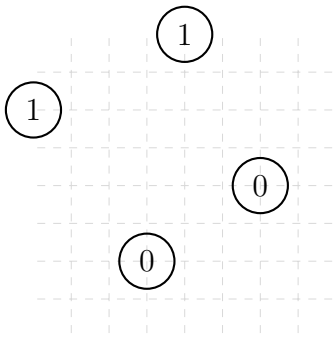
Aufgabe 3.3.8. Es sei $f_\theta(x_1, x_2) = \max\{w_1 x_1 + w_2 x_2 - b, 0\}$ das McCulloch-Pitts Neuron mit Parametern $\theta = (w_1, w_2, b)$.

(1) Sei $\theta = (1, 0, 1)$, also $w_1 = 1$, $w_2 = 0$ und $b = 1$. Skizzieren Sie die zwei Regionen

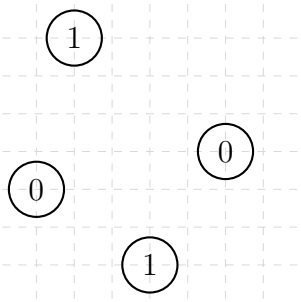
$$R_0 = \{x = (x_1, x_2) \in \mathbb{R}^2 \mid f_\theta(x_1, x_2) = 0\} \quad \text{und}$$
$$R_1 = \{x = (x_1, x_2) \in \mathbb{R}^2 \mid f_\theta(x_1, x_2) > 0\} -$$

(2) Beurteilen Sie jeweils für die beiden unten skizzierten Datensätze, ob wir die Parameter $\theta = (w_1, w_2, b)$ so setzen können, dass f_θ die Klassen 0 und 1 trennt. Begründen Sie Ihre Antwort.

Datensatz 1:



Datensatz 2:



Hinweis: Jeder Kreis steht für ein Datum. Die Position eines Kreises stellt den Wert des Inputdatums $(x_1, x_2) \in \mathbb{R}^2$ dar. Das Label stellt den Wert des Outputdatums (0 oder 1) dar.

Aufgabe 3.3.9. Gehen Sie durch das vierte Jupyter-Notebook. Passen Sie es so an, dass die Daten im `MNIST` Datensatz klassifiziert werden.

4 Large Language Models

Dies Kapitel basiert zu großen Teilen auf dem umfassenden Lehrbuch von Zhang, Lipton, Li und Smola [29]. Für das Kapitel über Large Language Models (LLMs) wurde außerdem das Youtube-Tutorial von Andrej Karpathy [14] herangezogen. Eine weitere hilfreiche Quelle ist die Youtube-Playlist von 3Blue1Brown mit dem Titel "Neural Networks" [6].

Ein *Language Model* hat das Ziel für eine Texteingabe ein neues Stück Text zu generieren, das der Eingabe folgenden soll bzw. sie vervollständigen soll. Z.B. könnte die Eingabe der Text "Mathematik-Lehrer:innen sind" sein. Dann wäre eine mögliche Vervollständigung "Mathematik-Lehrer:innen sind super!". Die Eingabe wird üblicherweise *Prompt* genannt. Ein Language Model lernt aus vorgegebenen Textdaten, wie Prompts typischerweise fortgesetzt werden.

Von einer mathematischen Sichtweise ist ein Language Model ein statistisches Modell $P_\theta(y \mid x)$; siehe Definition 3.2.1. Hierbei sind $x \in E$ und $y \in A$, wobei E die Menge aller möglichen (Eingabe-)Prompts und A die Menge aller möglichen Vervollständigungen ist, und θ bezeichnet wieder die Parameter des Modells. Für einen gegebenen Prompt x berechnet das Modell dann die Wahrscheinlichkeiten für alle möglichen Vervollständigungen y. Dies gibt die Wahrscheinlichkeitsverteilung $P_\theta(y \mid x)$. Anschließend generiert das Language Model eine Vervollständigung y gemäß dieser Verteilung. In obigen Beispiel wäre der Eingabeprompt für das Modell $x =$ "Mathematik-Lehrer:innen sind". Die Wahrscheinlichkeit von $y =$ "super!" gegeben x ist groß genug, so dass, wenn wir die Vervollständigung, generieren, wir mit hoher Wahrscheinlichkeit "super!" erhalten.

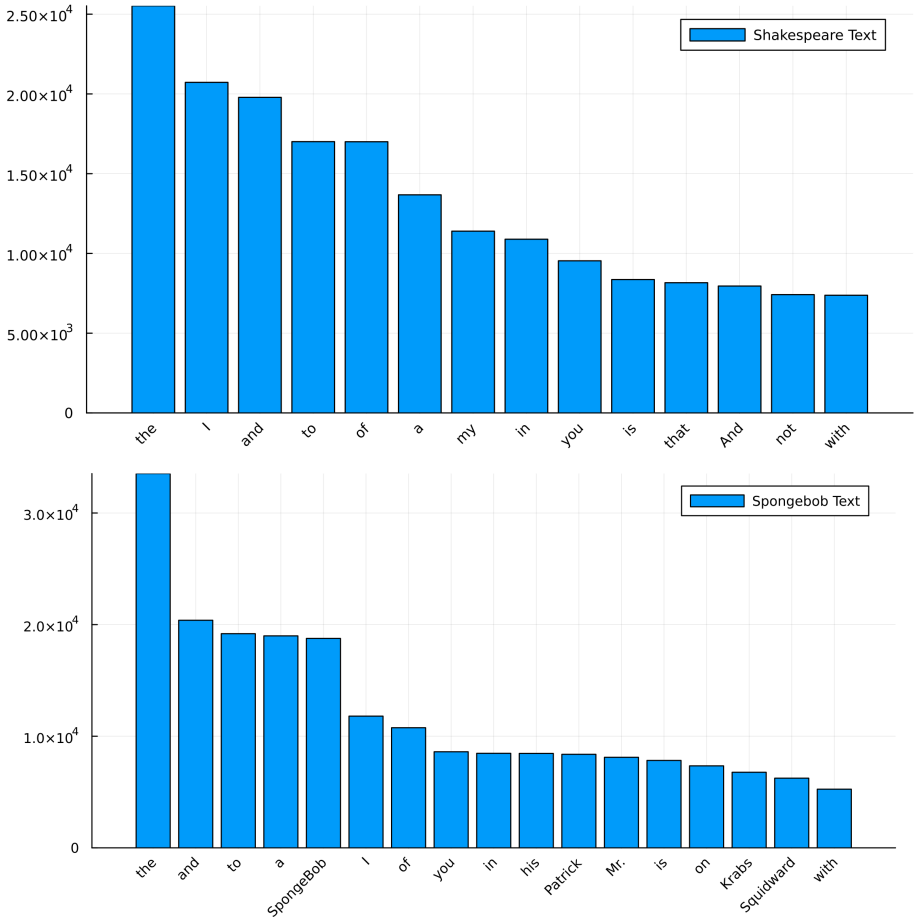
Beispiel 4.0.1. Das Uni-Gram Modell ist ein einfaches Sprachmodell, welches aber sehr hilfreich ist, um die grundlegende Idee hinter Sprachmodellen besser zu verstehen. Gegeben sei der Input Prompt $x =$ "Mathematik-Lehrer:innen sind". Das Uni-Gram Modell ignoriert den Prompt komplett und wählt das nächste Wort

nur anhand der relativen Häufigkeiten des Wortes in den Textdaten. Besteht der Text z.B. aus m Wörtern und sind n_{super} davon "super", so erhalten wir, in Anlehnung an Definition 2.5.1,

$$P_\theta(\text{“super!”} \mid x) = \frac{n_{\text{super}}}{m}.$$

Die Parameter θ dieses Modells sind die Anzahl der Wörter im Text.

Zwei Beispiel Texte, die als Trainingsdaten für ein Uni-Gram Modell verwendet werden könnten, sind Werke von Shakespeare [1] bereitgestellt vom Projekt Gutenberg und Transskripte von 393 Spongebob Episoden [2] von Kaggle. Beides sind englische Texte. Wir visualisieren die (absoluten) Häufigkeiten der Wörter (ohne Sonderzeichen) jeweils in einem Histogramm (siehe Definition 2.1.9).



Beispiel 4.0.2. Das Bi-Gram Modell (siehe z.B. [29, Abschnitt 9.3.1]) funktioniert ähnlich wie Uni-Gram in Beispiel 4.0.1, ignoriert aber den Eingabe-Prompt nicht. Gegeben sei wieder der Input Prompt $x =$ "Mathematik-Lehrer:innen sind". Das Bi-Gram Modell beachtet nur das letzte Wort "sind" und ignoriert alles, was davor kommt. Die Idee von Bi-Gram ist es, zu zählen wie oft "sind" und wie oft "sind super" im Trainingstext vorkommen. Seien dazu n_{sind} die Anzahl wie oft "sind" und $n_{\text{sind super}}$ die Anzahl wie oft "sind super" im Trainingstext vorkommen. Dann erhalten wir, inspiriert von Definition 2.5.1,

$$P_\theta(\text{"super!"} \mid x) = \frac{n_{\text{sind super}}}{n_{\text{sind}}}.$$

Die Parameter θ dieses Modells sind die Anzahl der Wörter und Wortpaare.

Ein Bi-Gram Modell, traininiert auf den Spongebob Daten [2], generiert z.B. für den Input Prompt "Happy" folgenden Output (vgl. Jupyter-Notebook 5):

> *Happy birthday, SpongeBob!*
> *SpongeBob: Oh, I just a good one, not a place him and the door and.*

Dies sieht bereits nach einem sinnvollen Text aus. Bei genaueren Hinschauen sieht man aber jedoch, dass Grammatik und Kontext nicht beachtet werden.

Es gibt auch das Tri-Gram Modell oder, ganz allgemein, das n-Gram Modell, welche Wort Triple bzw. Wort n-tuple zählen, um $P_\theta(y \mid x)$ zu berechnen. Eine etwas nuanciertere Methode ist Skip-Gram, welches wir in Abschnitt 4.1.1 besprechen.

Das Ziel dieses Kapitels ist es, zu erklären, wie moderne Large Language Models (LLM) funktionieren; wie also das statistische Modell $P_\theta(y \mid x)$ in einem LLM berechnet wird. Insbesondere Ein schematische Darstellung eines Language Models ist in Abbildung 4.1 gegeben. Im groben funktioniert es in drei Schritten.

(1) Zunächst wird der Prompt in eine digitale Repräsentation umgewandelt. Dies wird als *Einbettung* bezeichnet. D.h., wir ordnen dem Prompt x eine Folge von Vektoren im $\mathbb{R}^{n_{\text{embed}}}$ zu. Die Dimension n_{embed} heißt *Einbettungsdimension* und ist von Modell zu Modell unterschiedlich.

(2) Anschließend wird die Einbettung verarbeitet und $P_\theta(y \mid x)$ berechnet. In einem Large Language Model (LLM) geschieht diese Verarbeitung durch ein

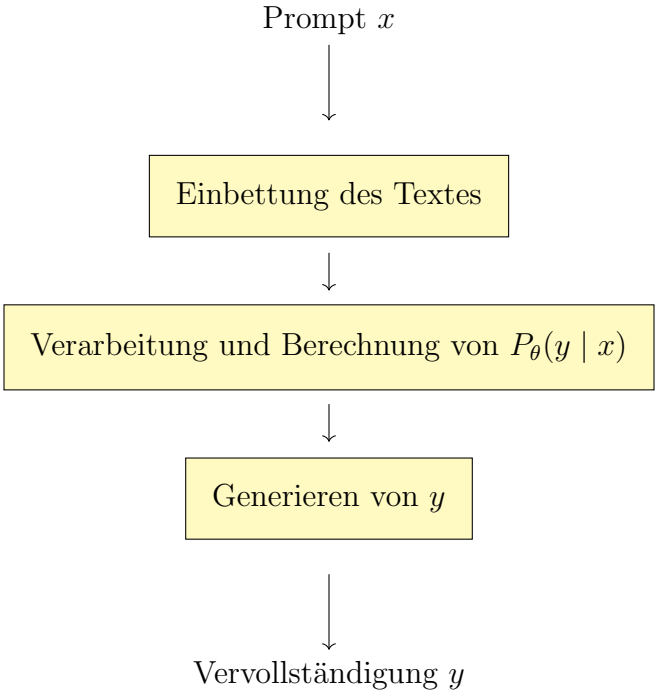


Abbildung 4.1: Schematische Darstellung eines Language Models.

spezielles neuronales Netzwerk, nämlich aus einem Netzwerk sogenannter Transformer-Blöcke. Wir werden Transformer ausführlich in Abschnitt 4.2 besprechen. Die zentrale Eigenschaft eines Transformers ist es, Kontext lernen zu können. Dies ist essentiell für das Ziel bedeutungsvollen Text zu generieren.

(3) Abschließend wird die Vervollständigung aus der Wahrscheinlichkeitsverteilung $P_\theta(y \mid x)$ generiert und ausgegeben.

Insbesondere arbeitet ein LLM, anders als Bi-Gram in Beispiel 4.0.2, mit Einbettungen und nicht direkt mit den Token. Texteinbettungen ermöglichen es geometrische Verfahren zu verwenden, um Information zu verarbeitet. Dadurch kann das Model komplexere Zusammenhänge lernen.

4.1 Text-Einbettung und Tokenisierung

Wie wollen wir einen Text-Prompt x in eine digitale Repräsentation umwandeln, damit ein Computer damit arbeiten kann. Dies geschieht bei LLMs in zwei Schritten: Zunächst wird der Text in sogenannte *Tokens* zerlegt. Dies sind Textbausteine, die aus einem oder mehreren Zeichen bestehen können. Der Prozess der Zerlegung eines Textes in Tokens wird als *Tokenisierung* bezeichnet. Die Menge aller möglichen Tokens wird als *Vokabular* bezeichnet.

Beispiel 4.1.1. Der Text "Mathematik-Lehrer:innen sind super!" könnte in die einzelnen Worte und Zeichen als Tokens

"Mathematik", "-", "Lehrer:innen", "", "sind", "", "super", "!"

zerlegt werden. Dies ist z.B. der Ansatz in Beispiel 4.0.2, wo wir Worte und Wortpaare gezählt haben. Eine andere Möglichkeit wäre

"Mathe", "matik", "-", "Lehrer", ":", "innen", "", "sind", "", "super", "!".

Eine weitere Möglichkeit wäre, den gesamten Text als ein einziges Token zu betrachten, oder jeden einzelnen Buchstaben als separates Token zu betrachten. Es gibt viele verschiedene Möglichkeiten, einen Text in Tokens zu zerlegen.

Sei wieder E die Menge aller möglichen Text-Prompts. Sei zudem F die Menge aller möglichen (endlichen) Tokenfolgen für ein gegebenes Vokabular $\mathcal{V}$; d.h., F ist die Menge von Folgen mit Elementen in $\mathcal{V}$.

Beispiel 4.1.2. Wir betrachten erneut den Text aus Beispiel 4.1.1. Angenommen die Tokens aus diesem Beispiel sind bereits das ganze Vokabular:

$$\mathcal{V} = \{\text{"Mathematik"}, \text{"-"}, \text{"Lehrer:innen"}, \text{""}, \text{"sind"}, \text{"super"}, \text{"!"}\}.$$

Dann ist z.B. die Tokenfolge $\big(\text{"Mathematik"}, \text{"-"}, \text{"!"}\big) \in F$. Genauso ist die Tokenfolge $\big(\text{"super"}, \text{""}, \text{"sind"}, \text{"-"}\big) \in F$. Die Tokenfolgen in F können unterschiedlich lang sein und müssen keinen Sinn ergeben.

Hier kommt die zentrale Definition dieses Abschnitts.

Definition 4.1.1. Sei $\mathcal{V}$ Vokabular und F die Menge aller möglichen Tokenfolgen für $\mathcal{V}$. Wir definieren:

(1) Ein *Tokenisierer* ist eine Abbildung

$$\tau : E \to F,$$

die einem Prompt x in die zugehörige Tokenfolge $\tau(x) \in F$ zerlegt.

(2) Sei $n_{\text{embed}} \in \mathbb{N}$. Eine *Einbettung* ist eine Abbildung

$$\varphi : \mathcal{V} \to \mathbb{R}^{n_{\text{embed}}}.$$

Sie ordnet jedem Token $t \in \mathcal{V}$ einen Vektor $\varphi(t) \in \mathbb{R}^{n_{\text{embed}}}$ zu. Die Dimension n_{embed} heißt *Einbettungsdimension*. Die Einbettung einer Tokenfolge $(t_1, t_2, \ldots, t_k) \in F$ ist dann definiert als die Folge der zugehörigen Einbettungs-Vektoren $(\varphi(t_1), \varphi(t_2), \ldots, \varphi(t_k))$.

Beispiel 4.1.3. Wir fahren mit Beispiel 4.1.3 fort. Das Vokabular hat 7 Elemente. Eine mögliche Einbettung $g : \mathcal{V} \to \mathbb{R}^7$ wäre

$$\begin{aligned}
\varphi(\text{“Mathematik”}) &= (1,0,0,0,0,0,0),\\
\varphi(\text{“-”}) &= (0,1,0,0,0,0,0),\\
\varphi(\text{“Lehrer:innen”}) &= (0,0,1,0,0,0,0),\\
\varphi(\text{“”}) &= (0,0,0,1,0,0,0),\\
\varphi(\text{“sind”}) &= (0,0,0,0,1,0,0),\\
\varphi(\text{“super”}) &= (0,0,0,0,0,1,0),\\
\varphi(\text{“!”}) &= (0,0,0,0,0,0,1).
\end{aligned}$$

Diese Art von Einbettung heißt auch *One-Hot-Encoding*, weil jeder Token durch einen Vektor dargestellt wird, der in genau einer Komponente den Wert 1 und in allen anderen Komponenten den Wert 0 hat.

Das Motivation Wort-Einbettungen zu verwenden ist es, geometrische Methoden in $\mathbb{R}^{n_{\text{embed}}}$ zu verwenden, um Information aus Texten zu extrahieren. Dazu ist *One-Hot-Encoding* nicht gut geeignet, weil alle Token einem Standardbasisvektor

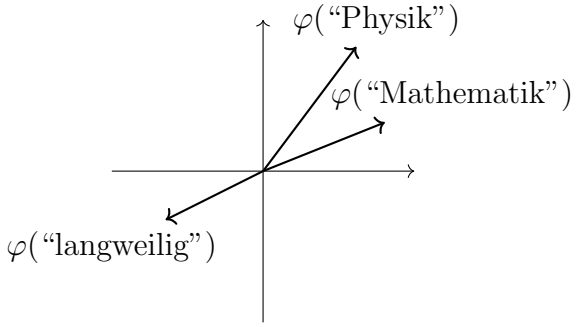


Abbildung 4.2: Beispiel für eine Einbettung in $\mathbb{R}^2$. Die Tokens “Physik” und “Mathematik” sind thematisch ähnlich und werden durch Vektoren dargestellt, die einen kleinen Winkel zueinander haben. Das Token “langweilig” ist thematisch anders und wird durch einen Vektor dargestellt, der einen großen Winkel zu den anderen beiden Vektoren hat.

zugeordnet werden, welche alle den gleichen Abstand und den gleichen Winkel zueinander haben. Bessere Einbettungen ordnen Token Vektoren zu, die semantisch ähnliche Token näher zueinander abbilden. Z.B. könnten die Tokens “Mathematik” und “Physik” Vektoren zugeordnet werden, die einen kleinen Winkel zueinander haben, weil sie thematisch ähnlich sind; siehe Abbildung 4.2. Die Idee ist es also, Information durch Geometrie zu repräsentieren!

Wir wollen im Rest dieses Abschnitts zwei Methoden kennenlernen, um die Einbettung $\varphi : \mathcal{V} \to \mathbb{R}^{n_{\text{embed}}}$ zu lernen. Die erste Methode basiert auf der sogenannten *Skip-Gram*-Modell, die zweite Methode heißt *Continuous-Bag-of-Words*. Beide werden unter dem Namen *Word2Vec* zusammengefasst und haben zum Ziel die Einbettung so zu lernen, dass semantisch ähnliche Tokens durch Vektoren dargestellt werden, die nahe beieinander liegen.

Interessanterweise wurde in [19] beobachtet, dass in diesen Modellen sich semantische Beziehungen zwischen Token in algebraischen Relationen zwischen Vektorn übersetzt werden. Sind z.B. $v_{\text{Berlin}}, v_{\text{Deutschland}}, v_{\text{Paris}}$ und $v_{\text{Frankreich}}$ die Einbettungen der Tokens “Berlin”, “Deutschland”, “Paris” und “Frankreich”, so gilt näherungsweise

$$v_{\text{Berlin}} - v_{\text{Deutschland}} + v_{\text{Frankreich}} = v_{\text{Paris}}.$$

Dies bedeutet, dass die Beziehung “Hauptstadt von” in der Vektorraum-Einbettung durch die obige algebraische Relation dargestellt wird. Solche Relationen sind besonders nützlich für viele Anwendungen in der natürlichen Sprachverarbeitung.

Insbesondere bleibt diese Relation unter der Transformation durch lineare Abbildungen (2.4) erhalten. Der Grund ist die Eigenschaft (2.5). D.h. wir können lineare Abbildungen verwenden, um diese Vektoren zu manipulieren ohne die semantischen Relationen zu verlieren!

4.1.1 Das Skip-Gram Modell

Die Word2Vec-Methode basiert auf der Beobachtung, dass jedes Wort bzw. Token zwei Rollen in einem Text einnehmen kann: Ess können als Wort an sich stehen oder Information zur Bedeutung anderer Wörter liefern. Um dies zu modellieren, werden *zwei* Einbettungsfunktionen gelernt:

$$\varphi : \mathcal{V} \to \mathbb{R}^{n_{\text{embed}}} \quad \text{und} \quad \psi : \mathcal{V} \to \mathbb{R}^{n_{\text{embed}}}.$$

Jedem Token $t \in \mathcal{V}$ werden zwei verschiedene Repräsentationen als Vektoren zugeordnet, eine als *Zentrums-Vektor* $\varphi(t)$ und eine als *Kontext-Vektor* $\psi(t)$. Die erste Einbettung $\varphi(t)$ soll die Rolle von t als Token an sich repräsentieren, während $\psi(t)$ die Rolle von t als Kontext für andere Tokens repräsentieren soll. Nachdem beide gelernt wurden, wird üblicherweise φ als Einbettung für das Language Modell verwendet.

Im Folgenden nehmen wir an, dass das Vokabular R Tokens enthält, die wie folgt nummeriert sind:

$$\mathcal{V} = \{t_1, \ldots, t_R\}. \tag{4.1}$$

Für alle k bezeichnen wir dann mit $v_k = \varphi(t_k)$ der Zentrums-Vektor und mit $w_k = \psi(t_k)$ den Kontext-Vektor des Tokens t_i. Wir können die Ziel- und Kontext-Vektoren in Matrix Form zusammenfassen.

$$V = \begin{pmatrix} | & & | \\ v_1 & \ldots & v_R \\ | & & | \end{pmatrix} \in \mathbb{R}^{n_{\text{embed}} \times R} \text{ und } W = \begin{pmatrix} | & & | \\ w_1 & \ldots & w_R \\ | & & | \end{pmatrix} \in \mathbb{R}^{n_{\text{embed}} \times R}, \tag{4.2}$$

wobei wir die Notation mittels Spaltenvektoren aus (2.2) verwendet haben.

Die Word2Vec-Methode lernt die Einbettungen φ und ψ, indem sie die Wahrscheinlichkeit modelliert, mit der ein Token t_i im Kontext eines anderen Tokens t_j

auftritt. Diese Wahrscheinlichkeit wird mit Hilfe des Skalarprodukt $\langle v_i, w_j \rangle$ (siehe Definition 2.6.1) der beiden Einbettungen modelliert. Je größer $\langle v_i, w_j \rangle$ (also je kleiner der Winkel zwischen v_i und w_j!), desto wahrscheinlicher soll t_j im Kontext von t_i auftreten.

Definition 4.1.2. (vgl. [29, Abschnitt 15.1.3].) Seien $t_i, t_j \in \mathcal{V}$. Wir definieren die bedingte Wahrscheinlichkeit, dass das Token t_j im Kontext des Tokens t_i auftritt:

$$P_\theta^{\text{skip-gram}}(t_j \mid t_i) = \frac{\exp(\langle v_i, w_j \rangle)}{\sum_{k=1}^{R} \exp(\langle v_i, w_k \rangle)}.$$

Die Parameter θ sind die Matrizen V und W aus (4.2).

Eine wichtige Beobachtung ist es, dass wir $P_\theta^{\text{skip-gram}}(t_j \mid t_i)$ erhalten, wenn wir die SoftMax-Aktivierungsfunktion (siehe Definition **??**) auf die Skalarprodukte $\langle v_i, w_k \rangle$ für alle $1 \leq k \leq R$ anwenden.

Die paarweisen Skalarprodukte lassen sich dann mit Hilfe von (2.3) als Matrix-Multiplikation schreiben. Wir fassen das als einen Satz zusammen.

Satz 4.1.1. *Seien V und W wie in* (4.2) *definiert. Dann gilt*

$$V^\top W = (\langle v_i, w_j \rangle)_{i,j=1}^{R}.$$

Demensprechend lässt sich die bedingte Wahrscheinlichkeit in Definition 4.1.2 als

$$P_\theta^{\text{skip-gram}}(t_j \mid t_i) = \frac{\exp\left((V^\top W)_{i,j}\right)}{\sum_{k=1}^{R} \exp\left((V^\top W)_{i,k}\right)}$$

schreiben; d.h., wir wenden SoftMax auf die Zeilen von $V^\top W$ an.

Gegeben sei nun eine Folge von Tokens

$$(s_1, s_2, \ldots, s_k) \in F$$

aus dem Trainings-Text, $s_i \in \mathcal{V} = \{t_1, \ldots, t_R\}$ für $i = 1, \ldots, k$. Das Word2Vec-Modell definiert ein festes *Kontextfenster* $m \in \mathbb{N}$. Innerhalb dieses Kontextfensters werden alle Wahrscheinlichkeiten $P_\theta^{\text{skip-gram}}(s_j \mid s_i)$ für alle Paare (s_i, s_j) berechnet, bei denen s_j im Kontextfenster von s_i liegt. D.h., es gilt $|i - j| \leq m$.

Beispiel 4.1.4. Wir betrachten wieder den Text “Mathematik-Lehrer:innen sind super!” aus Beispiel 4.1.1 zerlegt in die Tokenfolge

“Mathematik”, “-”, “Lehrer:innen”, “”, “sind”, “”, “super”, “!”.

Ist nun ein Kontextfenster $m = 2$ gegeben, dann sind die Tokens im Kontext von “Lehrer:innen” die Tokens “Mathematik”, “-”, “” und “sind”, weil diese Abstand von höchstens $m = 2$ zum Token “Lehrer:innen” haben. Andererseits ist “super” nicht im Kontextfenster von “Lehrer:innen”.

Gegeben sei nun ein Kontextfenster m. Wie in (3.2) erhalten wir folgende Likelihood-Funktion für das Skip-Gram Modell.

Definition 4.1.3. Die Likelihood-Funktiondes Skip-Gram Modells ist definiert als

$$L^{\text{skip-gram}}(\theta) = \prod_{i=1}^{k} \prod_{j:|i-j|\leq m} P_\theta^{\text{skip-gram}}(s_j \mid s_i).$$

Die Parameter θ dieses Modells sind die Matrizen V und W aus (4.2).

Für das Training des Skip-Gram Modells werden nun zufällig Tokenfolgen aus dem Trainings-Text gezogen und die Parameter θ so optimiert, dass die Likelihood-Funktion $L^{\text{skip-gram}}(\theta)$ möglichst groß wird. Das Training von Skip-Gram ist somit überwacht (siehe Abschnitt 3.2.3), weil die Trainingsdaten aus den Paaren (s_i, s_j) in Definition 4.1.3. Da das Modell die Paare selbst aus den Texten extrahiert, spricht man auch von *selbst-überwachtem Lernen*.

4.1.2 Das Continuous-Bag-of-Words Modell

Das Skip-Gram Modell aus dem vorherigen Abschnitt modelliert die Wahrscheinlichkeit, mit der ein Token im Kontext eines anderen Tokens auftritt. Das Continuous-Bag-of-Words Modell (CBOW) verfolgt einen umgekehrten Ansatz. Es modelliert die Wahrscheinlichkeit, mit der ein Token gegeben einen Kontext auftritt. In Anlehnung an Definition 4.1.2 erhalten wir die folgende Definition.

Definition 4.1.4. (vgl. [29, Abschnitt 15.1.4].) Seien $t_i, t_{j_1}, \ldots, t_{j_{2m}} \in \mathcal{V}$ (die Anzahl ist $2m$, weil wir alle Token neben t_i innerhalb eines Kontextfensters der Größe m verwenden wollen). Sei weiterhin

$$\overline{v} := \frac{1}{2m}(v_{j_1} + \cdots + v_{j_{2m}})$$

der (eintragsweise) Mittelwert der Ziel-Vektoren v_{j_k}. Wir definieren die bedingte Wahrscheinlichkeit, dass im Kontext der Token $t_{j_1}, \ldots, t_{j_{2m}}$ das Token t_i auftritt als

$$P_\theta^{\text{cbow}}(t_i \mid t_{j_1}, \ldots, t_{j_{2m}}) = \frac{\exp(\langle \overline{v}, w_i \rangle)}{\sum_{k=1}^{R} \exp(\langle \overline{v}, w_k \rangle)}.$$

Gegeben sei nun wieder eine Folge von Tokens $(s_1, s_2, \ldots, s_k) \in F$ und ein Kontextfenster $m \in \mathbb{N}$. Genau wie in Definiton 4.1.3 erhalten wir die Definition der Likelihood-Funktion für das CBOW Modell.

Definition 4.1.5. Die Likelihood-Funktiondes CBOW Modells ist definiert als

$$L^{\text{cbow}}(\theta) = \prod_{i=1}^{k} P_\theta^{\text{cbow}}(s_i \mid s_{i-m}, \ldots, s_{i-1}, s_{i+1}, \ldots, s_{i+m}).$$

Die Parameter θ dieses Modells sind die Matrizen V und W aus (4.2).

Für das Training des CBOW Modells werden wie für das Skip-Gram Modell zufällig Tokenfolgen aus dem Trainings-Text gezogen und daraufhin die Likelihood-Funktion $L^{\text{cbow}}(\theta)$ oder die Log-Likelihood-Funktion $l^{\text{cbow}}(\theta) = \log L^{\text{cbow}}(\theta)$ optimiert. Das Training von CBOW ist somit genau wie das Training für Skip-Gram überwacht bzw. selbst-überwacht.

Es gibt weitere Modelle zur Text-Einbettung, wie z.B. *GloVe*. GloVe verwendet zusätzlich zu einem Kontextfenster auch die globale Häufigkeit von Token-Paaren im Text, um die Einbettung zu lernen. Dies stellt eine signifikante Verbesserung des Ansatzes zum Lernen von Texteinbettungen dar, da sowohl Skip-Gram als auch CBOW nur lokale kontextuale Informationen verarbeiten können. Wir verweisen hier auf [29, Abschnitt 15.5.1] für weitere Details.

4.1.3 Übungsaufgaben

Aufgabe 4.1.1. Gegeben sei ein Tokenisierer mit Vokabular $V = \{\texttt{h}, \texttt{a}, \texttt{l}, \texttt{o}\}$.

(1) Entscheiden Sie, welche der folgenden Texte durch τ in Token zerlegt werden können:

- `hallo`
- `aal`
- `holla`
- `laos`

(2) Geben Sie für alle Texte in (a), wenn möglich, eine Tokenisierung an.

Aufgabe 4.1.2. Gegeben sei ein Tokenisierer mit Vokabular $V = \{\texttt{ab}, \texttt{bc}, \texttt{abc}\}$.

(1) Entscheiden Sie, welche der folgenden Texte durch τ in Token zerlegt werden können:

- `ababc`
- `abbc`
- `abcab`
- `abababc`

(2) Geben Sie für alle Texte in (a), wenn möglich, eine Tokenisierung an.

Aufgabe 4.1.3. Wir betrachten zwei einfache Tokenisierungsverfahren:

(1) *Wort-Tokenisierer:* Dieser Tokenisierer zerlegt einen Text in Wörter, die durch Leerzeichen getrennt sind. Z.B. wird der Text "Ich liebe Mathe" in die Token `Ich`, `liebe` und `Mathe` zerlegt.

(2) *Zeichen-Tokenisierer:* Dieser Tokenisierer zerlegt einen Text in einzelne Zeichen. Z.B. wird der Text "Ich liebe Mathe" in die Token `I`, `c`, `h`, ␣, `l`, `i`, `e`, `b`, `e`, ␣, `M`, `a`, `t`, `h` und `e` zerlegt.

Diskutieren Sie mögliche Vor- und Nachteile der beiden Tokenisierungsverfahren.

Aufgabe 4.1.4. (1) Gegeben sei der Text "Mathe". Auf wieviele Arten lässt sich der Text in Token zerlegen?

(2) Gegeben sei der Text "Osnabrück". Auf wieviele Arten lässt sich der Text in Token zerlegen?

(3) Ganz allgemein, auf wieviele Arten lässt sich ein Text, der aus N Buchstaben und Sonderzeichen besteht, in Token zerlegen?

4.2 Transformer

Text-Einbettungen wie Skip-Gram und CBOW haben einen entscheidenen Nachteil: Jedem Token wird ein fester Vektor zugeordnet, unabhängig davon, in welchem Kontext das Token auftritt. Dies ist problematisch, weil die Bedeutung eines Tokens stark vom Kontext abhängen kann. Z.B. hat das Wort "Flügel" in den Sätzen "Der Vogel hat Flügel." und "Ich spiele auf dem Flügel." unterschiedliche Bedeutungen. Der *Transformer* löst dieses Problem, indem er Texteinbettungen kontextabhängig weiterverarbeitet. Die Einbettung von "Flügel" wird also für die beiden Beispiel-Sätze unteschiedlich behandelt. Dies geschieht durch den sogenannten *Attention*-Mechanismus [25].

Ganz allgemein ist ein Transformer (manchmal auch: Transformer-Block) ein Modell der Form

$$f : \mathbb{R}^{n_{\text{embed}} \times m} \to \mathbb{R}^{n_{\text{embed}} \times m}, \quad X \mapsto f(X),$$

das eine Eingabe-Matrix $X \in \mathbb{R}^{n_{\text{embed}} \times m}$ auf eine Ausgabe-Matrix

$$f(X) \in \mathbb{R}^{n_{\text{embed}} \times m}$$

abbildet. Hierbei ist m die Größe des Kontextfensters, also die maximale Anzahl an Token, die gleichzeitig verarbeitet werden kann. Der Transformer *transformiert* die Einbettung X eine eine neue Einbettung $f(X)$. Diese neue Einbettung ist dann mit kontextualer Information versehen.

Formell müssten wir $f = f_\theta$ schreiben, um die Abhängigkeit von den Modellparametern θ zu verdeutlichen. Wir verzichten in diesem Abschnitt der Übersichtlichkeit halber darauf und diskutieren die Parameter getrennt von der Notation.

Es gibt natürlich viele verschiedene Architekturen für Transformer. Wir werden folgende simple Version besprechen:

$$T = f_{\text{NN}} \circ f_{\text{attention}}. \tag{4.3}$$

Hierbei ist $f_{\text{NN}} : \mathbb{R}^{n_{\text{embed}} \times m} \to \mathbb{R}^{n_{\text{embed}} \times m}$ ein mehrschichtiges neuronales Netz wie in Definition 3.3.1. Der essentielle Teil ist jedoch das Attention-Modell

$$f_{\text{attention}} : \mathbb{R}^{n_{\text{embed}} \times m} \to \mathbb{R}^{n_{\text{embed}} \times m}.$$

Wir werden dieses Modell im Folgenden studieren.

4.2.1 Attention

Das Attention-Modell basiert auf der Idee, dass die Eingabedaten $X \in \mathbb{R}^{n_{\text{embed}} \times m}$ drei verschiedene Repräsentationen haben:

(1) eine *Query*-Repräsentation $Q \in \mathbb{R}^{n_{\text{atten}} \times m}$,

(2) eine *Key*-Repräsentation $K \in \mathbb{R}^{n_{\text{atten}} \times m}$

(3) und eine *Value*-Repräsentation $V \in \mathbb{R}^{n_{\text{embed}} \times m}$

Die (Attention-)Dimension n_{atten} der Query- und Key-Repräsentation kann dabei unterschiedlich zur Einbettungsdimension n_{embed} sein.

Die Spalten von Q, K und V sind die Query-, Key- und Value-Vektoren der einzelnen Tokens in der Eingabe-Matrix X. Sie werden jeweils durch ein künstliches neuronales Netz aus X berechnet:

$$Q = f_Q(X), \quad K = f_K(X), \quad V = f_V(X),$$

wobei $f_Q, f_K : \mathbb{R}^{n_{\text{embed}} \times m} \to \mathbb{R}^{n_{\text{atten}} \times m}$ und $f_V : \mathbb{R}^{n_{\text{embed}} \times m} \to \mathbb{R}^{n_{\text{embed}} \times m}$ mehrschichtige neuronale Netze sind. Die Parameter dieser Netze sind Teil der Modellparameter θ des gesamten Transformers.

Die Attention-Matrix, welche aus Q, K und V generiert wird, wird nun durch die folgende Formel definiert (vgl. [25, Gleichung (1)]):

$$\text{Attention}(Q, K, V) = V \cdot \text{SoftMax}(K^\top Q) \in \mathbb{R}^{n_{\text{embed}} \times m}. \tag{4.4}$$

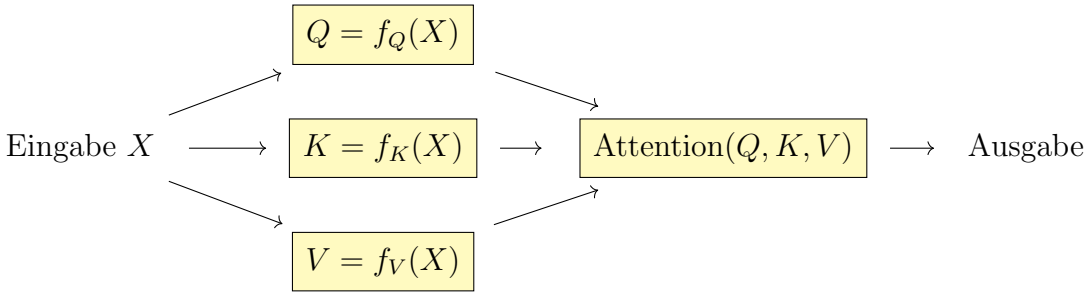


Abbildung 4.3: Schematische Darstellung des Attention Mechanismus.

Hierbei wird die SoftMax-Funktion komponentenweise auf die Spalten der Matrix $K^\top Q \in \mathbb{R}^{m \times m}$ angewendet. Dies ergibt eine Wahrscheinlichkeitsverteilung pro Spalte von $K^\top Q$. Matrixmultiplikation mit V erzeugt eine gewichtete Summe der Spalten der Values. Oft wird die Matrix $K^\top Q$ zusätzlich mit $1/\sqrt{n_{\text{embed}}}$ skaliert. Dies der Stabilisierung der Berechnung, wie in [25] erläutert wird.

Man kann sich die Rollen von Q, K und V wie folgt vorstellen. Die Spalten von Q sind Fragen (Queries), die von den Tokens gestellt werden. Die Spalten von K sind Schlüsselworte (Keys), die die Tokens beschreiben. Die Spalten von V sind die eigentlichen Informationen (Values), die die Tokens enthalten. Das Attention-Modell vergleicht nun eine Fragen mit allen Schlüsselworten, um zu bestimmen, welche Tokens für die Frage relevant sind. Dabei gehört die i-te Spalte von $K^\top Q$ zur Frage von Token i. Die SoftMax-Funktion wandelt diese Spalten von $K^\top Q$ in eine Wahrscheinlichkeitsverteilung um. Die Matrixmultiplikation mit V gewichtet dann die Value-Vektoren entsprechend dieser Wahrscheinlichkeiten. Das Ergebnis ist eine neue Repräsentation jedes Tokens, die Informationen aus den relevanten Tokens im Kontext berücksichtigt.

Wir interpretieren die Gleichung (4.4) geometrisch: Die Einträge von $K^\top Q$ sind die paarweise inneren Produkte der Spalten von Q und K (siehe (2.3)):

$$(K^\top Q)_{i,j} = \langle k_i, q_j \rangle.$$

Je größer der Eintrag $\langle k_i, q_j \rangle$ ist, desto eher ist das Schlüsselwort k_i eine Antwort auf die Frage q_j. Wieder wird also der Winkel zwischen Vektoren (siehe (2.1)) als Bewertung für kontextualen Zusammenhang verwendet!

Beispiel 4.2.1. Angenommen $m = 3$. Dann haben Q, K und V alle $m = 3$ Spalten. Seien diese

$$Q = \begin{pmatrix} | & | & | \\ q_1 & q_2 & q_3 \\ | & | & | \end{pmatrix}, \quad K = \begin{pmatrix} | & | & | \\ k_1 & k_2 & k_3 \\ | & | & | \end{pmatrix}, \quad V = \begin{pmatrix} | & | & | \\ v_1 & v_2 & v_3 \\ | & | & | \end{pmatrix}.$$

Dann gilt

$$K^\top Q = \begin{pmatrix} \langle k_1, q_1 \rangle & \langle k_1, q_2 \rangle & \langle k_1, q_3 \rangle \\ \langle k_2, q_1 \rangle & \langle k_2, q_2 \rangle & \langle k_2, q_3 \rangle \\ \langle k_3, q_1 \rangle & \langle k_3, q_2 \rangle & \langle k_3, q_3 \rangle \end{pmatrix}.$$

Die SoftMax-Funktion wird nun auf jede Spalte von $K^\top Q$ angewendet. Die erste Spalte von $\mathrm{SoftMax}(K^\top Q)$ ist also

$$\mathrm{SoftMax}(K^\top Q) = \begin{pmatrix} | & | & | \\ a_1 & a_2 & a_3 \\ | & | & | \end{pmatrix}, \quad a_i = \mathrm{SoftMax} \begin{pmatrix} \langle k_1, q_i \rangle \\ \langle k_2, q_i \rangle \\ \langle k_3, q_i \rangle \end{pmatrix}.$$

D.h. die j-te Spalte a_j ist die durch SoftMax entstehende Wahrscheinlichkeitsverteilung, die durch die Anfragen von q_j auf die Schlüsselworte k_1, k_2 und k_3 entsteht. Dementsprechend ist die j-te Spalte der Attention.Matrix die durch a_j gewichtete Summe der Spalten von V:

$$\mathrm{Attention}(Q, K, V) = \begin{pmatrix} | & | & | \\ Va_1 & Va_2 & Va_3 \\ | & | & | \end{pmatrix}.$$

Wir fassen zusammen.

Definition 4.2.1. Das Modell $f_{\text{attention}} : \mathbb{R}^{n_{\text{embed}} \times m} \to \mathbb{R}^{n_{\text{embed}} \times m}$ ist definiert durch

$$f_{\text{attention}}(X) = \mathrm{Attention}(Q, K, V),$$

wobei $Q = f_Q(X)$, $K = f_K(X)$ und $V = f_V(X)$ die Query-, Key- und Value-Repräsentationen sind, die durch neuronale Netze f_Q, f_K und f_V aus der Eingabe-Matrix X berechnet werden.

Üblicherweise sind die neuronalen Netze f_Q, f_K und f_V einfach lineare Abbildungen (also neuronale Netze mit einer Schicht und ohne Aktivierungsfunktion). Wir nennen diesen Fall die *Standardform* des Attention-Modells

Definition 4.2.2. Die Query-, Key- und Value-Repräsentationen in Definition 4.2.1 seien lineare Abbildungen, also $Q = W_Q\, X$, $K = W_K\, X$ und $V = W_V\, X$, wobei $W_Q, W_K \in \mathbb{R}^{n_{\text{atten}} \times n_{\text{embed}}}$ und $W_V \in \mathbb{R}^{n_{\text{embed}} \times n_{\text{embed}}}$ Gewichtsmatrizen sind. Dann nennen wir das Modell $f_{\text{attention}}$ ein Attention-Modell in Standardform.

Sei nun $f_{\text{attention}}$ ein Attention-Modell in Standardform. Wir können dann in (4.4) einsetzen und erhalten $f_{\text{attention}}(X) = W_V\, X \cdot \text{SoftMax}(X^\top W_K^\top W_Q^\top X)$. Diese Gleichung verdeutlicht eine Schwäche von Attention-Modellen in Standardform: Permutieren wir die Spalten von X (was äquivalent dazu ist, die Reihenfolge der Token zu ändern), so werden die Spalten von $f_{\text{attention}}(X)$ entsprechend permutiert. Wir sagen dazu, dass $f_{\text{attention}}(X)$ *Permutations-äquivariant ist*. Das Modell $f_{\text{attention}}$ ist also in gewisser Weise unempfindlich gegenüber der Reihenfolge der Eingabetoken. Eine Idee, um diesen Nachteil auszugleichen, ist es, ein neuronales Netz ρ, welches empfindlich auf Permutation reagiert, zu trainineren und das Attention-Modell zu $f_{\text{attention}}(X + \rho(X))$ zu modifizieren. In diesem Ansatz nennt man ρ *Positionscodierung*.

4.2.2 Weitere Strategien: Masking, Multihead Attention, Dropout, Skip Connections und Layer Normalisierung

Wenn alle Einträge von $K^\top Q$ in (4.4) vom Transformer verarbeitet werden, bedeutet dies, dass jedes Token Fragen an alle anderen Token stellen darf. Stellen wir uns nun aber vor, dass Token i per Vervollständigung aus den vorherigen Token enstanden ist, dann sollte Token j keine Frage an Token i für alle $i > j$ stellen können. Um dies im Transformer darzustellen wird eine Technik names *Maskierung* (Masking) eingesetzt. Die Idee ist simpel: Wir ersetzen $K^\top Q$ durch eine andere Matrix $\text{maskiert}(K^\top Q)_{i,j}$, wobei

$$\text{maskiert}(K^\top Q)_{i,j} = \begin{cases} (K^\top Q)_{i,j}, & \text{falls } i \leq j \\ -\infty, & \text{falls } i > j. \end{cases}.$$

Dies hat den Effekt, dass $\mathrm{SoftMax}(\text{maskiert}(K^\top Q))_{i,j} = e^{-\infty} = 0$, falls $i > j$. Die Anfrage von q_j an k_i findet in diesem Fall nicht statt.

Beispiel 4.2.2. Angenommen $m = 3$. Die Spalten von K seien k_1, k_2, k_3 und die Spalten von Q seien q_1, q_2, q_3. Dann sind die Einträge von $K^\top Q$ ohne und mit Masking gegeben durch

$$K^\top Q = \begin{pmatrix} \langle k_1, q_1\rangle & \langle k_1, q_2\rangle & \langle k_1, q_3\rangle \\ \langle k_2, q_1\rangle & \langle k_2, q_2\rangle & \langle k_3, q_3\rangle \\ \langle k_3, q_1\rangle & \langle k_3, q_2\rangle & \langle k_3, q_3\rangle \end{pmatrix},$$

$$\text{maskiert}(K^\top Q) = \begin{pmatrix} \langle k_1, q_1\rangle & \langle k_1, q_2\rangle & \langle k_1, q_3\rangle \\ -\infty & \langle k_2, q_2\rangle & \langle k_2, q_3\rangle \\ -\infty & -\infty & \langle k_3, q_3\rangle \end{pmatrix}.$$

In $\text{maskiert}(K^\top Q)$ kann also q_1 keine Anfrage an die späteren Token k_2 und k_3 stellen. Ebenso kann q_2 keine Anfrage an k_3 stellen.

Die ursprüngliche Publikation zu Attention [25] schlägt einige weitere Kniffe vor. Der erste davon ist Multihead Attention. Die Idee von Multihead Attention ist es, mehrere Attention-Modelle parallel zu verwenden und deren Outputs im Anschluss hintereinander zu legen. Dies ermöglicht es dem Modell, gleichzeitig mehrere verschiedene Aspekte des Kontextes zu erfassen.

Definition 4.2.3. Wir verwenden die Notation aus Definition 4.2.1. Angenommen n_{embed} lässt sich als Produkt $n_{\text{embed}} = n_h \cdot h$ schreiben. Das Modell

$$X \mapsto \begin{pmatrix} f^{(1)}_{\text{attention}}(X) \\ f^{(2)}_{\text{attention}}(X) \\ \vdots \\ f^{(h)}_{\text{attention}}(X) \end{pmatrix} \in \mathbb{R}^{n_{\text{embed}} \times m},$$

wobei, wie in Definition 4.2.1, die

$$f^{(i)}_{\text{attention}} : \mathbb{R}^{n_{\text{embed}} \times m} \to \mathbb{R}^{n_h \times m}, \quad i = 1, \ldots, h,$$

Attention-Modelle sind, heißt *Multihead Attention* mit h Köpfen.

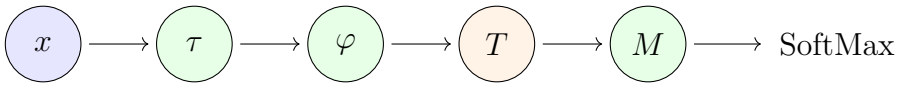
Desweiteren empfiehlt [25] die Verwendung von *Skip Connections* und *Layer Normalisierung*. Skip Connections addieren die Eingabe eines Modells zur Ausgabe des hinzu, bevor diese weitergegeben wird. D.h., Skip Connections verändern den Transformer (4.3) zu

$$T(X) = f_{\text{NN}}(Z) + Z, \qquad \text{wobei } Z = f_{\text{attention}}(X) + X.$$

Layer Normalisierung normalisiert die Ausgabe des Attention-Modells, um sicherzustellen, dass der empirische Mittelwert (Definition 2.2.5) und die empirische Varianz (Definition 2.2.7) der Daten 0 und 1 sind. Dies hilft, das Training zu stabilisieren und die Konvergenz zu beschleunigen. Zusätzlich wird die Verwendung von *Dropout* empfohlen. Dropout "versteckt" zufällig einige neuronale Verbindungen während des Trainings, um Überanpassung zu verhindern.

4.3 Das Large Language Model: Mathematische Perspektive

Wir sind nun bereit, das Large Language Model (LLM) mathematisch zu beschreiben. Dazu ist es hilfreich zunächst mit der einfachsten Form eines LLMs zu beginnen. Diese Form besteht aus einem Transformer(-Block) T und kann wie folgt visualisiert werden.



Hierbei sind

- $x \in E$ der (Eingabe-)Prompt,
- τ der Tokenisierer (siehe Definition 4.1.1), der die Eingabe x auf die Tokenfolge $t = (t_1, \ldots, t_m) = \tau(x) \in F$ abbildet,
- $\varphi : \mathcal{V} \to \mathbb{R}^{n_{\text{embed}}}$ eine Text-Einbettung (siehe Definition 4.1.1), die die Tokenfolge t auf die Einbettungsmatrix X abbildet:

$$X = \begin{pmatrix} | & & | \\ \varphi(t_1) & \ldots & \varphi(t_m) \\ | & & | \end{pmatrix} \in \mathbb{R}^{n_{\text{embed}} \times m},$$

- $T : \mathbb{R}^{n_{\text{embed}} \times m} \to \mathbb{R}^{n_{\text{embed}} \times m}$ ein Transformer,
- $M : R^{n_{\text{embed}} \times m} \to \mathbb{R}^R$ eine finale Matrixtransformation, genannt *Head*, der Form

$$M(A) = U\,A\,u, \quad U \in R^{R \times n_{\text{embed}}},\ u \in \mathbb{R}^m,$$

wobei $R = \#\mathcal{V}$ die Größe des Vokabulars $\mathcal{V}$ ist (siehe (4.1)).

Es sei jetzt

$$z := (M \circ T \circ \varphi \circ \tau)(x) \in \mathbb{R}^R.$$

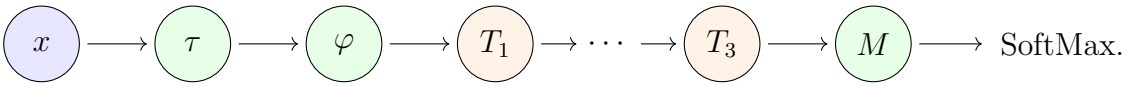
Der letzte Schritt ist dann SoftMax auf w anzuwenden. Dies gibt dann eine Wahrscheinlichkeitsverteilung auf $\mathcal{V}$:

$$P_\theta(t_i \mid x) = \text{SoftMax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^R e^{z_j}}.$$

Diese Formel gibt die Wahrscheinlichkeit Token i als Vervollständigung des Prompts x zu wählen an. Hierbei sind die Parameter θ gegeben durch

- die Parameter der Einbettung φ,
- die Parameter des Transformers T,
- die Matrix U und der Vektor u der Transformation M.

Alternativ können wir mehrere Transformer(-Blöcke) hintereinandern schalten.



Dies führt zu folgender Definition.

Definition 4.3.1. Ein Large Language Model (LLM) mit s Transformerblöcken $T_1, \ldots, T_s$, Vokabular $\mathcal{V} = \{t_1, \ldots, t_R\}$, Tokenisierung τ, Texteinbettung φ und Head M ist ein statistisches Modell der Form

$$P_\theta(t_i \mid x) = \text{SoftMax}(z)_i, \quad z = (M \circ T_s \circ \cdots \circ T_1 \circ \varphi \circ \tau)(x),$$

Da ein LLM ein statistisches Modell für Klassifizierung ist (welches Token, also welche Klasse, kommt als Nächstes?), verwendet man üblicherweise die Cross-Entropy (siehe Definition 3.2.2) als Verlustfunktion für das Training.

Ein LLM wie in Definition 4.3.1 vervollständigt einen Prompt x, indem es ein Token gemäßdem statistischen Modell $P_\theta(t_i \mid x)$ generiert. Dieser Token wird dann an x angehängt. Diese Vervollständigung ist dann ein neuer Prompt, der wiederum vervollständigt wird. Dieser Prozess wird rekursiv fortgesetzt, bis ein Abbruchkriterium erfüllt ist. So erzeugt ein LLM Text.

Beispiel 4.3.1. Wir verwenden ein LLM mit $s = 3$ Transformern und trainieren es auf den Transskripten von 393 Spongebob Episoden [2] (vgl. Jupyter-Notebook 6). Wir geben dem LLM den Prompt $x =$ "Patrick loves math and". Die Ausgabe ist dann z.B.:

> *Patrick loves math and before you leave me. So I just wanted to get this stuff, right and I'll catch them could be back. [sprays in his tears and throws the entire building].*

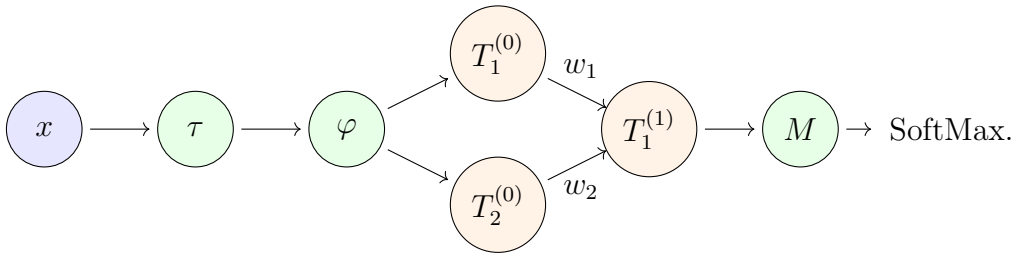
Dieser Text ergibt zwar wenig Sinn, die Grammatik ist jedoch korrekt und die Worte stehen alle in richtigem Kontext zueinander, insbesondere im Vergleich zum von Bi-Gram generierten Text in Beispiel 4.0.2. Dies zeigt, dass das LLM die Struktur der Sprache gelernt hat, auch wenn der semantische Inhalt nicht komplett sinnvoll ist.

Bekannte Sprachmodelle wie ChatGPT oder Gemini funktionieren im Grunde genauso. Das Besondere an diesen Modellen ist die schiere Anzahl an Parametern: Schätzungen zufolge [7] hat die Version ChatGPT-4o 4 Billionen ($4 \cdot 10^{12}$) Parameter und Gemini 1.5 Pro 1.5 Billionen Parameter ($1.5 \cdot 10^{12}$). Der Zusatz "Large" in "Large Language Model" bezieht sich vor allem auf diese große Anzahl der Parameter, mit denen das Modell seine Vorhersagen steuert. Mehr Parameter erhöhen die Kapazität, komplexe Muster aus Daten zu erkennen und zu verallgemeinern. Im Gegensatz zu Sprachmodellen, die ohne Transformer arbeiten, kann ein LLM mit mehr Parametern viel mehr Information verarbeiten. Das liegt daran, dass ein Transformer mehr als nur die unmittelbare Nachbarschaft in Texten modellieren kann. Der Aufmerksamkeitsmechanismus lernt Kontext, der über die Positionierung im Text hinausgeht. So können Beziehungen über weite Distanzen im Text erfasst werden, statt nur die unmittelbare Nachbarschaft zu betrachten. Diese Rechenart lässt sich zudem gut parallelisieren, was das Trainieren großer Modelle praktikabel macht. Diese Fähigkeit ermöglicht die Entstehung so fortschrittlicher KI-Systeme wie ChatGPT oder Gemini.

4.3.1 Expert:innen Modelle

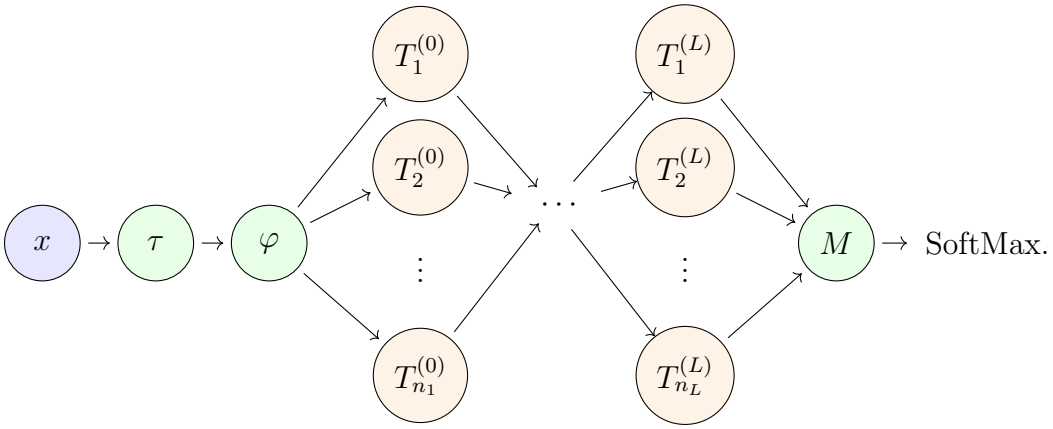
Im vorherigen Abschnitt haben wir das LLM definiert (Definition 4.3.1), indem wir mehrere Transformer hintereinander geschaltet haben. Wir können Transformer auch parallel schalten und wie bei einem künstlichen neuronalen Netz (siehe Abschnitt 3.3.3) mehrere *Schichten* an Transformern verwenden. Diese Art von Modellen heißen *gemischte Expert:innen Modelle* (Mixture of Experts).

Hier ist z.B. ein Modell mit zwei Transformer Schichten:



Hierbei bezeichnen $T_i^{(0)}$ die Transformer in der ersten Schicht und $T_1^{(1)}$ den Transformer in der zweiten Schicht. Wie bei den künstlichen neuronalen Netzen sind die Verbindungen zwischen den Transformern mit Gewichten versehen. Wie im Bild sei w_j das Gewicht der Verbindung von Transformer $T_j^{(0)}$ zu $T_1^{(1)}$. Die Eingabe von $T_1^{(1)}$ ist dann $X' := w_1\, T_1^{(0)}(X) + w_2\, T_2^{(0)}(X)$. Die Eingabe von M ist dann wiederum $T_1^{(1)}(X')$. Jeder Transformer ab der zweiten Schickt erhält also als Eingabe eine gewichtete Summe der Ausgaben der Transformer aus der vorherigen Schicht. Die Gewichte sind Parameter des Modells.

Dieses Beispiel mit zwei Schichten lässt sich auf beliebig viele Schichten und beliebig viele Transformer erweitern. Das folgende Bild zeigt ein LLM mit L Transformer Schichten, wobei die i-te Schicht n_i Transformer enthält.



Es müssen nicht alle Verbindungen vorhanden sein. Zudem können Verbindungen auch "nach links" gehen – genau wie bei den rekurrenten neuronalen Netzen (siehe Abschnitt 3.3.3).

Der Name *gemischtes Expert:innen Modell* kommt nun von der Strategie, die einzelnen Transformer im Netzwerk auf bestimmte Aufgaben zu trainieren. Jeder Transformer wird als ein:e *Expert:in* für eine bestimmte Aufgabe betrachtet. Das Netzwerk als Ganzes lernt dann, welche Aufgabe welche:r Expert:in gestellt werden muss, um das beste Ergebnis zu erzielen.

Bei modernen gemischten Expert:innen Modellen wird das Sprachmodell in viele spezialisierte Teilnetze zerlegt, die nicht nur aus einem einzelnen Transformer bestehen müssen, sondern einzelne Rechenblöcke innerhalb der Schichten,. Ein gezielter Auswahlmechanismus bewertet für jedes Eingabetoken, welche wenigen Expert:innen voraussichtlich am nützlichsten sind, und aktiviert nur diese; alle anderen bleiben inaktiv. So kann das Modell unterschiedliche Muster und Fähigkeiten arbeitsteilig abdecken, ohne dass jedes Token die Rechenarbeit aller Komponenten auslösen muss. D.h. nicht alle Expert:innen sind gleichzeitig aktiv. Pro Eingabetoken wird nur ein kleiner Teil aktiv geschaltet. Der entscheidende Vorteil liegt in der Kombination aus Größe und Effizienz: Die Gesamtzahl der erlernbaren Parameter kann sehr hoch sein (viele spezialisierte Expert:innen), während der tatsächliche Rechenaufwand pro Token nur von einer kleinen aktiven Teilmenge abhängt. Dadurch entsteht hohe Spezialisierung, ohne die Kosten pro Anfrage proportional zur Gesamtgröße steigen zu lassen. Gleichzeitig muss das System zuverlässig auswählen und die Arbeit gleichmäßig verteilen, damit keine Expert:innen überlastet werden und die gewonnenen Fähigkeiten des Modells tatsächlich wirksam werden.

In modernen KI-Sprachsystemen wird die praktische Ausrichtung durch nach dem Training stattfindende Feinabstimmung auf Anweisungen gewährleistet. Diese Feinabstimmung passiert u.A. durch menschliches Feedback und macht aus einem LLM eine hilfreiche, verlässliche Assistenz. Gemischte Expert:innen Modelle ergänzen dies um interne Spezialisierung: Sie verbinden eine breit skalierbare Architektur mit gezielter Aktivierung der passenden Teilnetze. So können solche Systeme zugleich sehr groß und effizient sein.

4.3.2 Übungsaufgaben

Aufgabe 4.3.1. Führen Sie das sechste Jupyter-Notebook über Large Language Models aus und versuchen Sie jeden Schritt nachzuvollziehen. Dieses Notebook basiert in großen Teilen auf dem Blog-Post *Generative transformer from first principles in Julia* von Lior Sinai [24].

Aufgabe 4.3.2. In dieser Vorlesung haben wir die mathematischen Grundlagen von KI und Datenanalyse kennengerlernt. Ein großer Fokus lag dabei auf sogenannten Modellen, inbesondere künstliche neuronale Netze und Large Language Models. Diskutieren Sie untereinander, ob und wie diese Inhalte in den Schullehrplan integriert werden könnten. Was wären geeignete Themenbereiche? Welche Altersstufen sind geeignet? Welche Vorkenntnisse werden benötigt? Welche Kompetenzen sollten vermittelt werden? Ist der mathematische Ansatz sinnvoll für den Schulunterricht? Oder braucht es einen umfassenderen Zugang, auch soziale und ethische Aspekte einbezieht? Wie könnte eine Fortbildung für Lehrkräfte aussehen, die diese Themen unterrichten sollen?

Aufgabe 4.3.3. Überlegen Sie, was für Sie das wichtigste und interessanteste ist, was Sie in dieser Vorlesung über KI gelernt haben. Teilen Sie – wenn Sie möchten – Ihre Gedanken über Social Media. Versuchen Sie Ihre Gedanken so zu formulieren, das es für ein breites Publikum verständlich ist. Evaluieren Sie die Rückmeldungen, die Sie erhalten. Was denken Sie ist die allgemeine Wahrnehmung von KI? Wie unterscheidet sich diese von Ihrer eigenen Sichtweise?

Aufgabe 4.3.4. Lesen Sie die Masterarbeit “Growing up with AI” [12]. Welche Implikation ergeben sich für den Umgang mit Schüler:innen, die mit KI-Systemen aufwachsen? Wie können Lehrkräfte und Schulen diese Herausforderungen adressieren?

Aufgabe 4.3.5. Beim Einfügen der Daten zur Referenz [12] wurde ein LLM benutzt. Dabei wurden männliche Pronomen ("er" und "seine") von der KI verwendet, obwohl die Autorin Stefania Druga weiblich ist. Was könnte der Grund dafür sein?

Aufgabe 4.3.6. Hören Sie die ARD Audiodokumentation "Künstliche Nähe – Doku über KI, Vertrauen und Abhängigkeit" [3] (Achtung: Triggerwarnung wegen Einsamkeit und Anorexie). Reflektieren Sie den Inhalt vor dem Hintergrund, was Sie in der Vorlesung gelernt haben.

Literaturverzeichnis

[1] https://www.projekt-gutenberg.org/autoren/namen/shakespr.html.

[2] https://www.kaggle.com/datasets/mikhailgaerlan/spongebob-squarepants-completed-transcripts.

[3] https://www.ardaudiothek.de/episode/urn:ard:section:b5a7480aad89b0b6/.

[4] https://www.weizenbaum-institut.de/forschung-aufbauphase/fg20/.

[5] Jupyter notebooks.
https://jupyter.org.

[6] 3Blue1Brown. Neural networks.
https://www.youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi.

[7] Ahmed Bahaa Eldin. Gemini 1.5 Pro vs. GPT-4o: A Head-to-Head Showdown.
https://medium.com/@neltac33/gemini-1-5-pro-vs-gpt-4o-a-head-to-head-showdown-29c4cc837e7b.

[8] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[9] Paul Breiding and Samantha Fairchild. *Mathematical Methods in Data Science*. Unpublished work in progress. https://pbrdng.github.io/MathData.pdf.

[10] Walter Buckley. *Sociology and modern systems theory*. Oxford, England: Prentice-Hall, 1967.

[11] Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020.

[12] Stefania Druga. Growing up with AI. Cognimates: from coding to teaching machines. Master's thesis, Massachusetts Institute of Technology, 2018.

[13] Howard Gardner. *Frames of Mind: The Theory of Multiple Intelligences*. Basic Books, 3rd edition, 2011.

[14] Andrej Karpathy. Let's build GPT: from scratch, in code, spelled out.
https://www.youtube.com/watch?v=kCc8FmEb1nY.

[15] Ina Kersten. *Analytische Geometrie und lineare Algebra, Band 1*. Universitätsdrucke Göttingen. Universitätsverlag Göttingen, Göttingen, 2005.

[16] Ulrich Krengel. *Einführung in die Wahrscheinlichkeitstheorie und Statistik*. Springer-Verlag, 2015.

[17] Kultusministerkonferenz (KMK). Verfahren zur Qualitätssicherung auf Schulebene.
https://www.kmk.org/themen/qualitaetssicherung-in-schulen/bildungsmonitoring/verfahren-zur-qualitaetssicherung-auf-schulebene.html.

[18] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

[19] Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *North American Chapter of the Association for Computational Linguistics*, 2013.

[20] Rainer Mühlhoff and Marte Henningsen. Chatbots im schulunterricht: Wir testen das fobizz-tool zur automatischen bewertung von hausaufgaben.

[21] George Pólya. Eine Wahrscheinlichkeitsaufgabe in der Kundenwerbung. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 10(1):96–97, 1930.

[22] Elaine Rich. *Artificial intelligence*. McGraw-Hill, 1983.

[23] Schönert, Ulf. ChatGPT gibt dir dafür eine Drei.
Erschienen in Die ZEIT (27.11.2025)
https://www.zeit.de/2025/50/ki-korrekturhilfen-klausuren-schule-chatgpt-lehrer.

[24] Sinai, Lior. Generative transformer from first principles in Julia.
https://liorsinai.github.io/machine-learning/2024/03/23/transformers-gpt.html.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[26] Frederic Vester. *Denken, Lernen, Vergessen. Was geht in unserem Kopf vor, wie lernt das Gehirn und wann läßt es uns im Stich?* Dtv, München, 1997.

[27] Harald Welzer. Künstliche Dummheit.
Erschienen in Die ZEIT (15.09.2019)
https://www.zeit.de/2019/34/digitalisierung-kuenstliche-intelligenz-algorithmen-denken-dummheit.

[28] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms, 2017.

[29] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*. Cambridge University Press, 2023. https://D2L.ai.

5 Abschließende Worte

Die Veröffentlichung von ChatGPT-3 Ende 2022 hat weltweit einen KI-Hype ausgelöst. Diesem Hype müssen sich auch Lehrer:innen im Unterrichtskontext stellen.

So gibt es diverse KI-Tools, die z.B. bei der Klausurkorrektur helfen sollen. Diese basieren oftmals auf sogenannten Large Language Models (LLMs). Doch wie wir in dieser Vorlesung ausgearbeitet haben, sind LLMs statistische Modelle und daher von Natur aus mit Unsicherheit behaftet. Dies wird in der Öffentlichkeit üblicherweise mit *Halluzinationen* bezeichnet, was jedoch verschleiert, dass die Unsicherheit Teil des Modells ist. Diese Modelle *verstehen* nicht, sondern generieren zufälligen Text. Dementsprechend sollte man vorsichtig bei der Benutzung von LLMs für kritische Aufgaben wie Klausurkorrekturen sein. Die Artikel [20, 23] beschreiben dies eindrücklich.

Das Ziel dieser Vorlesung war es daher, Lehrkräften verständlich zu machen, wie KI-Modelle funktionieren. Ein größer Fokus lag dabei auf LLMs, auf denen auch ChatGPT oder Gemini basieren. Indem Lehrer:innen lernen, wie diese Modelle funktionieren, können Sie sie kritisch einordnen und den Schüler:innen Medienkompetenz mit auf den Weg geben.

Dies geht über die bloße Nutzung von KI-Chatbots für schulische Aufgaben wie z.B. Hausaufgaben hinaus. Menschen, insbesondere Kinder, neigen dazu, Maschinen zu anthropomorphisieren. Stefanian Druga hat dies in ihrer Arbeit "Growing up with AI" [12] bereits 2018 wie folgt zusammengefasst:

> *"[...] humans anthropomorphize objects and are capable of engaging socially with machine."*

und weiter

> *"This leads us to question how much children could be influenced by AI now that it is becoming personified, embodied and able to lead conversations?"*

Lehrer:innen spielen eine zentrale Rolle bei der Beantwortung dieser Frage.

Darüber hinaus ist die öffentliche Debatte über KI meistens wirtschaftlicher oder technologischer Art. So heißt es auf der Website der Arbeitsgruppe "Kritikalität KI-basierter Systeme" des Weizenbaum-Instituts [4]

> *"Obwohl sie den Alltag der Bürger:innen bereits in vielfältigster Weise beeinflussen, agieren viele KI-Systeme bisher als Blackbox. Ihre öffentliche Wahrnehmung ist maßgeblich geprägt von Misstrauen, aber auch von Unwissenheit über das theoretische Gerüst dieser Systeme."*

Dazu passt der bereits 2019 in der ZEIT erschienene Artikel von Harald Welzer [27]. Dort schreibt er, dass "wir Digitalisierung endlich als gesellschaftspolitische Frage begreifen müssen". Er vermisst die Diskussion um die "flächendeckende Implementierung einer Großtechnologie".

Den allgegenwärtigen Einfluss von KI-Modellen zu moderieren oder sogar zu steuern ist eine wichtige gesellschaftliche Aufgabe der nahen Zukunft. Dabei ist es wichtig die Mathematik hinter der KI zu verstehen. Lehrer:innen kommt dabei eine entscheidende Rolle zu.