

# Zum Stellenwert von Verfahren des maschinellen Lernens im allgemeinbildenden Informatikunterricht

Eickhoff-Schachtebeck, A., Strecker, K.

DOI: 10.18420/ibis-03-01-08

## Zusammenfassung

Wir wollen mit diesem Artikel einen Beitrag liefern, der exemplarisch an einigen Beispielf Verfahren des maschinellen Lernens aufzeigt, welche allgemeinbildenden Lernziele damit im Unterricht verfolgt werden können. Dabei steht weniger das wiederholte Anwenden der Verfahren in verschiedenen Kontexten im Mittelpunkt des Unterrichts als vielmehr die daran herauszuarbeitenden Reflexionsaspekte.

## Einleitung

Sicher ist mittlerweile unbestritten, dass das Themengebiet „künstliche Intelligenz“ Einzug in die Lehrpläne und Curricula des Informatikunterrichts entweder schon bereits erhalten hat oder zukünftig haben wird. Auch Unterrichtsmaterialien zum Themengebiet KI und Maschinelles Lernen (ML) sind schon vielerorts entstanden und ihre Menge nimmt kontinuierlich zu.

Oft werden bei bereits vorhandenen Unterrichtsmaterialien einzelne Verfahren des ML (z.B. k-nächste Nachbarn, Q-Lernen-Algorithmus oder k-Means-Clustering) in den Mittelpunkt gestellt und anhand didaktisierter Materialien von den Schülerinnen und Schülern erkundet oder angewendet. Das ist auch richtig, denn ohne konkretes, anwendbares Verfahren bleibt das Themengebiet zu abstrakt und für die Schülerinnen und Schüler nicht greifbar. Um allgemeine Konzepte erarbeiten zu können, ist sogar die Kenntnis von mindestens zwei Verfahren der gleichen Kategorie notwendig, damit die Schülerinnen und Schüler überhaupt zwischen allgemeinem Konzept und verfahrensspezifischer Umsetzung unterscheiden können.

Trotzdem muss bei jedem neu zu integrierenden Thema im Informatikunterricht zusätzlich die Frage gestellt werden, welche allgemeinbildenden Lernziele sich daraus ableiten lassen, die nicht bereits mit den bisherigen inhalts- und prozessbezogenen Kompetenzen abgedeckt sind. Die Entmystifizierung des vermeintlich „schlau“ Computers kann als einziges Argument nicht gelten, zumal dies auch im bisherigen Informatikunterricht durch geeignete Unterrichtsszenarien im Bereich der Algorithmik

oder des physical computings geschehen kann. Die Integration des Themenbereichs Informatik und Gesellschaft findet meist bereits ebenfalls im herkömmlichen Informatikunterricht in Verknüpfung mit den Themengebieten Datenbanken oder Algorithmik statt. Wir sollten festmachen, wodurch genau die Entmystifizierung des „schlau“ Computers im Bereich des Maschinellen Lernens gelingen kann oder wie man im Unterricht vermitteln kann, wie die Begriffe „Lernen“ oder „Künstliche Intelligenz“ informatisch interpretiert werden.

ML als Themengebiet im Informatikunterricht sollte auch dadurch legitimiert werden, dass neben dem bekannten algorithmischen Problemlösen jetzt zusätzlich das datenbasierte Problemlösen tritt, wobei nicht der Schüler oder die Schülerin in seiner/ihrer Rolle als Programmierer oder Programmiererin explizit die Regeln zum Erzeugen von Ausgaben vorgibt, sondern eine Datenanalyse gegebener Daten implizit zu Ausgaben für neue, ähnliche Daten führt. Damit erweitern sich die Strategien der Problemlösung für die Schülerinnen und Schüler um den datenbasierten Ansatz.

Ist das Themengebiet ML durch eine alternative Form des Problemlösens begründet, so stellt sich immer noch die Frage, welche konkreten Verfahren in welcher Tiefe Bestandteil des Unterrichts sein sollten. Auch dies muss u.a. aus dem Blickwinkel der Allgemeinbildung begründet werden. Eine reine rezeptartige, wiederholte Anwendung (didaktisierter) Verfahren und Algorithmen des maschinellen Lernens kann nicht das alleinige Lernziel einer allgemeinbildenden Schule sein. Vielmehr muss aus der Kenntnis und Anwendung der informatischen Verfahren auf der Reflexionsebene im Unterricht auch herausgearbeitet werden, welche Potentiale und Nutzen, aber auch Schwächen und Grenzen die dahinterstehenden Konzepte eint, um die Schülerinnen und Schüler zu befähigen, auch zukünftig dem gesellschaftlichen Diskurs in diesem Themenbereich folgen zu können.

Daher wollen wir in diesem Artikel für die folgende Auswahl konkreter Verfahren des maschinellen Lernens die Frage nach den dahinterstehenden allgemeinbildenden Lernzielen beantworten. Implizit findet sich dazu auch eine Didaktisierung des jeweiligen Verfahrens und eine methodische Vorgehensweise, wobei

auf verschiedene Zugänge geachtet wird und neben enaktiven Herangehensweisen und Visualisierungen auch programmiersprachliche Zugänge in vereinfachter Form verwendet werden.

## Beispiele konkreter Verfahren

### k- nächste Nachbarn: didaktisiertes Verfahren und Lernziele

Wir beginnen mit dem Verfahren k-nächste Nachbarn. In dem Lernszenario aus (Brandt, Eickhoff-Schachtebeck & Strecker 2022a) geht es um die Suche der passenden Mantelgröße für den eigenen Hund (Datenpunkt B in Abbildung 1). Die Käufe zufriedener Kunden (Trainingsdaten) sind in einer Grafik abgebildet. Ohne Kenntnis des Verfahrens k-nächste Nachbarn sollen die Lernenden begründen, für welche Mantelgröße sie sich entscheiden würden (bzw. die Besitzer der Hunde F(ifi), R(ocky) und N(ala)). Ein Hund wird über einen Datenpunkt mit den Merkmalen Rückenlänge (x-Achse) und Schulterhöhe (y-Achse) abgebildet. Die zugehörige Mantelgröße wird über die Farbe des Datenpunktes bestimmt (Abbildung 1). Die Beschränkung auf drei Merkmale ermöglicht die Darstellung der Daten in einem zweidimensionalen Koordinatensystem und so einen enaktiven Zugang.

Es wird gehofft, dass sich eine Strategie unter den Ideen der Lernenden findet, die dem Verfahren k-nächste Nachbarn entspricht, nämlich sich für die Mantelgröße zu entscheiden, welche der Mehrheit der z.B. fünf nächsten Datenpunkte entspricht. In diesem Fall würde die Wahl für den eigenen Hund Blacky auf S fallen (Fifi: S, Rocky: M und Nala: L).

So entdecken die Lernenden selbst das Grundprinzip des Verfahrens k-nächste Nachbarn. Gleichzeitig kann man aber auch über das Verfahren an sich reflektieren und dabei folgende Punkte festhalten:

- Vielleicht ist Größe M für Blacky besser geeignet. Das Gewicht des Hundes wurde ja gar nicht berücksichtigt und könnte bei der Wahl der Mantelgröße ebenfalls eine Rolle spielen. Dahinter steht die Erkenntnis, dass die Daten im Beispiel auf drei Merkmale reduziert wurden (Rückenlänge des Hundes, Schulterhöhe des Hundes und die gewählte Mantelgröße). Die Frage ist, ob durch die Reduktion nicht relevante Merkmale unberücksichtigt geblieben sind, bzw. wie man überhaupt relevante Merkmale identifiziert.
- Durch die Reduktion auf zwei Merkmale des Hundes sind die Trainingsdaten vielleicht ungeeignet oder ungenügend.
- Bereits klassifizierte Trainingsdaten müssen in ausreichender Zahl vorliegen.
- Um den nächsten Nachbarn zu bestimmen, wurde der euklidische Abstand gewählt. Das ist aber nur möglich, wenn auf den Achsen Zahlenwerte vorliegen. Mindestens muss es eine Ordnung auf den Datenwerten geben. Auch müssen die Einheiten an den Achsen stimmig zueinander sein, bzw. die Daten normiert vorliegen. Was wäre beim Merkmal „Beruf“ der Abstand zwischen einem Feuerwehrmann und einer Polizistin?
- Welche Zahl wählt man für k? Würden wir k=3 wählen, würden wir für Nala im obigen Beispiel die Mantelgröße M vorhersagen, im Fall k=5 die Größe L. Dahinter steht eigentlich die Frage: Wann kann man von Mustern in Daten sprechen und was sind Ausreißer?

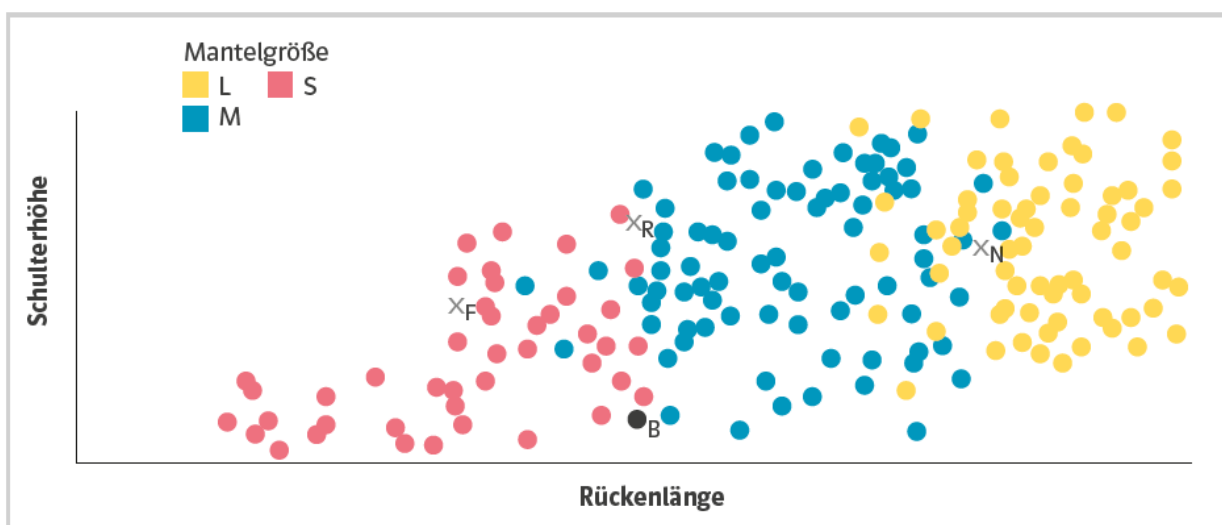


Abbildung 1: Mantelgröße für Blacky gesucht

- Basierend auf ausreichend vielen Trainingsdaten, die klassifiziert sind, treffen wir mit der Methode k-nächste Nachbarn eine Entscheidung für neue, ähnliche Daten. Dabei handelt es sich um eine datenbasierte Problemlösestrategie. Was sind die Unterschiede, was die Gemeinsamkeiten zu einer regelbasierten Problemlösestrategie?
- Probleme der Überanpassung oder Unteranpassung können ebenfalls reflektiert werden. Für  $k=1$  beispielsweise würden Ausreißer in Daten als Muster interpretiert. Die Wahl für den Datenpunkt N im Fall  $k=1$  wäre Mantelgröße M. Man spricht von Überanpassung. Für  $k=\text{Anzahl aller Datenpunkte}$  würde selbst für den Datenpunkt F die Mantelgröße M gewählt, weil die Anzahl blauer Datenpunkte insgesamt am größten ist. Man spricht auch von Unteranpassung.

### Entscheidungsbäume: didaktisiertes Verfahren und Lernziele

In dem Lernszenario zu den Abbildungen 2-5 aus (Brandt, Eickhoff-Schachtebeck & Strecker 2022b) geht es darum, dass eine Software zukünftig automatisiert entscheiden soll, ob geerntet oder noch gewartet werden soll. Dazu sind die Entscheidungen eines erfahrenen Landwirts in der Vergangenheit protokolliert worden. Aus diesen Trainingsdaten erstellt eine Software automatisch einen Entscheidungsbaum, der dann für zukünftige Entscheidungen genutzt wird. Die Daten beinhalten drei Merkmale: die aktuelle Regenwahrscheinlichkeit, die aktuelle Feuchte des Getreides und die Entscheidung des Landwirts zur Ernte. Weizen ist unter einer Feuchte von 14,5% trocken und erntereif. Droht Regen, ist es aber manchmal besser, nicht ganz trockenes Getreide einzufahren (auch wenn dies beim Verkauf Abzüge bedeuten könnte), als noch größere Einbußen durch den eventuellen Regen in Kauf zu nehmen. Die Trainingsda-

ten sind in Abbildung 2 gegeben. Abbildungen 3 bis 5 (nächste Seite) zeigen die Erstellung eines Entscheidungsbaums per Hand, obwohl die Konstruktion des Baums eigentlich von einer Software automatisiert durchgeführt wird. Die Beschränkung auf drei Merkmale ermöglicht die Darstellung der Daten in einem zweidimensionalen Koordinatensystem und so einen enaktiven Zugang, bei dem das Verfahren der rekursiven Partitionierung angewendet werden kann. Diese Didaktisierung ergibt sich also aufgrund der gewählten Methodik.

Basierend auf den Trainingsdaten, die das Erfahrungswissen des Landwirts abbilden, werden nach und nach vertikale oder horizontale Bereiche ausgewählt, die Datenpunkte gleicher Farbe vereinen. Mithilfe des Entscheidungsbaums würde für die, nicht in den Trainingsdaten abgebildete, Situation einer Feuchte des Weizens von 15% und einer Regenwahrscheinlichkeit von 40% die Entscheidung „noch warten“ getroffen.

Im Vergleich zum Verfahren k-nächste Nachbarn ergibt sich Folgendes, wodurch teilweise auch ein allgemeines Konzept unabhängig vom konkreten Verfahren verdeutlicht werden kann:

- Die Bedeutung der Trainingsdaten kann hier ebenso gut gezeigt werden, wie beim Verfahren k-nächste Nachbarn
- Auch die Beschränkung auf wenige Merkmale und sich daraus ergebende Probleme kann hier gut thematisiert werden, ebenso aber auch beim Verfahren k-nächste Nachbarn.
- Entscheidungs bäume benötigen weniger eine Vereinheitlichung an den Achsen, da die Merkmale nicht in dem Sinne miteinander kombiniert werden wie beim euklidischen Abstand.

Weitere Lernziele, die anhand von Entscheidungsbäumen angestrebt werden können, wer-

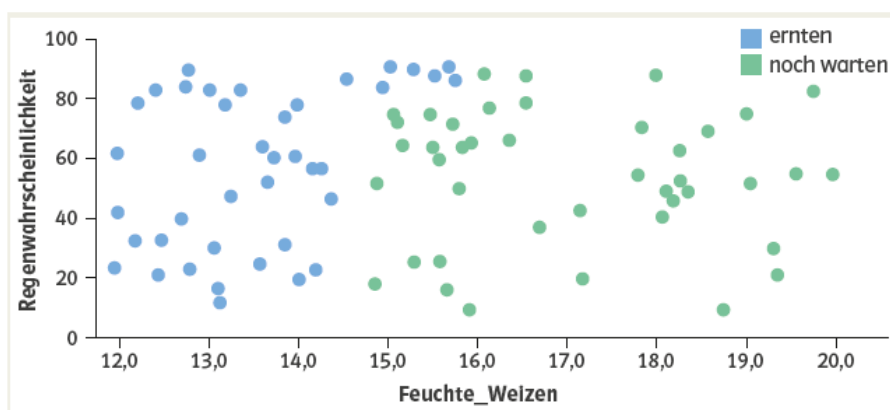


Abbildung 2: Trainingsdaten Ernteentscheidung

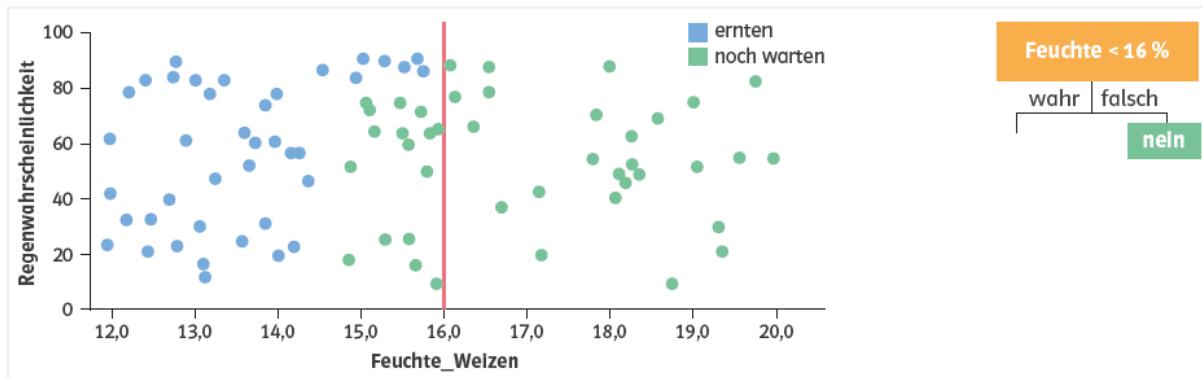


Abbildung 3: Erstellung eines Entscheidungsbaums

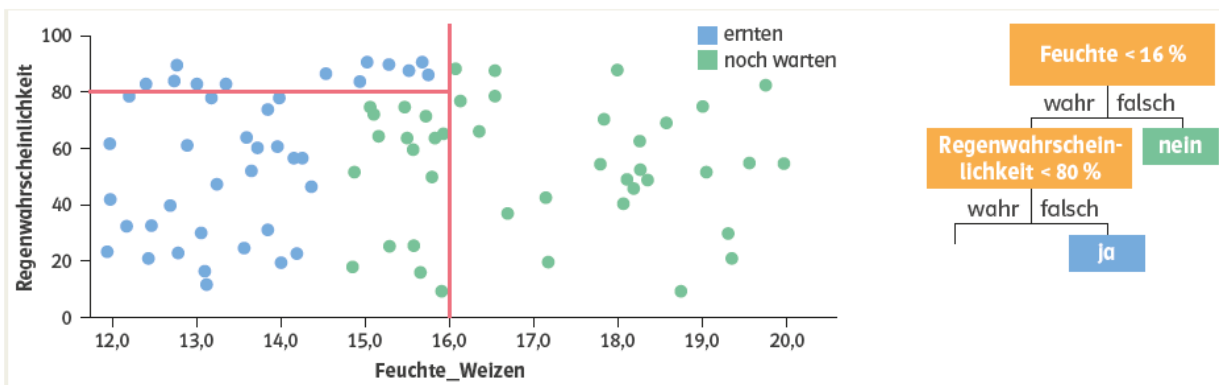


Abbildung 4: Erstellung eines Entscheidungsbaums

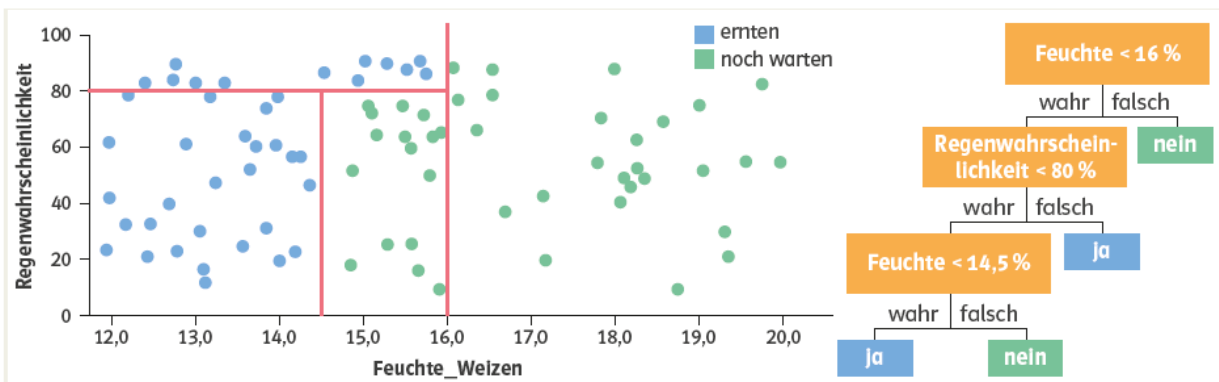


Abbildung 5: Erstellung eines Entscheidungsbaums

den deutlich, wenn man sich überlegt, was passieren würde, wenn es einen Ausreißer in den Daten gibt, bzw. (um bei unserer Geschichte zu bleiben) wenn der Mähdrescher an einem Tag nicht funktioniert hat und diese Entscheidung ebenfalls in die Trainingsdaten mit aufgenommen wurde (Abbildung 6).

- Der Entscheidungsbaum würde in dem Szenario in Abbildung 6 unnötig tief werden. Wir sprechen von Überanpassung. Lernende könnten den Algorithmus daraufhin dahingehend anpassen, dass eine Toleranz eingebaut wird, z.B. es werden horizontale oder vertikale Linien eingezeichnet, die Da-

ten gleicher Farbe vereinen, wobei z.B. 95% der Daten dieselbe Farbe haben müssen.

- Das Beispiel verdeutlicht, dass die Vorhersage eines Verfahrens des ML im individuellen Einzelfall falsch sein kann
- Vielleicht vermischt sich an dieser Stelle das Verfahren mit Vorkenntnissen, die die Lernenden im Themenbereich Algorithmik mit der Datenstruktur „Baum“ gesammelt haben. Deshalb ist es hier von Bedeutung, dass den Lernenden transparent wird, dass die Software den Entscheidungsbaum automatisch generiert, und zwar auf Basis von

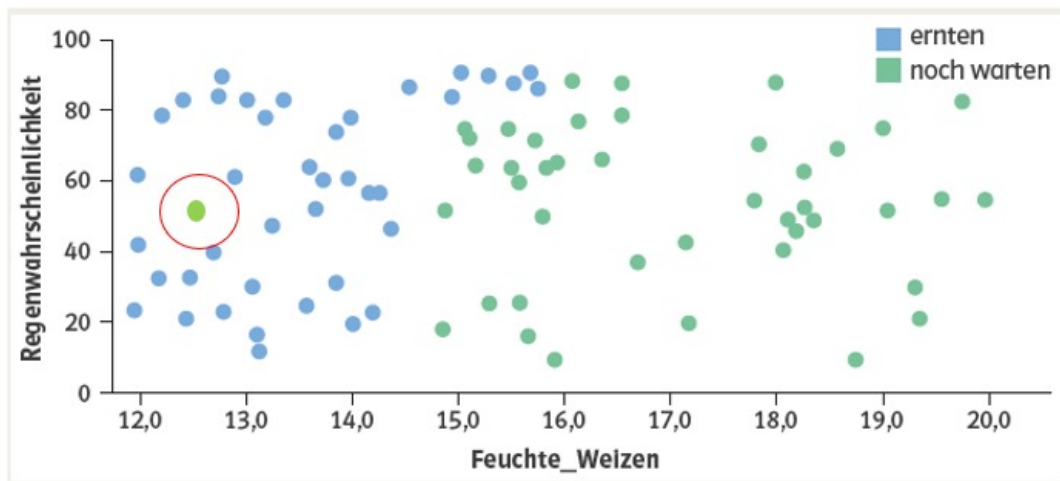


Abbildung 6: Ausreißer in den Daten

Trainingsdaten, und nicht der Programmierer oder die Programmiererin. Thematisiert man dies, wird der Unterschied zwischen einer regelbasierten und einer datenbasierten Strategie greifbar.

- Weiterhin wird bei diesem Verfahren sehr deutlich, dass die Ergebnisse und Vorhersagen des Verfahrens aus Daten der Vergangenheit berechnet werden.

### Lineare Regression: Lernziele

Stellt sich bei den verwendeten Beispielen zu k-nächste Nachbarn und Entscheidungsbäumen das Muster in den Daten durch die Farbe der Datenpunkte dar und hilft damit bei der Kategorisierung ähnlicher, neuer Daten, so kann sich das Muster in Daten auch anders zeigen. Gibt es beispielsweise eine lineare Abhängigkeit der Ausgabedaten von den Eingabedaten, dann kann man mithilfe einer Regressionsgeraden Aussagen für neue, ähnliche Daten treffen. An dieser Stelle verweisen wir auf Beispiele aus dem Mathematikunterricht. Die Thematisierung der linearen Regression als Beispiel für ein Verfahren des überwachten Lernens kann aber sehr gut genutzt werden, um den Unterschied zwischen Korrelation und Kausalität zu verdeutlichen. Wird z.B. das prominente Beispiel der Korrelation zwischen der Anzahl der Geburten und der Anzahl der Storchenpaare (siehe z.B. wikipedia 2025) verwendet, das auch eine lineare Regressionsgerade zeigt, kann der Unterschied zwischen Korrelation und Kausalität thematisiert werden, da Verfahren des ML nur Korrelationen in Daten identifizieren können. Für Unterrichtsideen verweisen wir auf (Eickhoff-Schachtebeck & Strecker 2025).

### Neuronale Netze: didaktisiertes Verfahren und Lernziele

Das Themengebiet maschinelles Lernen ohne neuronale Netze zu thematisieren, erscheint in der Übermächtigkeit des Verfahrens in der Realität schwierig, ist aber didaktisch eine Herausforderung, da das Prinzip der Backpropagation mathematisch komplex ist. Da ein einzelnes Neuron nicht so mächtig ist wie ein neuronales Netz, sollte auch die Funktionalität eines kleinen Netzes thematisiert und verstanden werden. Wir skizzieren an dieser Stelle das virtuell-entdeckte der Funktionsweise neuronaler Netze aus (Brandt, Eickhoff-Schachtebeck & Strecker 2022b) und verweisen zur näheren Erläuterung der hier verwendeten didaktischen Reduktion neuronaler Netze auf (Eickhoff-Schachtebeck & Strecker 2025).

In all unseren folgenden Beispielen verwenden wir die Methode des virtuell-entdeckten Erkundens. Wir präsentieren jeweils eine interaktive Simulation einzelner Neuronen oder kleiner neuronaler Netze, in der die Lernenden aktiv Eingaben verändern können. Durch Beobachtung der Ergebnisse und Reaktionen kann sich so zumindest in Teilen die Funktionsweise erschlossen werden. Die interaktiven Simulationen sind in Scratch implementiert, wodurch der Programmcode von den Lernenden nachvollzogen werden kann. Durch die Bereitstellung einer Implementierung wird gehofft, dass bei den Lernenden zumindest das Gefühl entsteht, einer Implementierung grundsätzlich „gewachsen“ zu sein. Mindestens wird aber deutlich, dass zur Implementierung neuronaler Netze nur algorithmische Grundbausteine von Scratch verwendet werden und keine „Magie“.

An einem einzelnen Neuron kann zunächst gut thematisiert werden, wie sich die Ausgabe aus

Eingaben, Gewichten und Schwellenwert berechnet, wenn wir folgende Didaktisierung zugrunde legen: Das Neuron feuert (Ausgabe dabei wird von dem Programmierer oder der Programmiererin festgelegt), wenn die Summe der Produkte aus Eingabe und dem zugehörigen Kantengewicht einen Schwellenwert erreicht bzw. überschreitet. Sonst feuert das Neuron nicht (Ausgabe dabei wird von dem Programmierer oder der Programmiererin festgelegt.).

Das vorrangige und erste Lernziel sollte sein, dass sich die Funktionalität eines Neurons auch durch die Wahl der Gewichte ergibt. Ändert man die Gewichte eines Neurons oder neuronalen Netzes, ändert sich u. U. auch dessen Funktionalität, was im ersten Schritt auch händisch an einem konkreten Beispiel überlegt werden kann (siehe Eickhoff-Schachtebeck & Strecker 2025), das wir im Folgenden kurz vorstellen. Es geht um den Kontext, dass eine Gärtnerei ihre Freilandpflanzen automatisch bewässern möchte. Es gibt drei verschiedene Gärtnerarten:

Die erste Gärtnerin hat folgende Anforderung: Es soll immer dann bewässert werden, wenn es sehr unwahrscheinlich ist, dass es regnen wird. Egal wie trocken der Boden ist. Die neu gepflanzten Pflanzen brauchen viel Wasser.

Der zweite Gärtner wünscht sich von der automatischen Bewässerung folgendes: Da Wasser gespart werden soll, soll immer nur dann bewässert werden, wenn es sehr unwahrscheinlich ist, dass es regnen wird und gleichzeitig der Boden sehr trocken ist. Und der dritte Gärtner hat diese Wünsche: Es soll immer dann bewässert werden, wenn der Boden sehr trocken ist. Auf die Wettervorhersage verlässt er sich nicht.

Es wird deutlich, dass die Funktionsweise des Neurons von der Wahl der Gewichte abhängt, denn bei unterschiedlicher Wahl der Gewichte reagiert die Bewässerungsanlage auf dieselben Eingaben jeweils verschieden. In den Abbildungen 7 und 8 sind deshalb dieselben Eingabewerte verwendet worden. Die Wahl der Gewichte passt in Abbildung 7 zu den Wünschen der ersten Gärtnerin und in Abbildung 8 zu den Wünschen des dritten Gärtners. Die Wahl der Gewichte für das zweite Szenario kann von den Schülerinnen und Schülern überlegt werden.

In einem zweiten Schritt muss transparent gemacht werden, dass das Einstellen der Gewichte eines Neurons nicht händisch passiert, sondern die Gewichte automatisch in einer Trainingsphase angepasst werden. Dafür ist z.B. ein Fehlersignal notwendig, wenn die Gewichte noch nicht optimal eingestellt sind.

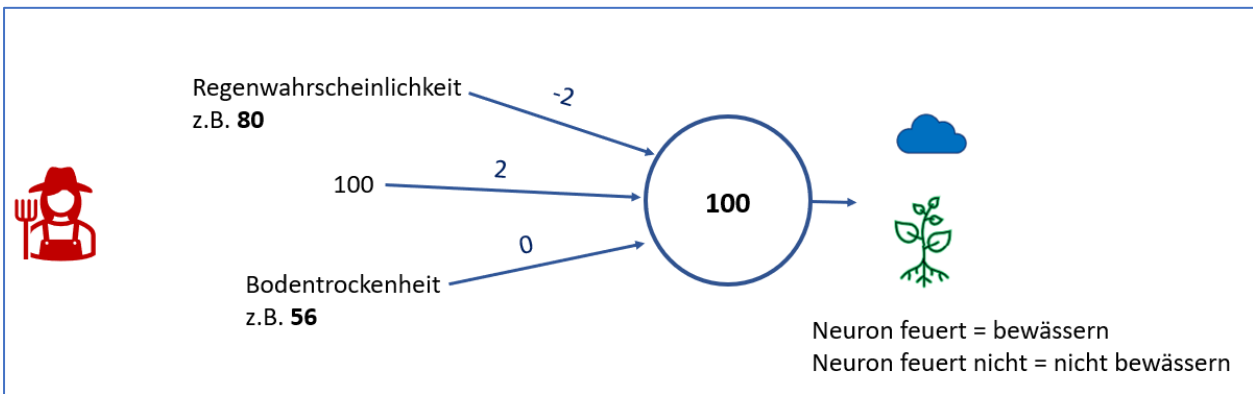


Abbildung 7: Erkunden der Funktionsweise eines Neurons

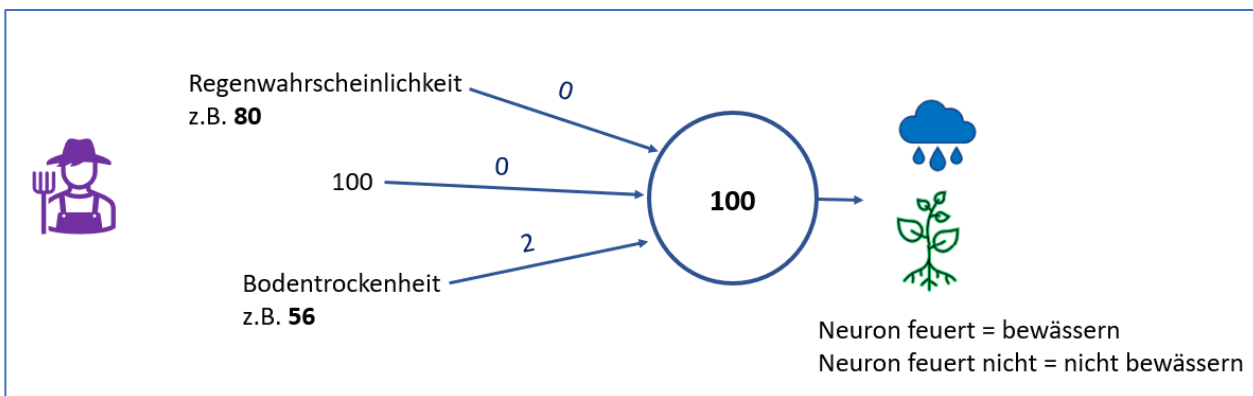


Abbildung 8: Erkunden der Funktionsweise eines Neurons

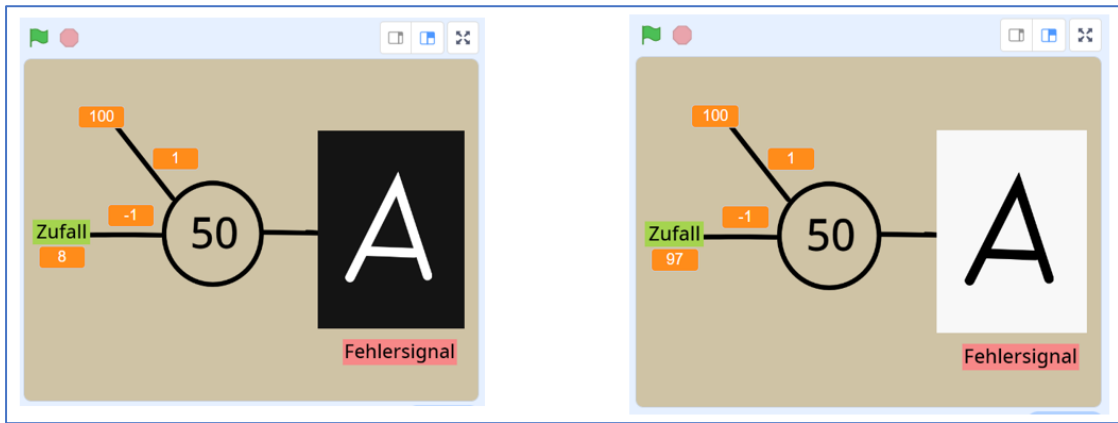


Abbildung 9: Wahl einer Schriftfarbe bei dunklem und hellem Hintergrund

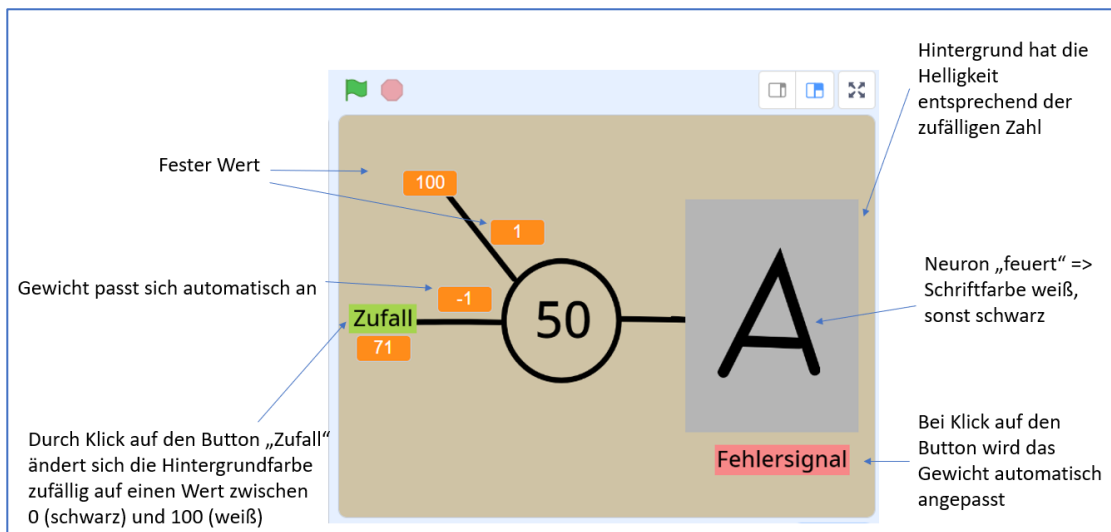


Abbildung 10: Scratch-Simulation zur automatischen Wahl einer Schriftfarbe

Auf diesen Aspekt geht das folgende Beispiel u.a. ein: Für dunklen Hintergrund eignet sich helle Schriftfarbe, für hellen Hintergrund dunkle Schriftfarbe. Bei Grautönen dazwischen können die Empfindungen individuell unterschiedlich sein.

Die Änderung des Gewichts beläuft sich dabei auf den Lernalgorithmus in Abbildung 11 und findet sich im Skript zum Button „Fehlersignal“:



Abbildung 11: Skript zum Ändern des Gewichts

Für unsere nächsten beiden Beispiele verwenden wir folgenden Kontext: Es geht um eine automatische Fenstersteuerung im Smart-Home. Es gibt zwei Sensoren: ein Sensor, der angibt, ob es draußen laut (Lautstärke=1) oder leise (Lautstärke=-1) ist und ein Sensor, der angibt, ob es draußen hell (Lichtstärke=1) oder dunkel (Lichtstärke=-1) ist. Das neuronale Netz, bzw. ein Neuron liefert den Wert 1, wenn es feuert und sonst den Wert -1. Die Ausgabe des Netzes bewirkt ein automatisches Öffnen des Fensters (wenn das Neuron der letzten Schicht feuert) oder ein Schließen des Fensters (wenn das Neuron der letzten Schicht nicht feuert). Die Begründung für die Implementierung mit einem Verfahren des maschinellen Lernens, entgegen der mit individuellen regelbasierten Algorithmen für jeden der Bewohnerinnen und Bewohner, ist folgende: Vier Bewohnerinnen und Bewohner des Hauses wollen die automatische Fensteröffnung in ihrem Schlafzimmer installieren, aber jede der Personen hat andere Wünsche, unter welchen Bedingungen sich das Fenster öffnen oder schließen soll. Implemen-

tiert man das System mit einem neuronalen Netz, kann jeder der Bewohnerinnen und Bewohner es den eigenen Wünschen entsprechend trainieren und so liefert derselbe Algorithmus (durch jeweils unterschiedliche Gewichte nach der Trainingsphase) bei jeder Person andere Ergebnisse. Funktioniert die automatische Fensteröffnung noch nicht wie gewünscht, wird das Fenster „manuell“ mit Klick auf das Fenster in der Simulation geöffnet oder geschlossen. Diese „manuelle“ Betätigung des Fensters ist das Fehlersignal, woraus geschlossen werden kann, wie die Gewichte verringert oder erhöht werden müssen.

Für ein einzelnes Neuron verwenden wir zur Anpassung der Gewichte die folgenden Lernalgorithmen mit einer Lernrate von 0,5. (Eine Erläuterung findet sich in (Eickhoff-Schachtebeck & Strecker 2025)).

alle folgenden Wünsche der Bewohnerinnen und Bewohner realisieren zu können:

Onkel Andreas: „Ich arbeite Nachtschicht. Mein Fenster soll nachts zu sein, damit keine Einbrecher einsteigen. Tagsüber schlafe ich. Da soll das Fenster nur offen sein, wenn es draußen leise ist.“ (Brandt, Eickhoff-Schachtebeck & Strecker 2022b)

Oma Emmi: „Mein Fenster soll nachts offen sein, wenn es leise draußen ist, damit ich gut schlafen kann. Bei Lautstärke kann ich nicht schlafen. Tagsüber soll das Fenster offen sein, wenn die Enkel im Garten spielen und es laut ist, damit ich sie hören kann. Ist es tagsüber leise, soll das Fenster zu bleiben.“ (Brandt, Eickhoff-Schachtebeck & Strecker 2022b)

Lin: „Ich kann nur schlafen, wenn das Fenster nachts offen und es leise ist. Bei Lärm soll das Fenster nachts zu bleiben. Tagsüber soll das Fenster zum Lüften aber immer offen stehen,



Abbildung 12: Lernalgorithmen

Die Simulation stellt sich den Lernenden wie

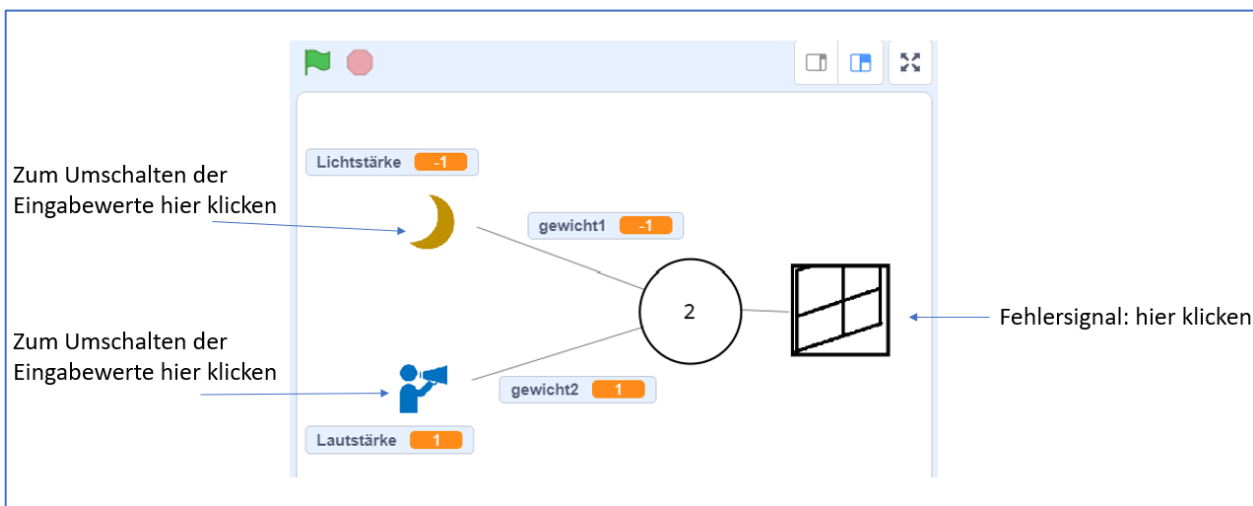


Abbildung 13: Scratch-Simulation zur automatischen Fensteröffnung mit einem Neuron

folgt dar:

Allerdings lassen sich mit einem Neuron nicht alle möglichen Wünsche und Einstellungen realisieren. Ein neuronales Netz ist notwendig, um

egal wie laut oder leise es ist. (Brandt, Eickhoff-Schachtebeck & Strecker 2022b)

Papa Erik: „Ich bin tagsüber immer bei der Arbeit und daher soll das Fenster tagsüber zu sein.

Nachts soll es aber zum Schlafen offen sein, egal wie laut oder leise es ist.“ (Brandt, Eickhoff-Schachtebeck & Strecker 2022b)

Wir verweisen an dieser Stelle darauf, dass die Anforderungen von Oma Emmi einem NOT-XOR entsprechen und damit nur mit einem neuronalen Netz implementiert werden können, nicht mit einem einzelnen Neuron.

Wir verwenden ein didaktisiertes neuronales Netz mit festen Kantengewichten in der letzten Schicht. Die Simulation stellt sich den Schülerinnen und Schülern wie folgt dar:

Durch die festen Gewichte in der letzten Schicht kann mit ähnlichen Lernalgorithmen wie in Abbildung 12 nachvollzogen werden, welche Gewichte bei einem Fehlersignal wie verändert werden müssen. Gleichzeitig können für alle denkbaren Szenarien und Wünsche der Bewohner:innen pas-

angepasst werden und die Wahl der Gewichte die Funktionalität des Netzes bestimmt, bedeutet das in der Konsequenz folgendes: wenn in der Trainingsphase ungenügende oder ungeeignete Daten verwendet werden, ist die Vorhersage des Netzes für ähnliche Daten schlecht.

- Es wird deutlich, dass das „Lernen“ eines Neurons dem automatischen Anpassen der Gewichte entspricht.
- Es wird ebenfalls deutlich, dass neuronale Netze nur diejenigen der Aufgaben sehr gut lösen können, für die sie trainiert wurden.
- Ein virtuell-enaktives Erkunden des Verfahrens neuronale Netze zeigt deutlich, dass das Verfahren algorithmisch beschreibbar ist, also nur algorithmische Grundbausteine verwendet werden und keine „Magie“.
- Skaliert man das neuronale Netz und spricht z.B. von 100 Milliarden Neuronen, dann wird

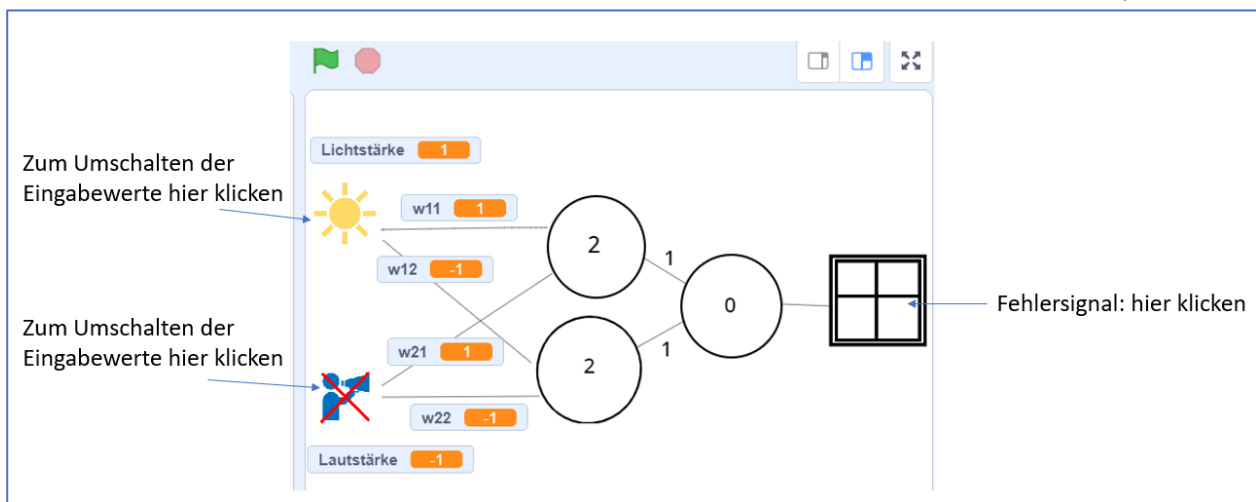


Abbildung 14: Scratch-Simulation zur automatischen Fensteröffnung mit einem neuronalen Netz

sende Gewichte automatisch „gelernt“ werden. Die theoretischen Hintergründe werden in (Eickhoff-Schachtebeck & Strecker 2025) beschrieben.

Das Erkunden der Funktionsweise eines didaktisch reduzierten neuronalen Netzes ermöglicht folgende Reflexion und hat dementsprechend folgende Lernziele:

- Die Trainingsphase wird hier sehr gut greifbar. Während bei den anderen vorgestellten Verfahren die Trainingsphase und die Anwendungsphase deutlich getrennt waren, sieht man hier, dass Trainings- und Anwendungsphase theoretisch auch miteinander verzahnt ablaufen können.
- Man sieht weiterhin sehr gut, dass die Trainingsdaten die Funktionalität bestimmen. Da die Gewichte aufgrund der Trainingsdaten

deutlich, dass das Ergebnis eines Netzes für den Menschen nicht nachzurechnen ist. Es ist also nicht nachvollziehbar, wie das Netz zu seinem Ergebnis kommt.

## Zusammenfassung der Lernziele konkreter Verfahren und Fazit

In der Einleitung haben wir gefordert, dass sich die Auswahl und Tiefe konkreter Verfahren auch daraus bestimmen sollte, welche dahinterstehenden allgemeinbildenden Lernziele abgedeckt werden. Wir fassen solche Lernziele unserer Beispiele zusammen und nennen dahinter die Verfahren, mit denen die Lernziele unserer Meinung nach besonders gut erreicht werden können:

- Entmystifizierung des Begriffs der künstlichen Intelligenz (alle)
- „Was heißt Lernen?“ (alle)
- Entscheidung beruht auf Daten der Vergangenheit (Entscheidungsbäume, k-nächste Nachbarn)
- Unterschied Ausreißer und Muster (Entscheidungsbäume, k-nächste Nachbarn)
- Ergebnisse können im individuellen Einzelfall falsch sein (k-nächste Nachbarn, Entscheidungsbäume)
- Unterscheidung regelbasierte und datenbasierte Systeme (k-nächste Nachbarn, neuronale Netze)
- Überanpassung (Entscheidungsbäume, k-nächste Nachbarn)
- Unteranpassung (k-nächste Nachbarn)
- Notwendigkeit von Trainingsdaten (alle)
- Trainingsdaten bestimmen die Ergebnisse (alle)
- Notwendigkeit der Datenaufbereitung (k-nächste Nachbarn)
- Geeignete Merkmale der Daten identifizieren (alle)
- Unterschied zwischen Korrelation und Kausalität (lineare Regression)

Wir haben drei Verfahren des überwachten Lernens im Informatikunterricht etwas genauer vorgestellt und hoffen, die dahinterstehenden Lernziele transparent gemacht zu haben. Dabei betonen wir ausdrücklich, dass diese Liste nicht vollständig ist. Sie lässt sich insbesondere dann erweitern, wenn weitere Vorgehensweisen des ML hinzugenommen werden. Betrachten wir stellvertretend das verstärkende Lernen, z.B. im Unterricht konkretisiert mit dem Q-Lernen-Verfahren (Eickhoff-Schachtebeck & Strecker 2025). Beim verstärkenden Lernen wird nicht vorgegeben, wie ein Problem gelöst werden soll, sondern es werden nur erwünschte Ergebnisse belohnt. Die Schülerinnen und Schüler machen im Informatikunterricht vor allem beim algorithmischen Problemlösen die Erfahrung, dass es unterschiedliche Lösungen für eine Problemstellung gibt, also Algorithmen nicht alternativlos sind. Wenn beim verstärkenden Lernen nur das Ergebnis belohnt wird, kann reflektiert werden, dass der „gelernte“ Weg dorthin auch ungewöhnlich sein kann oder vielleicht sogar unerwünschte Nebeneffekte hat, da nur festgelegt wird, welches Ergebnis belohnt und nicht wie es erreicht wird.

Auch das unüberwachte Lernen (z.B. konkretisiert mit dem k-Means-Clustering-Algorithmus) (siehe z.B. Eickhoff-Schachtebeck & Strecker 2025) fordert eine Reflexion über Ähnlichkeit in Daten und den Zusammenhang zwischen Ähnlichkeit und Abstand. Das wiederum kann aufgegriffen werden, wenn über eine Modellierung von Semantik

der Daten reflektiert wird, z.B. bei der Thematisierung von Grundprinzipien großer Sprachmodelle (siehe Eickhoff-Schachtebeck & Strecker 2025).

Auch wenn die vorgestellte Liste nicht vollständig ist, soll sie aber einen Anlass geben, bei der Integration des Themenbereichs maschinelles Lernen in den Informatikunterricht neben den konkreten Verfahren auch die allgemeinbildenden Aufgaben von Schule im Blick zu behalten. Für die einzelnen Verfahren sollte neben der didaktischen und methodischen Aufbereitung also stets mitgedacht werden, welche Lernziele auf der Reflexionsebene langfristig für die späteren mündigen Bürgerinnen und Bürger relevant und gewinnbringend sind und zwar unabhängig von verfahrensspezifischen Besonderheiten.

## Material

Fenstersteuerung mit Neuron in Scratch:  
<https://scratch.mit.edu/projects/1128527593>

Fenstersteuerung mit Neuronalem Netz in Scratch:  
<https://scratch.mit.edu/projects/1128528506>

Links zuletzt geprüft: 19.02.2025

## Quellen

Eickhoff-Schachtebeck, A. und Strecker, K. (2025, im Erscheinen): „Themenheft starkeSeiten Künstliche Intelligenz und Maschinelles Lernen“, Klett-Verlag, ISBN: 978-3-12-007626-1

Brandt, Y., Eickhoff-Schachtebeck, A. und Strecker, K. (2022a): „Schulbuch starkeSeiten Informatik Jahrgang 9/10 Differenzierende Ausgabe Niedersachsen“, Klett-Verlag, ISBN: 9783120075707

Brandt, Y., Eickhoff-Schachtebeck, A. und Strecker, K. (2022b): „Schulbuch starkeSeiten Informatik Jahrgang 9/10 Gymnasium Niedersachsen“, Klett-Verlag, ISBN: 9783120075721

wikipedia (2025): URL: <https://de.wikipedia.org/wiki/Scheinkorrelation> (Zugriff: 23.1.25)

## Lizenz



Dieser Artikel steht unter der Lizenz CC BY NC 4.0 zur Verfügung.

## Kontakt

Annika Eickhoff-Schachtebeck  
Regionales Landesamt für Schule und Bildung  
Braunschweig  
[annika.eickhoff-schachtebeck@rlsb.de](mailto:annika.eickhoff-schachtebeck@rlsb.de)

Kerstin Strecker  
Georg-August Universität Göttingen  
[kerstin.strecker@informatik.uni-goettingen.de](mailto:kerstin.strecker@informatik.uni-goettingen.de)