Informatik Spektrum (2022) 45:70–79
https://doi.org/10.1007/s00287-022-01438-3

HAUPTBEITRAG

Privatsphärefreundliches maschinelles Lernen

Teil 1: Grundlagen und Verfahren

Joshua Stock[1] · Tom Petersen[1] · Christian-Alexander Behrendt[2] · Hannes Federrath[1] · Thea Kreutzburg[2]

Angenommen: 30. Januar 2022 / Online publiziert: 11. März 2022
© Der/die Autor(en) 2022

Zusammenfassung
Maschinelle Lernverfahren finden seit einigen Jahren in immer mehr Bereichen vielfältige Anwendung, wodurch die Relevanz der dabei verwendeten Techniken deutlich wird. Unter dem Begriff des maschinellen Lernens (ML, oft auch „künstliche Intelligenz“) existieren zahlreiche Algorithmen, die unterschiedliche Komplexität und verschiedene Eigenschaften mit sich bringen. Für das Training dieser Algorithmen sind meist große Mengen an Daten notwendig. Insbesondere bei der Verwendung von personenbezogenen Daten stellen sich hierbei Fragen rund um den Datenschutz und die Privatsphäre von Betroffenen.
Dies ist der erste Teil eines zweiteiligen Artikels zum Thema privatsphärefreundliches ML. Dieser erste Teil bietet einen leicht verständlichen Einstieg in das Thema des ML und geht dabei auf die wichtigsten Grundbegriffe ein. Außerdem werden einige der meistverwendeten ML-Verfahren, wie Entscheidungsbäume und neuronale Netze, vorgestellt. Im zweiten Teil, der in der kommenden Ausgabe des Informatik Spektrums erscheint, werden Privatsphäreangriffe und datenschutzfördernde Maßnahmen im Kontext von ML behandelt.

Einleitung

Der Einsatz von Verfahren des maschinellen Lernens (ML) erfreut sich seit einigen Jahren immer größerer Beliebtheit in vielen Einsatzgebieten. Dieser Trend lässt sich unter anderem mit sinkenden Kosten für Rechenleistung und immer geringeren Einstiegshürden in Entwicklung und Anwendung entsprechender Algorithmen assoziieren. Je nach Einsatzgebiet können ML-Verfahren Vorhersagen unterschiedlicher Art treffen:

- Verhaltensvorhersagen (Wahrscheinlichkeit zur Einhaltung von Behandlungsempfehlungen, Rückzahlungswahrscheinlichkeit von Krediten, Rückfälligkeit von Straftätern etc.)
- Mustererkennung (z. B. Komplexität der anatomischen Verhältnisse in Schnittbildgebungen vor interventionellen Eingriffen, Gesichtserkennung auf Überwachungsvideos, Spracherkennung)
- Medizinische Entscheidungshilfen (etwa individuelle patientenzentrierte Behandlungsempfehlungen oder Risikovorhersagen für akutes Nierenversagen)

ML wird meist als Unterkategorie des breit gefassten Begriffs der künstlichen Intelligenz (KI) geführt, manchmal fälschlicherweise sogar synonym verwendet. In Kombination mit einer oftmals undifferenzierten Verwendung weiterer Fachbegriffe sorgt dies häufig für Missverständnisse bezüglich der Leistungsfähigkeit von ML-Verfahren und ihrer jeweiligen Grenzen.

In der Regel geht dem Einsatz eines ML-Systems eine Lern- bzw. Trainingsphase voraus, die unter Nutzung von Trainingsdaten ein Modell erzeugt. Kommen dabei perso-

✉ Joshua Stock
joshua.stock@uni-hamburg.de

Tom Petersen
tom.petersen@uni-hamburg.de

Christian-Alexander Behrendt
ch.behrendt@uke.de

Hannes Federrath
hannes.federrath@uni-hamburg.de

Thea Kreutzburg
t.kreutzburg@uke.de

[1] Universität Hamburg, Hamburg, Deutschland

[2] Universitätsklinikum Hamburg-Eppendorf, Hamburg, Deutschland

nenbezogene Daten zum Einsatz, kann dies vielfältige und häufig auch nicht direkt ersichtliche Gefahren für die Privatsphäre und damit für das Grundrecht auf informationelle Selbstbestimmung Betroffener bedeuten. Auch bei der anschließenden Nutzung des Modells für Vorhersagen können Fragen des Datenschutzes eine große Rolle spielen, wenn personenbezogene Daten verwendet werden. Dies ist insbesondere der Fall, wenn besonders sensible Daten wie Sozial- oder Gesundheitsdaten verarbeitet werden. Da ML-Verfahren vermehrt in der medizinischen Forschung, aber auch im Klinikalltag eingesetzt werden, ist ein umfassendes Verständnis der Datenverarbeitung und der Einsatz geeigneter Maßnahmen notwendig, um die Privatsphäre der Betroffenen zu schützen und konform mit Rechtsnormen wie der Datenschutz-Grundverordnung (DSGVO) zu agieren. Selbstverständlich werden ML-Verfahren nicht nur im medizinischen Kontext eingesetzt. Wir fokussieren die Beispiele in diesem Artikel aber auf diesen Bereich.

Dies ist der erste Teil einer zweiteiligen Artikelserie, die sich mit den Gefahren für die Privatsphäre durch den Einsatz von ML beschäftigt und Maßnahmen vorstellt, um das Risiko für Betroffene zu minimieren. In diesem Teil der Serie werden zunächst einige Grundlagen des ML und anschließend verbreitete ML-Verfahren vorgestellt. Für die jeweiligen Verfahren werden zudem Praxisbeispiele für den Einsatz im medizinischen Kontext genannt.

Ziel dieses Artikels ist es, insbesondere Nicht-Experten den Einstieg in die Thematik zu erleichtern und zur Vermeidung der oben genannten Missverständnisse beizutragen. Dieser Teil der Serie bildet damit die Basis für den in der nächsten Ausgabe erscheinenden zweiten Teil, der sich unter dem Titel *Privatsphäreangriffe und Privacy-Preserving Machine Learning* mit Angriffen auf ML-Verfahren und datenschutzgerechtem ML befasst.

Grundlagen

Dieser Abschnitt führt in die Grundlagen des maschinellen Lernens ein. Neben einer grundlegenden Kategorisierung von ML-Verfahren anhand ihres Lerntyps wird genauer auf die Trainingsphase (das „Lernen“) eingegangen. Weiterhin werden wichtige Problemfelder im ML beschrieben: Verzerrungen und Fairness, Über- und Unteranpassung sowie Erklärbarkeit.

Lerntypen

Es gibt zahlreiche Algorithmen, die alle unter dem Begriff ML zusammengefasst werden können. Sie lassen sich dabei in die drei Überkategorien überwachtes Lernen, unüberwachtes Lernen und bestärkendes Lernen (engl. *supervised*, *unsupervised* und *reinforcement learning*) einteilen [2].

Mit der ersten Kategorie, dem *überwachten Lernen*, können Klassifikationsprobleme (z. B. Mustererkennung, Prädiktion oder Ausreißererkennung) sowie Regressionsprobleme (Vorhersage von Werten) gelöst werden [2]. Algorithmen, die dabei zum Einsatz kommen, müssen auf eine Dateneingabe X eine (möglichst korrekte) Ausgabe Y liefern, die in der Regel wahrscheinlichkeitsbasiert ist. ML-Algorithmen des überwachten Lernens erlernen diese Abbildung von X zu Y in einer Trainingsphase. Im überwachten Lernen sind die in dieser Phase genutzten Daten stets mit ihrem jeweiligen Zielwert $y \in Y$ annotiert. Ein klassisches Anwendungsbeispiel ist die Entwicklung von Risikovorhersagemodellen (sog. Risikoscores) zur Abschätzung des Risikos für unerwünschte Ereignisse (z. B. für eine Herz-Kreislauf-Erkrankung) anhand von Daten in medizinischen Registern.

In der Kategorie des *unüberwachten Lernens* hingegen sind die Ausgabewerte Y zu den Trainingsdaten nicht von vornherein bekannt [2]. Ein klassisches Anwendungsbeispiel sind Clusteranalysen, die Regelmäßigkeiten in den Eingabewerten erkennen und ähnliche Datensätze in Clustern zusammenfassen. Die Cluster werden dabei dynamisch in Abhängigkeit von der jeweiligen Datenbeschaffenheit erstellt. Damit finden Methoden des unüberwachten Lernens z. B. für Genomanalysen oder Analysen zu Schnittbildern Anwendung. Weitere mögliche Anwendungen für unüberwachtes Lernen sind in der Dimensionalitätsreduktion zu finden, die häufig der Auswahl relevanter Parameter für andere ML-Verfahren dient.

Die dritte Kategorie, *bestärkendes Lernen*, ermöglicht das Lernen von Verhaltensweisen in Form von Aktionsabfolgen [2]. Ein beliebtes Anwendungsfeld sind Brettspiele. Charakteristisch für die Praktikabilität des bestärkenden Lernens ist hierbei, dass nicht durch jede einzelne Aktion (bzw. durch jeden Spielzug) unmittelbar ein positiver Effekt eintreten muss, sondern die langfristigen Strategien der Akteure den Spielausgang bestimmen. In der Trainingsphase erlernt ein ML-Algorithmus hierbei durch das wiederholte, experimentelle Anwenden verschiedener Aktionssequenzen und die gleichzeitige Beobachtung der jeweiligen Spielausgänge, welche Strategien am wirksamsten sind. Obwohl das bestärkende Lernen in der Medizin noch wenig Anwendung findet, sind zahlreiche Anwendungsbereiche denkbar. Beispielsweise könnten Abläufe in robotikbasierten Medizinprodukten optimiert werden, indem exaktere Bewegungen durch bestärkendes Lernen ermöglicht werden.

Neben diesen drei Grundparadigmen des ML existieren Mischformen, kleinere Kategorien und Unterkategorien, die den Umfang dieses Beitrags übersteigen. Für die Umsetzung der jeweiligen Lernformen stehen eine Vielzahl an Verfahren und Algorithmen zur Verfügung, die oft auch für mehrere Kategorien eingesetzt werden können.

In den derzeit gängigen Anwendungen des bestärkenden Lernens ist eine Gefährdung der Privatsphäre Betroffener nicht gegeben, da hier keine potentiell personenbezogenen Trainingsdaten benötigt werden. Dieser Artikel konzentriert sich daher auf überwachtes und unüberwachtes Lernen.

Trainings- und Testdaten

Grundsätzlich lassen sich beim Einsatz von ML-Verfahren zwei Phasen unterscheiden. Während der *Lern-* oder auch *Trainingsphase* wird ein Modell auf existierenden Trainingsdaten trainiert, um das gewünschte Vorhersageverhalten zu erlernen. In der *Inferenzphase* wird ein so trainiertes Modell dazu genutzt, das erlernte Verhalten auf von den Trainingsdaten abweichende Daten anzuwenden.

Zum Training wird ein Trainingsdatensatz $x = (x^{(1)}, ..., x^{(l)})$ verwendet, der aus l Datenpunkten $x^{(i)}$ besteht. Ein Teil dieses Datensatzes (mindestens 10%, je nach Datenbeschaffenheit auch bis zu 50%) wird nicht für das Training verwendet, sondern als separater *Testdatensatz* vorgehalten. Die Testdaten dienen dazu, die Performancesteigerung zwischen Trainingsiterationen zu quantifizieren. Sie sind außerdem wichtig, um eine etwaige Überanpassung an die Trainingsdaten zu erkennen bzw. zu verhindern (siehe Abschnitt „Varianz, Unter- und Überanpassung“). Anstatt die Teilung in Trainings- und Testdaten einmalig vorzunehmen, kann die Partitionierung auch rotieren, beispielsweise im Rahmen eines Kreuzvalidierungsverfahrens [2].

Jeder der Datenpunkte $x^{(i)} = (x_1^{(i)}, ..., x_n^{(i)})$ wird durch n Merkmale (engl. *features*) beschrieben. Hierbei kann es sich je nach Szenario um verschiedene Merkmale, wie die Medikamentdosierung während der Behandlung von Patientinnen und Patienten, die einzelnen Pixel einer medizinischen Schnittbildgebung oder auch die Freitextbeschreibungen von behandelnden Ärztinnen und Ärzten zu klinischen Beschwerden, handeln. Die Gesamtheit aller möglichen Merkmale bildet den Merkmalsraum (engl. *feature space*).

Ein Beispiel zur Verdeutlichung: Es soll ein Modell entwickelt werden, das das Risiko einer abwendbaren Amputation für Patientinnen und Patienten mit peripherer arterieller Verschlusskrankheit (PAVK, die sog. „Schaufensterkrankheit“) vorhersagt. Hierfür eignet sich der Einsatz eines überwachten Lernverfahrens. In der Trainingsphase wird ein Trainingsdatensatz genutzt, der aus einer Vielzahl von Datenpunkten besteht, die jeweils eine Patientin oder einen Patienten mit einer oder mehreren Behandlungen beschreiben. Jeder dieser Datenpunkte besteht aus den gleichen Merkmalen, wie dem Patientenalter oder Informationen über Nebenerkrankungen sowie der Annotation mit dem Zielwert – in diesem Beispiel also der Angabe, ob eine Amputation erfolgte oder nicht. Ein mit diesen Daten trainiertes Modell kann anschließend in der Inferenzphase dazu genutzt werden, für neue Datenpunkte vorherzusagen, wie hoch das Risiko für eine notwendige Amputation in der Zukunft ist.

Verzerrung und Fairness

Die Auswahl geeigneter Trainingsdaten und Verfahren ist jedoch keineswegs trivial. Ungünstige Entscheidungen können die Ursache sogenannter *Verzerrung* bilden. Verzerrung (manchmal auch systematische Abweichung oder systematischer Fehler, engl. *bias*) beschreibt die systematische Abweichung eines durch ein ML-Verfahren vorhergesagten Wertes vom tatsächlichen Wert. Eines der bekanntesten Beispiele für Verzerrungen in ML-Verfahren wurde 2016 in dem Artikel *Machine Bias* der Organisation ProPublica veröffentlicht, in dem über Bias in der zur Beurteilung von Straftätern verwendeten Software COMPAS berichtet wurde [4]. Der in COMPAS eingesetzte Algorithmus besaß signifikant unterschiedliche Falsch-Positiv-Fehlerraten für das erneute Begehen von Straftaten für verschiedene Bevölkerungsgruppen. Die Wahrscheinlichkeit eines Rückfalls für afroamerikanische Straftäter wurde im Gegensatz zu Mitgliedern anderer Bevölkerungsgruppen systematisch überschätzt.

Verzerrung kann aus vielfältigen Gründen auftreten [25]. Beispiele für häufig auftretende Verzerrungen sind:

- *Sample Bias* basiert auf einer Andersverteilung der Trainingsdaten im Vergleich zur realen Datenlage.
- *Exclusion Bias* ist bedingt durch den Ausschluss von relevanten Merkmalen vor der Trainingsphase.
- *Measurement Bias* wird durch systematisch unterschiedliche Trainings- und Produktivdaten hervorgerufen.
- *Algorithmic Bias* entsteht durch eine ungeschickte Auswahl von ML-Verfahren oder deren Parametern.

Auch bei der Anwendung von ML-Verfahren im medizinischen Bereich besteht die Gefahr verschiedenster Verzerrungen [13]. Beispielsweise werden für Patientinnen und Patienten mit einem geringeren sozioökonomischen Status weniger diagnostische Tests für chronische Erkrankungen durchgeführt. Dies kann bei der Nutzung von Datensätzen zu einer privilegierten oder unterprivilegierten Kohorte die Generalisierbarkeit auf die gesamte Zielpopulation einschränken.

Um Verzerrungen entgegenzuwirken, rückt der Begriff der *Fairness* zunehmend in den Fokus von Forschung und Praxis. Dies bezeichnet vor allem die Eigenschaft von ML-Algorithmen, niemanden zu diskrimieren. Da Diskriminierungen in vielen Bereichen des Lebens an der Tagesordnung sind, schlägt sich dies auch oft in den für das Training von ML-Algorithmen verwendeten Datensätzen nieder. Das Erreichen von Fairness ist daher nicht trivial und Forschungsgegenstand von *Ethical AI*.

Es existieren zahlreiche Frameworks wie etwa *AI Fairness 360* von IBM oder Aequitas [29], die Modellentwicklerinnen und -entwickler dabei unterstützen können, ihre Algorithmen unter verschiedenen Fairness-Aspekten zu beleuchten und zu verbessern. Eine umfassende Übersicht über Fairness in ML bietet [25].

Varianz, Unter- und Überanpassung

Zusätzlich zur Verzerrung kann auch die Varianz eines Modells betrachtet werden. Varianz kann als die Empfindlichkeit des Modells gegenüber kleinen Schwankungen in den Trainingsdaten angesehen werden. Im Bereich des überwachten Lernens tritt das sogenannte Verzerrungs-Varianz-Dilemma auf. Verfahren können entweder die Verzerrung oder die Varianz minimieren, aber nicht beides.

In der Praxis führt dieses Dilemma dazu, dass es beim Training von Modellen zu *Unteranpassung* oder *Überanpassung* kommen kann. Unteranpassung (engl. *underfitting*) beschreibt ein Modell mit großer Verzerrung, dessen Vorhersagen den echten statistischen Zusammenhang nicht abbilden. Überanpassung (engl. *overfitting*) beschreibt ein Modell mit großer Varianz, das zu sehr auf den Trainingsdaten beruht und beispielsweise vorliegende Messfehler oder Rauschen mit einbezieht. Eine Visualisierung dieser Konzepte für Regressions- und Klassifikationsprobleme ist in Abb. 1 dargestellt.

Erklärbares maschinelles Lernen

Bereits die Identifikation von problematischen Entscheidungen während der Entwicklung und Kalibrierung eines ML-Verfahrens vor oder während der Trainingsphase, die sich etwa in Verzerrungen, Unter- oder Überanpassungen manifestieren, ist häufig schwierig. Dies liegt insbesondere darin begründet, dass die Funktionsweisen vieler ML-Verfahren für Menschen nicht direkt verständlich sind. Erklärbares maschinelles Lernen (engl. *interpretable or explainable ML/AI, XML/XAI*) beschreibt Methoden, die die von einem ML-Verfahren getroffenen Vorhersagen für menschliche Benutzer nachvollziehbar machen [10]. Erklärbarkeit ist keine notwendige Eigenschaft, falls der Einsatz von fehlerhaften ML-Modellen keine signifikanten Auswirkungen im Einsatzkontext hat. Gerade beim Einsatz neuer Verfahren in kritischen Kontexten kann die Nachvollziehbarkeit von Entscheidungen eines Modells jedoch essenziell sein, um das Vertrauen in die Technik im Sinne einer Qualitätssicherung zu stärken oder ihren Einsatz überhaupt erst zu ermöglichen. Die Erklärbarkeit kann auch dabei unterstützen, Verzerrungen in Modellen aufzudecken und so zwischen einfacher Korrelation und Kausalität unterscheiden zu können. Ein Beispiel für eine ungewollte Korrelation, die durch den Einsatz von Erklärverfahren aufgedeckt wer-

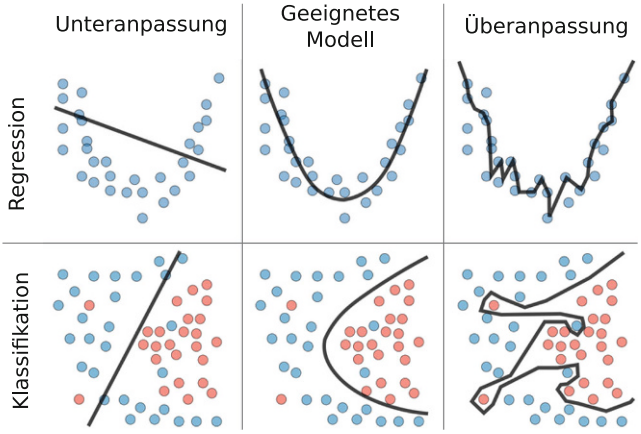

Abb. 1 Visualisierung von Unteranpassung, einem geeignetem Modell und Überanpassung für ein Regressions- und ein Klassifikationsproblem (Klassifizierung in rote und blaue Klasse). Abbildung angelehnt an [3]

den konnte, ist in Abb. 2 abgebildet. Die Nachvollziehbarkeit kann damit insgesamt zu robusteren Modellen führen, aber auch ein tieferes Verständnis für bisher unbekannte Zusammenhänge fördern [16]. Dieses Problemfeld ist auch im Hinblick auf die in der DSGVO geforderte verständliche Erklärung der Entscheidungsfindung in automatisierten Entscheidungsprozessen (Artikel 13-15, 22 der DSGVO) relevant [17].

Die Verfahren des erklärbaren maschinellen Lernens lassen sich auf verschiedene Weisen kategorisieren [26]. *Lokale* Erklärungsmodelle bieten Erklärungen dafür, wie ein Modell eine Vorhersage für einen spezifischen Datensatz getroffen hat, während *globale* Erklärungsmodelle Erklärungen dafür liefern, wie ein Modell insgesamt Entscheidungen trifft oder zumindest welche Auswirkungen bestimmte Teile eines Modells (beispielsweise einzelne Gewichte) hervorrufen. Es kann weiterhin zwischen *intrinsisch erklärbaren Verfahren* und *Post-hoc-Erklärungen* unterschieden werden. Intrinsisch erklärbare Verfahren sind in ihrer Struktur so einfach, dass sie als menschlich interpretierbar angesehen werden. Beispiele hierfür sind Entscheidungsbäume oder einfache lineare Regressionen. Post-hoc-Erklärungen sind Methoden, die nach der Trainingsphase eines Modells angewendet werden und etwa die Wichtigkeit einzelner Eingangsfeatures bewerten können. Eine weitere Möglichkeit der Differenzierung besteht zwischen *modellspezifischen* und *modellagnostischen* Verfahren – je nachdem, ob Erklärungen nur für bestimmte Arten von Modellen generiert werden können oder ein generischer Ansatz verfolgt wird.

Beispiele für Verfahren im Bereich des erklärbaren maschinellen Lernens sind *Local Interpretable Model-agnostic Explanations (LIME)* [28], *Shapley Additive Explanations (SHAP)* [23], *Gradient-weighted Class Activation Mapping (Grad-CAM)* [31] und *Layer-Wise Relevance Propagation*

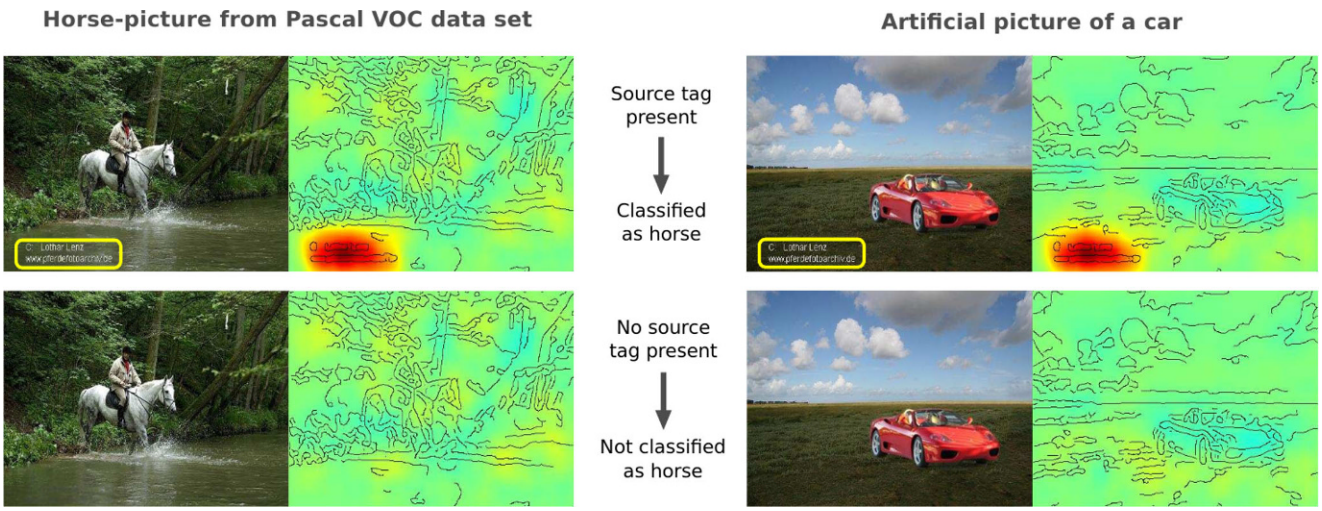

Abb. 2 Im Rahmen der *PASCAL Visual Object Classes Challenge 2007* wurde ein Modell für die Objektklassifizierung in Bilddaten erstellt. Da viele der enthaltenen Bilder von Pferden den Namen des Fotografen enthielten, lernte das Modell diesen Zusammenhang. Wurde der Namenszug auf das Bild eines Autos gesetzt, so wurde auch dieses fälschlicherweise als Pferd klassifiziert. Durch ein *Heatmap*-basiertes Erklärungsverfahren wird dieser ungewollte Zusammenhang deutlich. Abbildung entnommen aus [22]

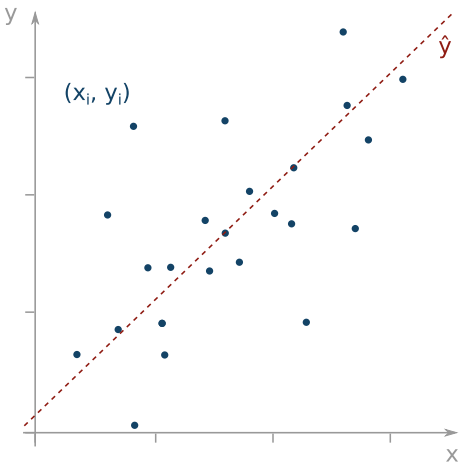

Abb. 3 Beispiel einer einfachen linearen Regression. Die *blauen Punkte* stellen die einzelnen Trainingsdaten dar, die *rote Linie* die durch die Trainingsdaten bedingte Vorhersage

(LRP) [5]. Eine Übersicht über den aktuellen Stand der Erklärbarkeit von ML-Verfahren und auch beispielhafte Anwendungsfälle aus dem medizinischen Bereich lassen sich in [20] finden.

Verfahren

Nachdem die Grundlagen des Feldes ML beschrieben wurden, widmet sich der folgende Abschnitt konkreten Verfahren aus den Bereichen des überwachten und unüberwachten Lernens. Da eine große Vielfalt an Verfahren und Varianten existiert, werden hier nur die grundlegenden Funktionsweisen einiger gängiger Möglichkeiten beschrieben. Wenn die vorgestellten Verfahren produktiv eingesetzt werden sollen, kann im Normalfall auf existierende Implementierungen in Form von Softwarebibliotheken wie *TensorFlow* [1] oder *PyTorch* [27] zurückgegriffen werden.

Lineare Regression

Lineare Regressionsverfahren dienen der überwachten Regression von Trainingsdaten. Die Verfahren wählen dabei Linearparameter β, sodass die für einen Trainingsdatensatz $x_i = (x_1^{(i)}, ..., x_n^{(i)})$ beobachteten Werte y_i und die durch die Linearfunktion berechneten Werte $\widehat{y}_i = \beta_0 + \beta_1 \cdot x_1^{(i)} + ... + \beta_n \cdot x_n^{(i)}$ möglichst wenig voneinander abweichen. Eine einfache lineare Regression ist in Abb. 3 dargestellt.

Eine mögliche Variante ist die sogenannte *Methode der kleinsten Quadrate*, bei der für die Abweichung die Summe der Fehlerquadrate betrachtet wird. Es entsteht ein Minimierungsproblem:

$$\min_{\beta} \sum_{i=1}^{l} (\widehat{y}_i - y_i)^2$$

Da dieses Verfahren in seiner einfachsten Form zu *Overfitting* neigt, können zusätzliche Nebenbedingungen, sogenannte Strafterme, hinzugefügt werden. Man spricht hierbei auch von *Regularisierung*. Beispiele hierfür sind Ridge- und LASSO-Regularisierung [15, 32].

Dank ihrer generischen Funktionsweise können lineare Regressionsverfahren auf zahlreiche Art in statistischen Analysen im Medizinkontext eingesetzt werden. Neben der bloßen Beschreibung von Zusammenhängen zwischen beobachteten Werten sind auch Schätzungen und Prognosen der Zielvariablen möglich, z. B. bei der Mortalitätswahr-

scheinlichkeit von Patientinnen und Patienten mit sorgfältig ausgewählten Risikofaktoren [6, 21].

k-Nearest-Neighbour

Der *k-Nearest-Neighbour*-Algorithmus (kNN, dt. k nächste Nachbarn) kann zur überwachten Klassifikation oder Regression eingesetzt werden. Die Grundidee besteht darin, für einen Datenpunkt die k ähnlichsten Datenpunkte (Nachbarn) innerhalb der Trainingsdaten zu finden und als Ergebnis die häufigste Klasse (Klassifikation) oder den Durchschnitt der Zielwerte (Regression) dieser Nachbarn zu verwenden. Abb. 4 stellt das Verfahren für die Klassifikation dar.

In der Praxis sind hierbei verschiedene Entscheidungen zu treffen. Neben der Wahl eines geeigneten Parameters k und eines geeignetes Maßes zur Bestimmung des Abstands zweier Datensätze entscheidet insbesondere auch die Auswahl geeigneter Features über die Qualität der Ergebnisse. Um eine bessere Performance zu erreichen, können vor der eigentlichen Ausführung auch wenige aussagekräftige Vertreter aus den Trainingsdaten ausgewählt werden, die dann zur Bestimmung der Nachbarn genutzt werden. Auf diese Weise müssen weniger Distanzen berechnet werden. Alternativ hierzu steht eine Vielzahl von Verfahren zur Verfügung, die durch den geschickten Einsatz von Datenstrukturen oder Heuristiken ein schnelleres Finden der Nachbarn erlauben [7].

In der Medizin können k-Nearest-Neighbour-Klassifikatoren unter anderem dafür eingesetzt werden, Zusammenhänge zwischen ähnlichen Symptomatiken und Krankheitsbildern festzustellen [18].

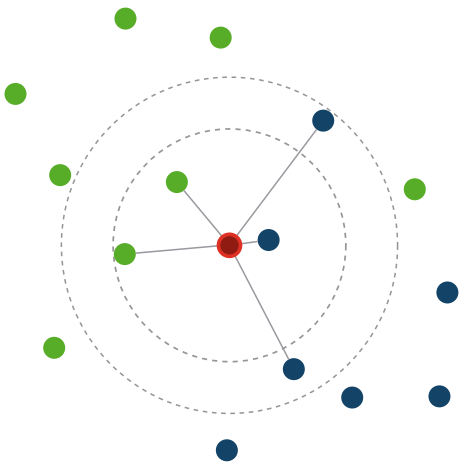

Abb. 4 k-Nearest-Neighbour-Klassifikation für die Werte $k = 3$ und $k = 5$. Für die unterschiedlichen Werte ergeben sich für die Klassifizierung des rot umrandeten Zieldatensatzes unterschiedliche Ergebnisklassen: Für $k = 3$ wird das grüne Klassenlabel vergeben, für $k = 5$ wird der Datensatz als der blauen Klasse zugehörig klassifiziert

k-Means-Clustering

Der k-Means-Algorithmus wird zum unüberwachten Clustering von Daten verwendet [24]. Ziel ist es, Daten derartig in k Cluster $S = \{S_1, ..., S_k\}$ mit zugehörigen Cluster-Mittelwerten μ_i (auch Schwerpunkte oder Zentroide genannt) zu partitionieren, dass die Summe aller Abweichungen von diesen Mittelwerten minimal ist. Abb. 5 stellt ein mögliches Resultat des Algorithmus für drei Cluster dar. Es ergibt sich das Optimierungsproblem der Minimierung von

$$\sum_{j=1}^{k} \sum_{x_i \in S_j} \| x_i - \mu_j \|^2$$

über die Zuweisung der Daten x_i zu den Clustern S_j. Das Finden einer optimalen Lösung für dieses Problem ist schwierig, es existieren jedoch viele heuristische Verfahren, wie etwa Lloyd's Algorithmus. Hierbei werden initial k Mittelwerte zufällig gewählt (z. B. schlicht k zufällige Datensätze). Anschließend wird jeder Datensatz jeweils dem am nächsten liegenden Mittelwert zugewiesen und es werden für die so entstehenden Cluster neue Mittelwerte berechnet. Dieser Vorgang wird so lange wiederholt, bis sich die Zuweisung von Datensätzen zu Clustern nicht mehr ändert.

Unüberwachte Clusteralgorithmen wie der k-Means-Algorithmus können in der Medizin unter anderem zur Partitionierung von Patientengruppen eingesetzt werden, etwa bei Alzheimerpatientinnen und -patienten. Die maschinelle Verarbeitung von Patientendaten kann hierbei deutlich komplexere Zusammenhänge berücksichtigen als eine manuelle Analyse. Anhand der gefundenen Gruppen können anschließend zielgerichtete medizinische Studien für verbesserte Diagnostik- und Therapiemöglichkeiten durchgeführt werden.

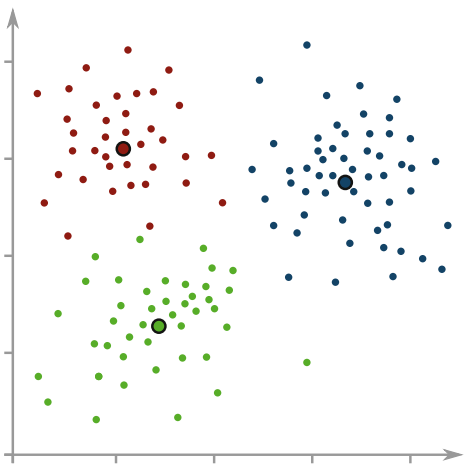

Abb. 5 Beispiel eines k-Means-Ergebnisses für $k = 3$. Die Cluster-Mittelwerte werden durch die *gerahmten Datenpunkte* dargestellt

Entscheidungsbäume und Random Forest

Entscheidungsbäume dienen der überwachten Klassifikation (*classification trees*) und Regression (*regression trees*) [8]. In Abb. 6 ist ein einfacher Entscheidungsbaum für die Klassifikation abgebildet. Während der Trainingsphase wird der Entscheidungsbaum in einem *top-down*-Ansatz vom Wurzelknoten aus aufgebaut. Hierzu werden rekursiv Knoten anhand eines Merkmals partitioniert, bis die entstehenden Blattknoten ausreichend homogen in Bezug auf die Zielklasse oder den Zielwert sind. Datensätze werden während der Inferenzphase ausgehend vom Wurzelknoten eines Baums bis zu einem Blattknoten bewertet. Hierzu werden an den Knoten des Baums Entscheidungen in Bezug auf ein Feature getroffen. Diese Entscheidungen ergeben einen Pfad bis zu einem Blattknoten, der die Klasse oder den Zielwert für den Datensatz beschreibt. Da ein optimaler Entscheidungsbaum schwierig zu berechnen ist, werden in der Praxis heuristische Ansätze wie CART verwendet [8].

Entscheidungsbäume bilden ein leicht zu verstehendes und gut zu visualisierendes Verfahren, das dadurch auch leicht zu interpretieren ist [19]. In ihrer einfachsten Form neigen sie jedoch insbesondere bei der Betrachtung von vielen Features zur Überanpassung. Abhilfe kann der Einsatz mehrerer, separat trainierter Entscheidungsbäume schaffen, aus deren Ergebnissen eine Mehrheitsentscheidung getroffen wird. Dieses Verfahren wird *Random Forest* genannt. Das Training der verschiedenen Bäume erfolgt hierbei auf einer zufälligen Menge der Trainingsdaten und mit einer Zufallsauswahl relevanter Merkmale für die Partitionierungen. Hierdurch werden unterschiedliche Entscheidungsbäume erreicht. Die Bewertung eines Datensatzes in der Inferenzphase erfolgt durch separate Bewertung in jedem Baum. Das finale Ergebnis für den Datensatz bildet die Klasse, die am häufigsten von den einzelnen Entscheidungsbäumen als Resultat berechnet wurde.

Aufgrund ihrer transparenten Entscheidungsfindung wurden Entscheidungsbäume bereits in zahlreichen medizinischen Kontexten eingesetzt, vor allem im Bereich der Diagnostik. Beispielsweise wurde bereits vor vielen Jahren gezeigt, dass Klassifikationsbäume Herzinfarkte bei entsprechender Datenverfügbarkeit mit 94-prozentiger Genauigkeit (*Fläche unter der Kurve*) richtig diagnostizieren können [33]. Insbesondere als Ergänzung zu herkömmlichen Methoden der medizinischen Diagnostik birgt dies große Chancen.

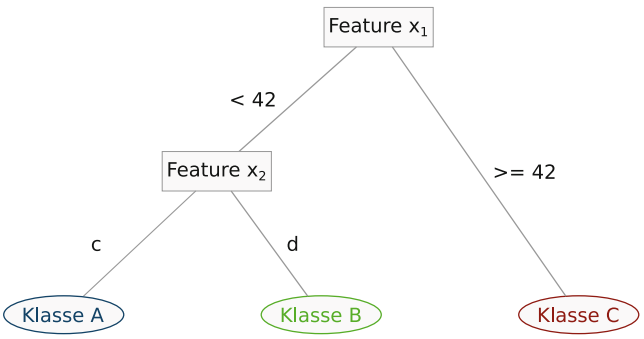

Abb. 6 Ein einfacher Entscheidungsbaum, der Datensätze basierend auf den zwei Features x_1 und x_2 in drei Klassen A, B und C einordnet

Support Vector Machines

Support Vector Machines (SVMs, dt. *Stützvektormaschinen*) werden zur überwachten Klassifikation eingesetzt [9]. Grundsätzlich sind SVMs nur für binäre Klassifikationsaufgaben (mit zwei möglichen Klassen) geeignet. Die grundlegende Idee besteht darin, in der Trainingsphase eine bestmögliche Trennung der Trainingsdaten in die beiden (binären) Klassen durch Hyperebenen im Merkmalsraum zu finden. Hierzu wird eine Hyperebene ermittelt, die eine maximale Distanz zu den ihr am nächsten liegenden Trainingsdatensätzen der entsprechenden Klassen erreicht. Die orthogonalen Stützvektoren (engl. *Support Vectors*) zwischen Hyperebene und diesen Trainingsdatensätzen geben dem Verfahren seinen Namen. Durch Maximierung der Distanz sollen gute Ergebnisse für spätere Klassifikationen erreicht werden, selbst wenn in den Trainingsdaten unähnliche Daten verwendet werden. Abb. 7 zeigt ein einfaches Beispiel inklusive der trennenden Hyperebene (in diesem Fall eine Gerade) und der Stützvektoren. Da viele Klassifizierungsprobleme nicht linear lösbar sind, können sich SVMs des sogenannten *Kernel-Tricks* bedienen, der die Daten in höhere Dimensionen abbildet, in denen eine lineare Trennung möglich ist. Sogenannte Kernel-Funktionen ermöglichen dabei die implizite Berechnung von für die SVM relevanten Vektorprodukten, ohne die Abbildung in höhere Dimensionen explizit machen zu müssen [14].

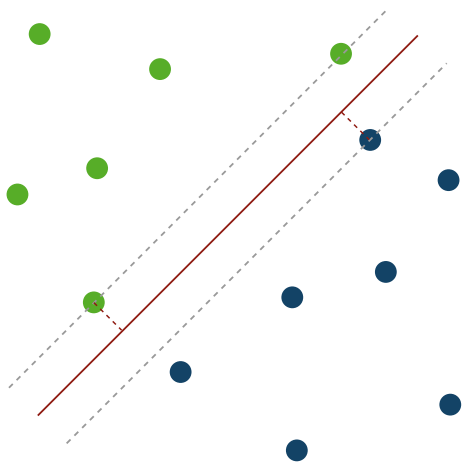

Abb. 7 Darstellung einer einfachen Klassifikation durch eine *Support Vector Machine* mit den dazugehörigen *Support Vectors*

Es existieren sowohl Erweiterungen für die Klassifikation mit mehr als zwei Klassen, indem mehrere Einzelprobleme als *Ist-in-Klasse vs. Ist-nicht-in-Klasse* betrachtet und die Einzelergebnisse geeignet verknüpft werden, als auch Erweiterungen zur Nutzung des Prinzips für die Regressionsanalyse [11]. Ein wesentlicher Vorteil von SVMs besteht darin, dass sie auch für hochdimensionale Daten, also Daten, die viele Merkmale besitzen, und wenige Trainingsdatensätze gute Ergebnisse liefern können.

In der Medizin können SVM z. B. zur Klassifizierung von Diabetikerinnen und Diabetikern sowie Menschen mit einer Vorstufe von Diabetes genutzt werden. Dies kann bei einer datenbasierten, frühzeitigen Erkennung von Diabetes bereits vor den klinisch apparenten Stadien helfen und somit eine frühe Therapie ermöglichen [35].

Neuronale Netze

Neuronale Netze in unterschiedlichen Ausprägungen haben eine breite Anwendbarkeit in verschiedenen Aufgaben des maschinellen Lernens. Neben überwachter Klassifikation und Regression, unüberwachtem Clustering und der Detektion von Anomalien in Daten können mit ihrem Einsatz auch (im echten Wortsinne) übermenschliche Leistungen in klassischen Brettspielen wie Schach und Go erzielt werden.

Die grundlegende Idee ist inspiriert vom Aufbau und von den Vorgängen im menschlichen Gehirn. Die kleinsten Bestandteile künstlicher neuronaler Netze bilden sogenannte Neuronen, die üblicherweise in Schichten angeordnet und untereinander durch Kanten verbunden sind. Ein Netz besteht, wie in Abb. 8 dargestellt, aus einer Eingabeschicht (engl. *input layer*) (beispielsweise bestehend aus einem Neuron pro Pixel eines Eingabebildes), einer Menge von Zwischenschichten (engl. *hidden layers*) und einer Ausgabeschicht (engl. *output layer*), die beispielsweise ein Neuron pro möglicher Klasse in einem Klassifizierungsproblem enthält. Werden viele Zwischenschichten verwendet, so wird auch von *Deep Neural Networks* oder allgemeiner von *Deep Learning* gesprochen [30]. Ähnlich zu den Vorgängen im Gehirn führen verschiedene Eingaben zu unterschiedlich starken Aktivierungen der Neuronen. Abhängig von ihrer Aktivierungsintensität geben die Neuronen über ihre Kantenverbindungen Signale an die nächste Schicht weiter. Das Ergebnis kann letztlich aus der Aktivierung der Ausgabeneuronen in der letzten Schicht abgelesen werden.

Im Detail wird die jeweilige Aktivierung folgendermaßen berechnet: Ein Neuron empfängt zunächst eine Menge von Eingaben I und berechnet daraus mithilfe einer nichtlinearen Aktivierungsfunktion Φ über die Summe aller Eingaben seine Ausgabe O. Beliebte Aktivierungsfunktionen sind *ReLU* (engl. *Rectified Linear Unit*) und *Sigmoid*. Erstere schneidet mittels $\Phi_{\text{ReLU}}(x) = max(0, x)$ effektiv alle Werte < 0 ab, sodass die Funktionsausgabe immer positiv ist. Die Sigmoid-Funktion $\Phi_{\text{sig}}(x) = \frac{1}{1+e^{-x}}$ bildet alle Eingaben auf einen Wert zwischen 0 und 1 ab. Die beiden Funktionsgraphen sind in der Abb. 9 dargestellt.

Die Eingaben für einzelne Neuronen I werden jeweils mit Gewichten w_i (von engl. *weights*) versehen und zusätzlich mit einem sogenannten Bias-Term b addiert:

$$O = \Phi(b + \sum_{i=1}^{k} w_i \cdot I_i)$$

Zu Beginn des Trainingsprozesses eines neuronalen Netzes werden die Gewichte meist mit Zufallswerten initialisiert. Die Trainingsphase besteht nun darin, dass das Netz mit Trainingsdaten gespeist wird und unter Beobachtung der Ausgabewerte die Modellparameter (Kantengewichte und Bias-Terme) angepasst werden. Als Ziel der Anpassung dient die Minimierung des beobachteten Fehlers, beispiels-

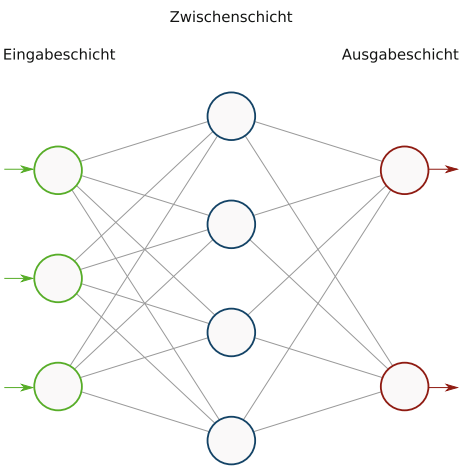

Abb. 8 Schematische Darstellung der Schichten eines neuronalen Netzes

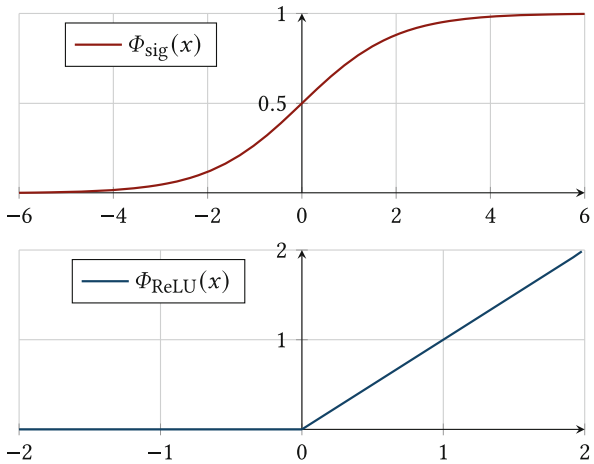

Abb. 9 Funktionsgraphen der Aktivierungsfunktionen Sigmoid $\Phi_{\text{sig}}(x) = \frac{1}{1+e^{-x}}$ und ReLU $\Phi_{\text{ReLU}}(x) = max(0, x)$, die innerhalb von Neuronen zum Einsatz kommen können

weise der Abweichung des Ergebnisses zur echten Klasse des Trainingsdatensatzes in einem überwachten Lernprozess. Dieser Prozess wird *Back-Propagation* genannt, da der Fehler von der letzten Schicht über die Zwischenschichten bis zu den Kantengewichten der Eingabeschicht rückwärts durch das Netz propagiert wird und dabei die für die jeweilige Entscheidung relevanten Gewichte angepasst werden.

Es gibt zahlreiche Strategien für einen effizienten und effektiven Trainingsprozess, die je nach Aufgabengebiet, Datenbeschaffenheit und Kontext variieren. Eine beliebte Technik ist die Aufteilung der Trainingsdaten in sogenannte *(Mini-)Batches*, um die Modellparameter nicht nach jedem einzelnen Datenwert, sondern erst nach Beobachtung des Netzes bei der Eingabe eines ganzen *Batches* anzupassen. Zusätzlich werden die Trainingsdaten wiederholt in mehreren Durchläufen (engl. *epochs*) verarbeitet, in der Regel einmal pro Durchlauf. Dabei kann die Zusammenstellung der *Batches* auch nach jedem Durchlauf variiert werden, um die Auswirkungen unausgeglichener *Batch*-Zusammenstellungen zu minimieren. Auch die Lernrate (engl. *learning rate*), also das Flexibilitätsmaß der Modellparameter, mit dem die beobachteten Fehler korrigiert werden, kann im Verlauf des Lernprozesses geändert werden. In der Regel wird mit einer höheren Lernrate begonnen, um die initialen Zufallswerte der Modellparameter schnell zu korrigieren. Nach einigen Durchläufen kann die Lernrate reduziert werden, um das Modell schrittweise gegen ein lokales Optimum (hinsichtlich der Fehlerminimierung) konvergieren zu lassen.

Neben der Lernrate und der Anzahl von *Batches* und Trainingsdurchläufen muss auch der Aufbau eines Netzes (die Anzahl der Neuronenschichten, die Anzahl der Neuronen pro Schicht und weitere Eigenschaften) im Rahmen des Entwurfs und der Entwicklung eines neuronalen Netzes festgelegt werden; diese Werte werden auch als *Hyperparameter* bezeichnet. Die Auswahl der richtigen Hyperparameter erfordert in der Regel viel Geschick und Übung. Neben der händischen Auswahl dieser Hyperparameter existieren jedoch bereits Ansätze unter dem Oberbegriff *automated machine learning (AutoML)*, um auch diesen Schritt zu automatisieren.

Im Vergleich zu anderen ML-Verfahren sind die Vorgänge in neuronalen Netzen schwieriger nachzuvollziehen und zu erklären. Weiterhin benötigen sie im Normalfall eine große Menge von Trainingsdaten und der Ressourcenbedarf im Trainingsprozess liegt deutlich über dem anderer Methoden.

Dennoch können neuronale Netze auf vielfältige Weise in der Medizin eingesetzt werden. Einsatzgebiete sind die Erkennung von Auffälligkeiten im Rahmen der bildgebenden Diagnostik [34] oder die Filterung von Stör- und Hintergrundgeräuschen in Hörgeräten [12]. Aktuelle Projekte beschäftigen sich mit der Erkennung von Gefäßen in Schnittbildgebungen ohne Kontrastmittel, was zu einer Vermeidung von kontrastmittelassoziierten Komplikationen im Rahmen dieser Standardbildgebung führen könnte. In der Praxis scheitert ihr Einsatz in kritischen Gesundheitsbereichen jedoch häufig an den hohen Anforderungen an medizinische Softwareprodukte. Vor allem die mangelnde Transparenz der Entscheidungsfindung kann problematisch sein, wobei Techniken des erklärbaren ML (siehe Abschn. 2.5) eventuell Abhilfe schaffen können.

Fazit und Ausblick

In diesem ersten Teil der Artikelserie wurden die Grundlagen von ML aufbereitet und einige verbreitete Verfahren vorgestellt. Es zeigt sich, dass dem Begriff ML vielerlei Verfahren untergeordnet werden können und bei ihrem Einsatz diverse Probleme auftreten können. Dass zusätzlich auch Gefahren für die Privatsphäre Betroffener bestehen und wie diesen Gefahren begegnet werden kann, wird im Mittelpunkt des in der nächsten Ausgabe erscheinenden Folgeartikels stehen. Dabei werden insbesondere Privatsphäreangriffe auf ML-Verfahren wie *Model Inversion* und Techniken des Privacy-Preserving Machine Learning wie *Federated Learning* behandelt.

Förderung Diese Publikation entstand im Rahmen des vom Innovationsfonds des Gemeinsamen Bundesausschusses geförderten fakultätsübergreifenden Konsortialprojektes RABATT (01VSF18035). Ein Autor wird aus Mitteln des Bundesministerium für Wirtschaft und Klimaschutz im Rahmen des ZIM-Projektes PANDA (ZF4498402DH9) gefördert.

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access Dieser Artikel wird unter der Creative Commons Namensnennung 4.0 International Lizenz veröffentlicht, welche die Nutzung, Vervielfältigung, Bearbeitung, Verbreitung und Wiedergabe in jeglichem Medium und Format erlaubt, sofern Sie den/die ursprünglichen Autor(en) und die Quelle ordnungsgemäß nennen, einen Link zur Creative Commons Lizenz beifügen und angeben, ob Änderungen vorgenommen wurden.

Die in diesem Artikel enthaltenen Bilder und sonstiges Drittmaterial unterliegen ebenfalls der genannten Creative Commons Lizenz, sofern sich aus der Abbildungslegende nichts anderes ergibt. Sofern das betreffende Material nicht unter der genannten Creative Commons Lizenz steht und die betreffende Handlung nicht nach gesetzlichen Vorschriften erlaubt ist, ist für die oben aufgeführten Weiterverwendungen des Materials die Einwilligung des jeweiligen Rechteinhabers einzuholen.

Weitere Details zur Lizenz entnehmen Sie bitte der Lizenzinformation auf http://creativecommons.org/licenses/by/4.0/deed.de.

Literatur

1. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M et al (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems (arXiv: 1603.04467)
2. Alpaydin E (2019) Maschinelles Lernen. De Gruyter Oldenbourg, München
3. Amidi A, Amidi S (2021) Machine learning cheat sheet. https://github.com/afshinea/stanford-cs-229-machine-learning. Zugegriffen: 24. Nov. 2021
4. Angwin J, Larson J, Mattu S, Kirchner L (2016) Machine bias. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing. Zugegriffen: 1. Apr. 2021
5. Bach S, Binder A, Montavon G, Klauschen F, Müller KR, Samek W (2015) On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS ONE. https://doi.org/10.1371/journal.pone.0130140
6. Behrendt CA, Kreutzburg T, Nordanstig J, Twine CP, Marschall U, Kakkos S, Aboyans V, Peters F (2022) The OAC3-PAD risk score predicts major bleeding events one year after hospitalisation for peripheral artery disease. Eur J Vasc Endovascular Surg. https://doi.org/10.1016/j.ejvs.2021.12.019
7. Bentley JL (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18(9):509–517
8. Breiman L, Friedman J, Stone CJ, Olshen RA (1984) Classification and regression trees. CRC press, Boca Raton
9. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
10. Doshi-Velez F, Kim B (2017) Towards a rigorous science of interpretable machine learning (arXiv: 1702.08608)
11. Drucker H, Burges CJC, Kaufman L, Smola A, Vapnik V (1996) Support vector regression machines. MIT Press, Cambridge
12. Fang H, Carbajal G, Wermter S, Gerkmann T (2021) Variational autoencoder for speech enhancement with a noise-aware encoder. In: IEEE International Conference on Acoustics, Speech and Signal Processing, S 676–680
13. Gianfrancesco MA, Tamang S, Yazdany J, Schmajuk G (2018) Potential biases in machine learning algorithms using electronic health record data. JAMA 178(11):1544–1547
14. Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning: Data mining, inference, and prediction. Springer, New York
15. Hoerl AE, Kennard RW (1970) Ridge regression: Biased estimation for nonorthogonal problems. Technometrics 12(1):55–67
16. Holzinger A (2018) Explainable AI (ex-ai). Inform Spektrum 41(2):138–143
17. Holzinger A, Biemann C, Pattichis CS, Kell DB (2017) What do we need to build explainable ai systems for the medical domain? (arXiv: 1712.09923)
18. Khamis HS, Cheruiyot KW, Kimani S (2014) Application of k-nearest neighbour classification in medical data mining. Int J Inf Commun Technol Res 4(4):121–128
19. Knuth T (2021) Lernende Entscheidungsbäume. Inform Spektrum 44(5):364–369
20. Kraus T, Ganschow L, Eisenträger M, Wischmann S (2021) Erklärbare KI: Anforderungen, Anwendungsfälle und Lösungen. https://www.digitale-technologien.de/DT/Redaktion/DE/Downloads/Publikation/KI-Inno/2021/Studie_Erklaerbare_KI.html. Zugegriffen: 27. Mai 2021
21. Kreutzburg T, Peters F, Kuchenbecker J, Marschall U, Lee R, Kriston L, Debus ES, Behrendt CA (2021) Editor's choice – The GermanVasc score: A pragmatic risk score predicts five year amputation free survival in patients with peripheral arterial occlusive disease. Eur J Vasc Endovascular Surg 61(2):248–256
22. Lapuschkin S, Wäldchen S, Binder A, Montavon G, Samek W, Müller KR (2019) Unmasking Clever Hans predictors and assessing what machines really learn. Nat Commun 10(1):1–8
23. Lundberg SM, Erion GG, Lee SI (2019) Consistent individualized feature attribution for tree ensembles (arXiv: 1802.03888)
24. MacQueen J et al (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Bd. 1
25. Mehrabi N, Morstatter F, Saxena N, Lerman K, Galstyan A (2021) A survey on bias and fairness in machine learning. ACM Comput Surv 54(6):1–35
26. Molnar C (2021) Interpretable machine learning. https://christophm.github.io/interpretable-ml-book/. Zugegriffen: 15. März 2021
27. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L et al (2019) Pytorch: An imperative style, high-performance deep learning library. Adv Neural Inf Process Syst 32:8026–8037
28. Ribeiro MT, Singh S, Guestrin C (2016) Why should I trust you? Explaining the predictions of any classifier. In: 22nd ACM SIGKDD international conference on knowledge discovery and data mining
29. Saleiro P, Kuester B, Hinkson L, London J, Stevens A, Anisfeld A, Rodolfa KT, Ghani R (2019) Aequitas: A bias and fairness audit toolkit (arXiv: 1811.05577)
30. Schmidhuber J (2015) Deep learning in neural networks: An overview. Neural Networks 61:85–117
31. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-CAM: Visual explanations from deep networks via gradient-based localization. In: IEEE international conference on computer vision
32. Tibshirani R (1996) Regression shrinkage and selection via the lasso. J R Stat Soc Series B Stat Methodol 58(1):267–288
33. Tsien CL, Fraser HS, Long WJ, Kennedy RL (1998) Using classification tree and logistic regression methods to diagnose myocardial infarction. In: MEDINFO'98. IOS, Amsterdam, S. 493–497
34. Tuladhar A, Schimert S, Rajashekar D, Kniep HC, Fiehler J, Forkert ND (2020) Automatic segmentation of stroke lesions in non-contrast computed tomography datasets with convolutional neural networks. IEEE Access 8:94871–94879
35. Yu W, Liu T, Valdez R, Gwinn M, Khoury MJ (2010) Application of support vector machine modeling for prediction of common diseases: the case of diabetes and pre-diabetes. BMC Med Inform Decis Mak 10(1):1–7

Hinweis des Verlags Der Verlag bleibt in Hinblick auf geografische Zuordnungen und Gebietsbezeichnungen in veröffentlichten Karten und Institutsadressen neutral.