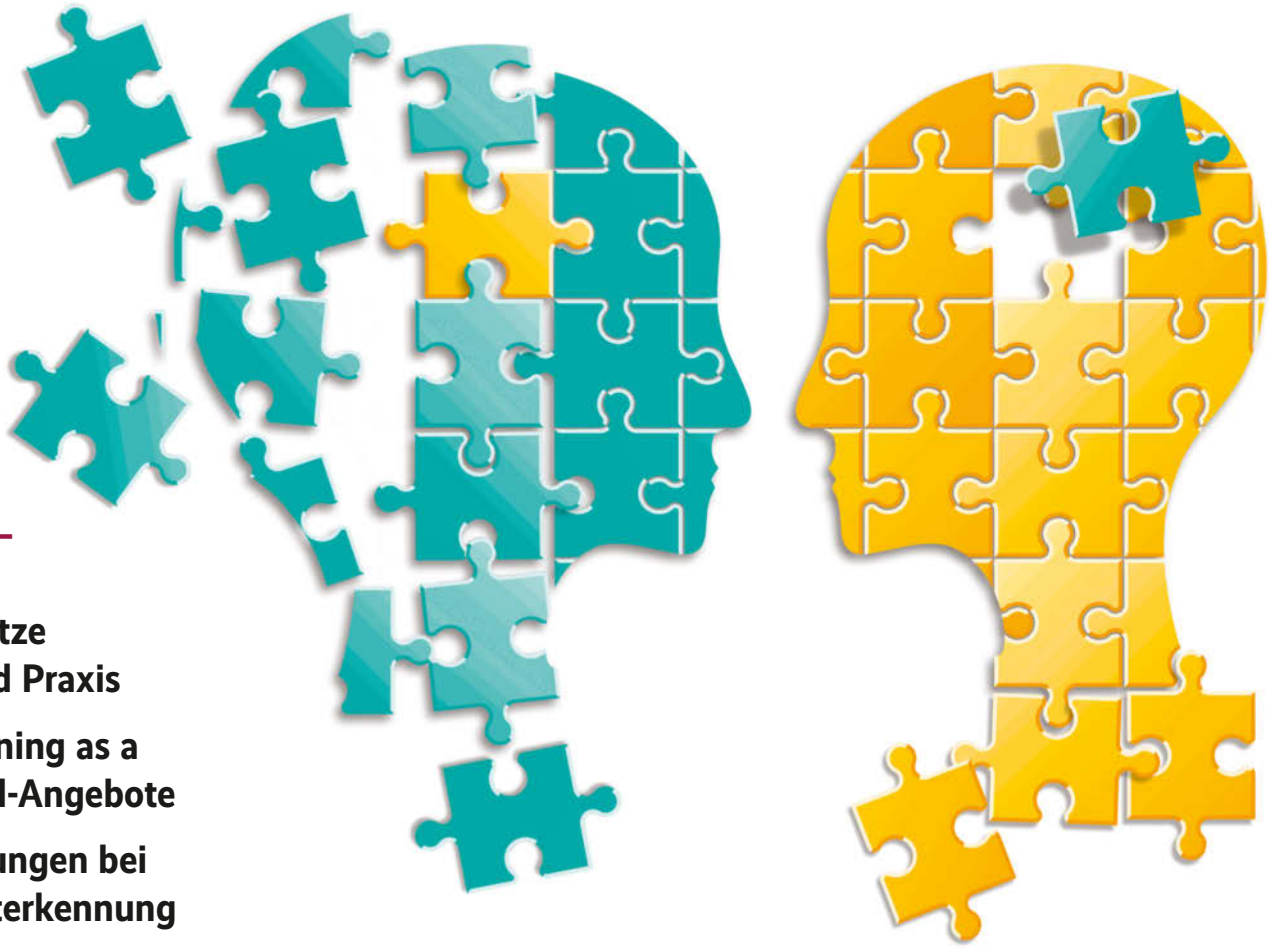
iX DEVELOPER

Winter 2018

# Machine Learning

## Verstehen, verwenden, verifizieren



**Data Science – Status quo**

Neuronale Netze in Theorie und Praxis

Machine Learning as a Service: Cloud-Angebote

Herausforderungen bei Bild- und Texterkennung

Blick in die Blackbox

**TensorFlow, Keras & Co.**

ML-Frameworks und -Bibliotheken

Programmiersprachen: Python, Scala, C++

**Hardware pusht ML**

GPUs und CPUs für Machine Learning

Neuromorphe Chips für neuronale Netze

**Mensch trifft KI**

Rechtsfragen bei Künstlicher Intelligenz

Verantwortungsvoller Einsatz und Ethik

Minds
Mastering
Machines

DIE KONFERENZ FÜR
MACHINE LEARNING UND
KÜNSTLICHE INTELLIGENZ

**JETZT PROPOSAL EINREICHEN!**
Bis zum 6. Januar 2019

## THEMEN

- Machine Learning und Deep Learning
- ML-Bibliotheken und -Frameworks
- Data Mining
- Cognitive Computing
- Computer Vision, Natural Language Processing, Bild- und Spracherkennung
- Predictive Analytics
- Wissensbeschaffung, -darstellung und -vernetzung
- Anbindung an Big-Data-Systeme
- Cloud-Services für Machine Learning
- Hardware: GPUs, CPUs und spezielle Chips
- Ethik und Recht

**14.-16. MAI 2019**

CONGRESS CENTER ROSENGARTEN,
MANNHEIM

WWW.M3-KONFERENZ.DE

# Kein Wunder

Dass Künstliche Intelligenz allgegenwärtig ist, ist kein Wunder – schließlich haben sich die Meldungen in den letzten Jahren überschlagen: Maschinen haben Menschen erst beim Schach, dann in Spielshows und schließlich sogar im komplexen Go geschlagen. Daneben schreitet die Entwicklung selbstfahrender Autos voran, und Roboter bringen erstaunliche Leistungen. Weitaus spannender als die Schlagzeilen sind jedoch die zahlreichen Systeme, die bereits im Produktiveinsatz sind.

Die Methoden des Machine Learning helfen unter anderem bei der Betrugserkennung, beim Vorhersagen von Maschinenausfällen und beim Finden der sprichwörtlichen Nadel im Heuhaufen der Daten. In der Medizin unterstützt die Bildanalyse Ärzte bei der Frühdiagnose von Hautkrebs. Und während serienmäßig selbstfahrende Autos noch Zukunftsmusik sind, nutzen Fahrerassistenzsysteme bereits heute ML-Ansätze unter anderem zum Erkennen von Hindernissen.

Die wachsende Zahl der Anwendungen von Machine Learning ist ebenfalls kein Wunder, sondern vor allem zwei Entwicklungen zu verdanken: der geeigneten Hardware und dem – nicht uneigennützigen – Engagement großer Unternehmen. In den 1990er-Jahren geisterte der Begriff neuronales Netz als Wundermittel zum Lösen der Probleme umher, bei denen von Menschen programmierte Algorithmen an ihre Grenzen stoßen. Es fehlte jedoch sowohl an ausreichend leistungsfähigen Rechnern als auch an passenden Werkzeugen, und Daten zum Training der Systeme waren im Vergleich zu heute Mangelware.

Vor allem Hardwareentwicklungen für massiv parallele Verarbeitung in GPUs und speziellen CPUs sowie Big Data haben dazu geführt, dass die Vision der lernenden Maschinen in der Realität angekommen ist. Dank zahlreicher Open-Source-Frameworks ist es für Entwickler und Data Scientists zudem so einfach wie nie, Machine Learning in eigene Projekte zu integrieren.

Die Systeme vollbringen aber keine Wunder: Realität ist derzeit lediglich die schwache KI, die in der Lage ist, spezielle Aufgaben zu lösen. Ein System zur Bildanalyse müsste zur Interaktion mit Menschen ein zusätzliches Chatbot-System zuschalten. Starke KI, die mit menschlicher Intelligenz mithalten kann und allgemeine Probleme kognitiv angeht, ist nach wie vor Zukunftsmusik.

Machine Learning ist nicht das Wundermittel, das Aufgaben selbsttätig erledigt: Trotz der Möglichkeiten sind die Herausforderungen an die Entwickler und Forscher enorm. Wie erstellt man ein Machine-Learning-Modell und erkennt, ob es für die eigenen Zwecke passt? Was lässt sich überhaupt mit künstlicher Intelligenz sinnvoll umsetzen? Und wie prüft man das Ergebnis, das nicht mehr aus Algorithmen besteht, die sich debuggen lassen?

Auf all diese Fragen möchten wir Ihnen mit diesem Sonderheft Antworten geben. Sie lernen die Grundlagen von ML-Methoden und neuronalen Netzen. Sie erhalten einen Überblick über die wichtigsten Open-Source-Frameworks. Umfangreiche Praxisartikel helfen beim Erkennen des Potenzials von ML für eigene Projekte. Nicht zuletzt behandelt das Heft die wichtigen juristischen und ethischen Fragen, die das Thema mit sich bringt. *iX* und *heise Developer* wünschen Ihnen viel Spaß beim Lesen!

RAINALD MENGE-SONNENTAG

# Überblick

Nach der Entscheidung für den Einsatz von Machine Learning fängt die Suche nach den passenden Methoden an. Spezielle Hardware beschleunigt schließlich die Arbeit und ermöglicht viele Anwendungen überhaupt erst.

ab Seite 7

## Überblick



## Frameworks



## Praxis



## Infrastruktur



## Ethik und Recht



## Sonstiges

# Praxis

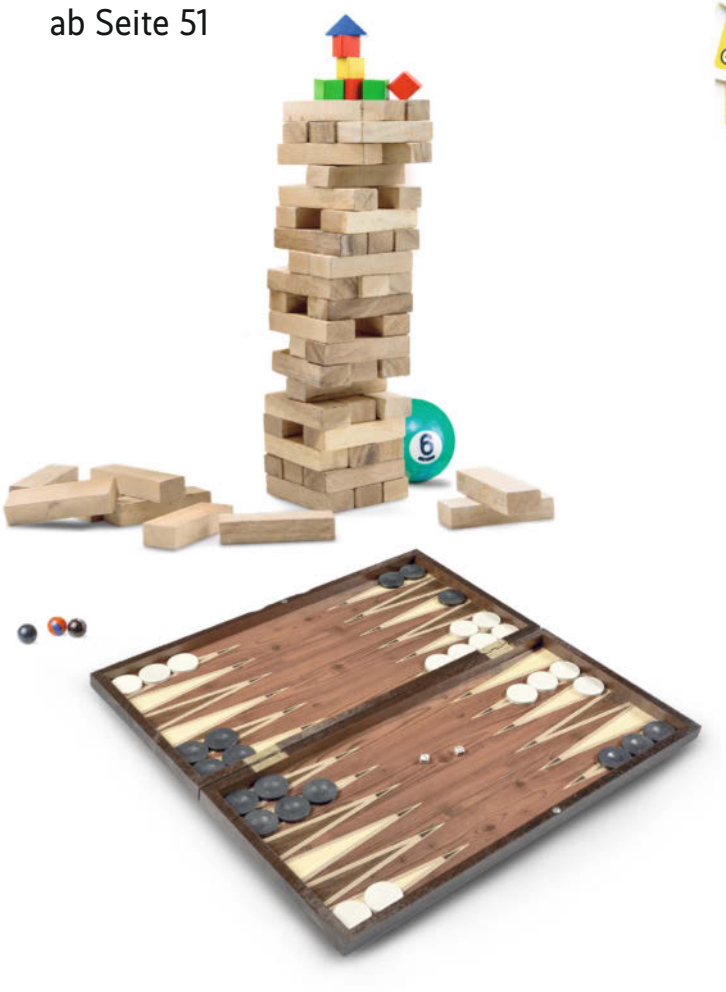
Inzwischen ist KI gereift genug für Erfahrungsberichte aus allgemeinen Bereichen wie Bild- und Textanalyse und dem Verarbeiten natürlicher Sprache, aber auch aus speziellen Anwendungsfeldern wie dem Erkennen relevanter Inhalte.

ab Seite 77


# Frameworks

Die Werkzeugkiste für Machine-Learning-Anwendungen besteht zum Großteil aus gut gepflegten, kostenlosen Open-Source-Tools. Neben dem Platzhirsch TensorFlow rückt vor allem Keras immer mehr in den Fokus. Wer es hardwarenah mag, kann auf cuDNN setzen.

ab Seite 51



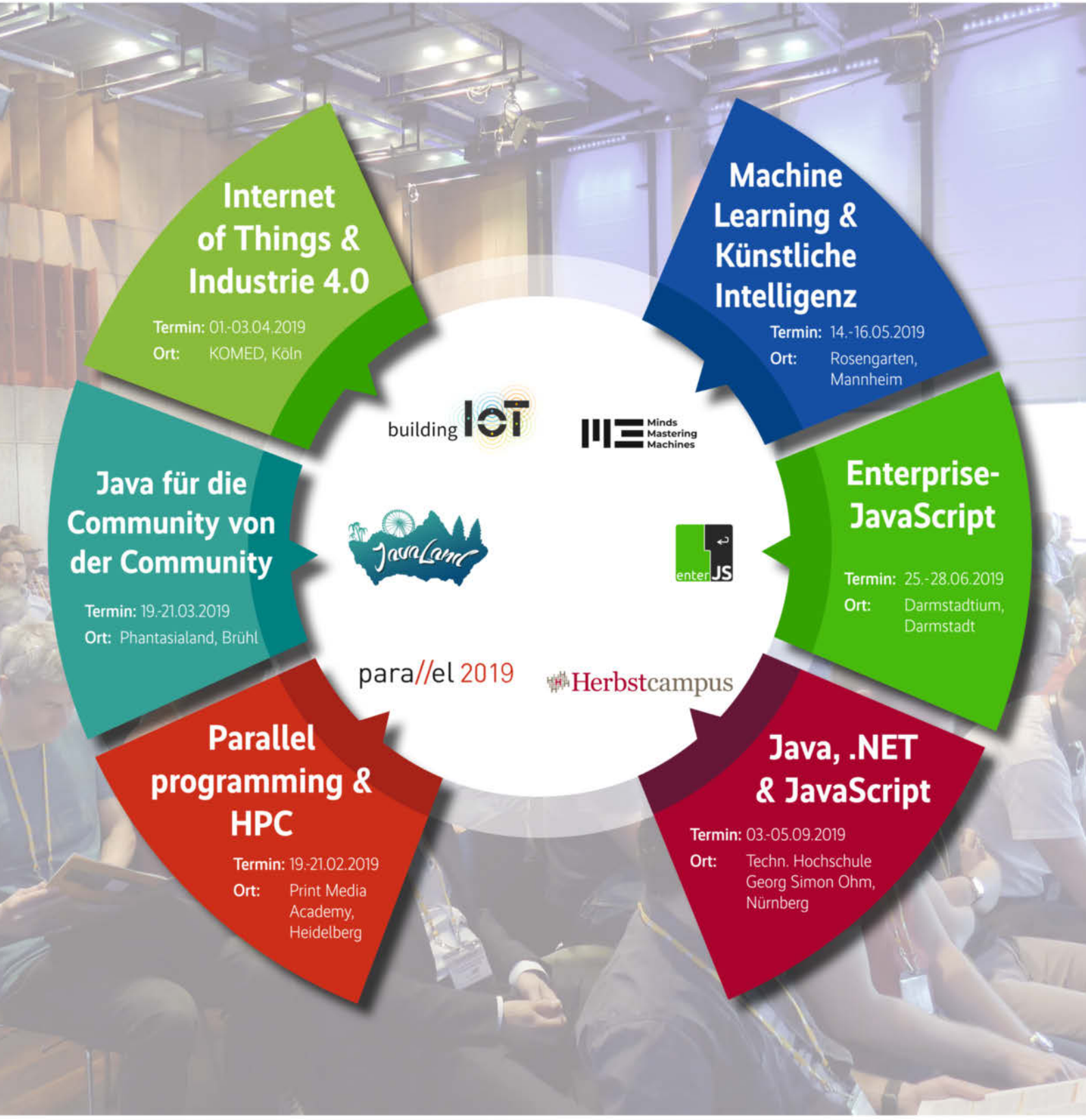

# Ethik und Recht

Der Einsatz von Machine Learning wirft auch juristische und ethische Fragen: Was ist machbar und was nur Hype? Wer haftet für Schäden? Wie können Forscher verantwortungsvoll mit KI umgehen? Und wie nehmen Menschen Maschinen mit künstlicher Intelligenz wahr?

ab Seite 137

Alle Links: www.ix.de/ix1715004 Artikel mit Verweisen ins Web enthalten am Ende einen Hinweis darauf, dass diese Webadressen auf dem Server der *iX* abrufbar sind. Dazu gibt man den *iX*-Link in der URL-Zeile des Browsers ein. Dann kann man auch die längsten Links bequem mit einem Klick ansteuern. Alternativ steht oben rechts auf der *iX*-Homepage ein Eingabefeld zur Verfügung.

# DEVELOPER-KONFERENZEN + -WORKSHOPS 2019



## Internet of Things & Industrie 4.0

**Termin:** 01.-03.04.2019
**Ort:** KOMED, Köln

## Machine Learning & Künstliche Intelligenz

**Termin:** 14.-16.05.2019
**Ort:** Rosengarten, Mannheim

## Java für die Community von der Community

**Termin:** 19.-21.03.2019
**Ort:** Phantasialand, Brühl

## Enterprise-JavaScript

**Termin:** 25.-28.06.2019
**Ort:** Darmstadtium, Darmstadt

## Parallel programming & HPC

**Termin:** 19.-21.02.2019
**Ort:** Print Media Academy, Heidelberg

## Java, .NET & JavaScript

**Termin:** 03.-05.09.2019
**Ort:** Techn. Hochschule Georg Simon Ohm, Nürnberg

building IoT

Minds Mastering Machines

JavaLand

enterJS

para//el 2019

Herbstcampus

Veranstalter:

Weitere Informationen unter:

**www.heise.de/developer/**

## Überblick

Machine Learning hat viele Facetten, und nach der Entscheidung für den Einsatz fängt die Suche nach den passenden Methoden an, allen voran Supervised/Unsupervised/Reinforcement Learning und diverse Arten neuronaler Netze. Oft sind die ML-Ergebnisse nicht mehr nachvollziehbar, aber ein Blick in die Blackbox hilft bei der Erklärung. Spezielle Hardware beschleunigt die Arbeit und ermöglicht viele Anwendungen überhaupt erst.

Grundlagen der Künstlichen Intelligenz

# Maßanfertigung

**Oliver Zeigermann**

Der Oberbegriff Machine Learning deckt zahlreiche Methoden ab, mit denen Software lernen kann. Ein allgemeingültiges Rezept existiert nicht: Bildanalyse erfordert ein anderes Vorgehen als die Auswertung tabellarischer Daten. Ein Überblick über die Ansätze hilft dabei, die passende Methode für eigene Projekte zu finden.

Die Grundidee von Machine Learning ist, einem System durch bestehende Beispieldaten eine Fähigkeit beizubringen. Die besten Voraussetzungen für den erfolgreichen Einsatz sind ein nicht ganz klar definiertes Problem auf der einen und ein großer Pool mit Beispieldaten auf der anderen Seite. Beim heutigen Stand der Technik sollte man mit Fehlern oder einer gewissen Unsicherheit im Ergebnis leben können. Am einfachsten lässt sich das Vorgehen mit einer konkreten Aufgabenstellung erklären.

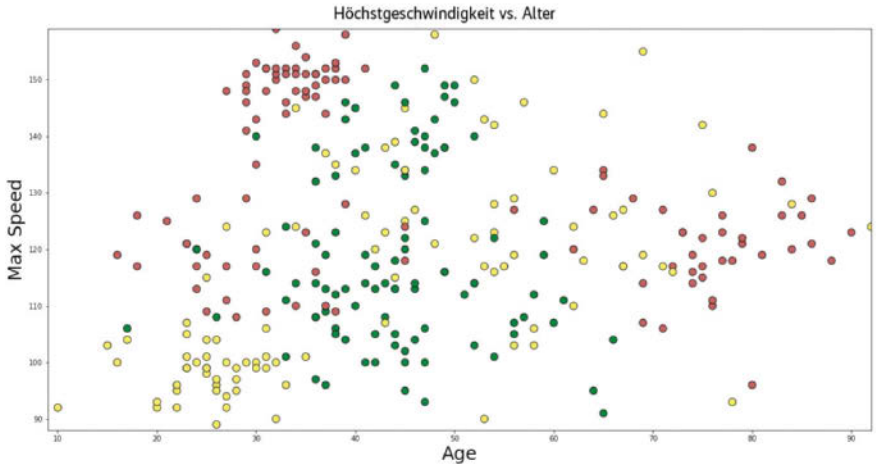
## Beispielanwendung Versicherungswesen

Als Beispiel dient ein frisch gegründetes, fiktives Versicherungsunternehmen für Kraftfahrzeuge. Anders als herkömmliche Versicherer bestimmt es die Beitragsklasse anhand tatsächlich auftretender Schadensfälle. Abbildung 1 zeigt einen Auszug aus der Datenbank bestehender Kunden. Für den US-amerikanischen Markt ist darin das Alter der Versicherten gegen die Höchstgeschwindigkeit ihres jeweiligen Fahrzeugs aufgetragen. Zusätzlich ist in der Farbe der Punkte die Schadenshäufigkeit des Kunden dargestellt. Rot steht für viele Schadensfälle, Grün für wenige oder keine und Gelb für den mittleren Bereich. Darüber hinaus existieren weitere Informationen über die Kunden, für den Anfang sollen aber zunächst diese beiden Variablen dienen.

Ein Übersichtsplot zeigt alle zur Verfügung stehenden Daten an (s. Abb. 2). Er hat die drei Dimensionen Alter, Höchstgeschwindigkeit des Fahrzeugs und jährliche Fahrleistung in Meilen. Die Diagonale zeigt für jede Schadensklasse eine Verteilung der Kundendaten über ihren Wertebereich. In den anderen Teilen ist wie in Abbildung 1 jeweils eine Dimension gegen eine andere aufgetragen. Die Linien beschreiben zudem eine lineare Korrelation der Dimensionen pro Schadensklasse. Ein solcher Plot ist nicht für eine PowerPoint-Präsentation geeignet (Explanation), sondern dient der Erkundung (Exploration) von Daten. Damit lässt er sich sowohl zur Analyse als auch zur Vorbereitung von Machine Learning nutzen.

## Herkömmlicher Ansatz

In einem klassischen Softwareentwicklungsumfeld würde das Team an die Aufgabe vermutlich analytisch herangehen. Ana-




**Die Beispieldaten für die Versicherung, jeder Punkt ist ein Kunde (Abb. 1)**

lysten finden einige Regeln heraus, die sie den Entwicklern mitteilen. Letztere setzen die Regeln anschließend in Software um und nutzen sie für die Klassifikation von Kunden. Damit ließe sich beispielsweise für neue Kunden ein Beitragssatz bestimmen.

Als Annahme sollen folgende Regeln gelten (s. Listing 1):

1. Jüngere Kunden haben eher ein höheres Risiko, mit schnellen Autos ein sehr hohes Risiko,
2. ältere Kunden haben tendenziell ein hohes Risiko,
3. je mehr ein Kunde fährt, desto höher ist das Unfallrisiko und
4. andere Kunden haben ein geringes Risiko.

Zu allen Regeln erzeugen die Entwickler Python-Code. Er weist als weitere Variable die jährliche Fahrleistung in Meilen auf. Da das Beispiel sich auf den US-amerikanischen Markt bezieht, ist auch die Geschwindigkeit in Meilen pro Stunde angegeben.

Aus dem simplen Aufbau der Regeln ergeben sich Fragen, die einige Schwächen offenbaren: Sind weitere Regeln erforderlich? Sind es die richtigen? Sind die Maße nicht sehr willkürlich?

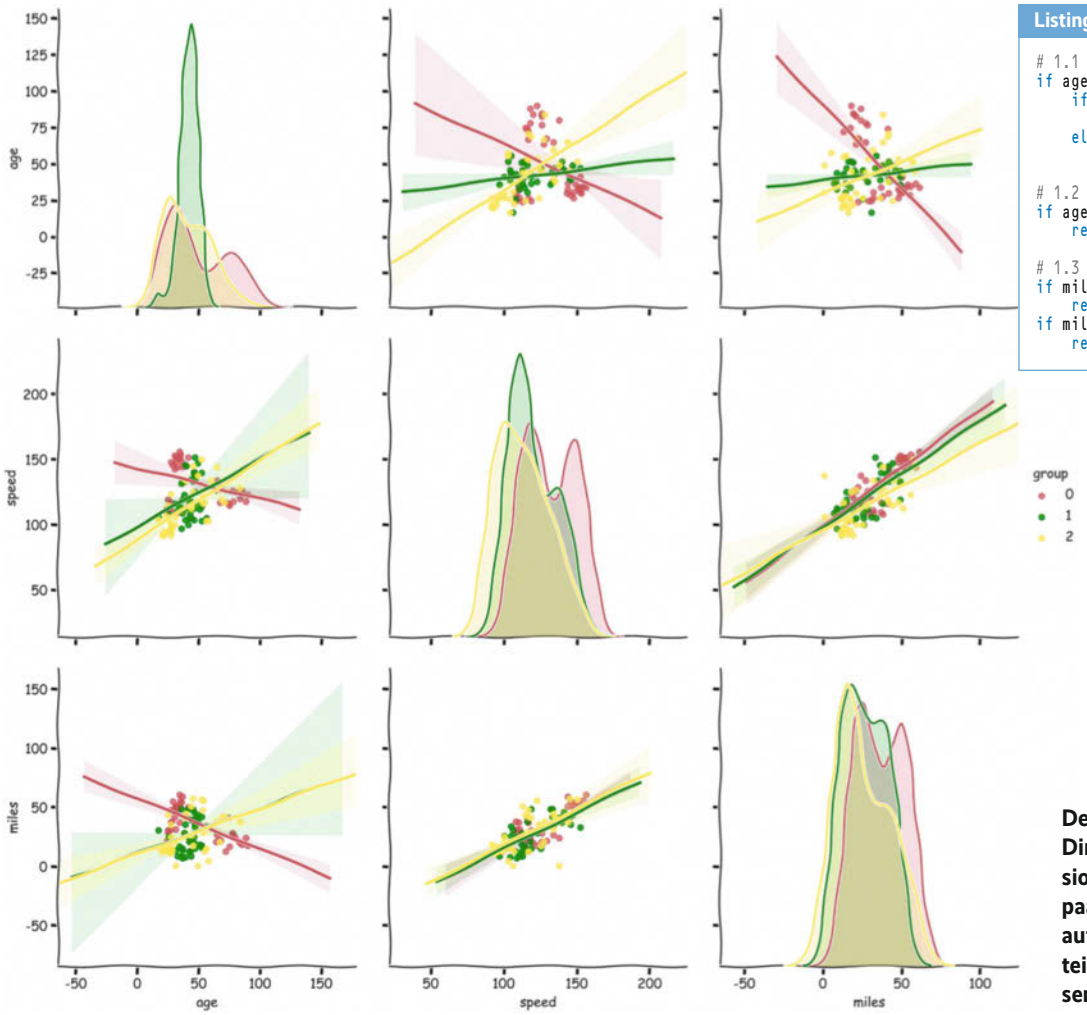
Das Ergebnis, der Regeln ist in Abbildung 3 zu sehen. Zusammen mit den bekannten Datenpunkten sind im Hintergrund die aus den Regeln hergeleiteten Vorhersagen Vorhersagen er-

**Listing 1: Bewertungs-Algorithmen**

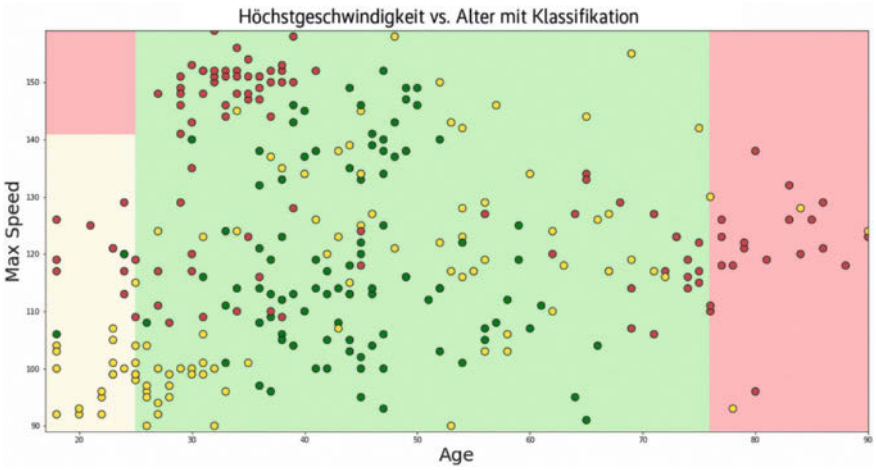
```
# 1.1 Junge Kunden, schnelle Autos
if age > 25:
    if speed > 140:
        return red
    else:
        return yellow

# 1.2 Ältere Kunden
if age > 75:
    return red

# 1.3 Gefahrene Meilen
if miles_per_year > 30000:
    return red
if miles_per_year > 20000:
    return yellow
```




**Der Übersichtsplot mit drei Dimensionen; jede Dimension ist gegen jede andere paarweise aufgetragen und auf der Diagonalen die Verteilung der einzelnen Klassen (Abb. 2).**




**Manuelle Klassifikation mit den Regeln; die Klassifizierung ist durch die Hintergrundfarbe gegeben (Abb. 3).**

kennbar. Der grüne Hintergrund in der Mitte bedeutet beispielsweise, dass alle Punkte innerhalb dieser Fläche eine Klassifikation als guter Kunde mit wenig Schadensfällen bekommen.

Auf den ersten Blick ist ersichtlich, dass viele Punkte eine falsche Zuordnung bekämen. Der komplette Bereich von Fahrern um die 30 Jahre, die schnelle Autos fahren, ist falsch klassifiziert. In Zahlen liegt die Genauigkeit bei etwa 44 Prozent. Das bedeutet, dass weniger als die Hälfte der bekannten Datenpunkte richtig vorausgesagt würden. Man könnte die Regeln nun verfeinern und hoffen, eine höhere Genauigkeit zu erreichen. In jedem Fall ist das Vorgehen aufwendig und erfordert ein gutes Domänenwissen. Darüber hinaus bleibt die Frage offen, wie gut die neuen Algorithmen mit unbekannten Datensätzen funktionieren.

## Machine Learning statt Algorithmen

Machine Learning ist ein komplett anderer Ansatz, der weder Domänenwissen noch das Aufstellen statischer Regeln erfordert. Das Erstellen der Regeln erfüllt die am Artikelanfang genannten Voraussetzungen für den Einsatz von ML: Eine hundertprozentige Genauigkeit in der Vorhersage ist weder notwendig noch erreichbar, und die umzusetzenden Regeln wirken willkürlich. Tatsächlich ist es nicht dramatisch, wenn die Versicherung einen Kunden bei der Aufnahme falsch einschätzt, da sie im schlimmsten Fall eine Schadensklasse falsch bestimmt.

Üblicherweise gilt folgende Unterscheidung für ML-Ansätze:

- Supervised Machine Learning,
- Unsupervised Machine Learning und
- Reinforcement Learning.

Diese Aufteilung passt zwar nicht immer genau, und häufig lässt sich ein Ansatz keiner genauen Kategorie zuweisen, aber die Grundlagen lassen sich mit den drei Kategorien gut beschreiben.

Für das konkrete Beispiel gilt es zunächst, die drei Ansätze zu betrachten, um zu entscheiden, welcher am besten für die Anwendung passt.

## Reinforcement Learning

Beim Reinforcement Learning oder verstärkenden Lernen gibt man lediglich ein Maß dafür an, wie gut eine Maschine ein Problem löst und braucht dazu eine Beschreibung der Aufgabe und die Angabe, welche Aktionen möglich sind. Den Rest findet die Maschine eigenständig heraus. Ein prominentes Beispiel dafür ist das AlphaGo-Zero-System, das durch Spielen von Go-Partien gegen sich selbst herausgefunden hat, wie man Go spielt. Als Eingabe bekam das System lediglich die Regeln für das Brettspiel Go: also wann welcher Zug erlaubt ist und was jeweils passiert.

Das funktionierte dermaßen gut, dass ein solches System die besten Go-Spieler der Welt schlagen kann. Dazu muss es keine Bewertung einer Stellung wie beim Schach erkennen, sondern lediglich, wann eine Partie gewonnen ist. Die Bewertung erlernt es eigenständig.

Dieser Ansatz klingt vielversprechend und kommt auch ohne Trainingsdaten aus. Voraussetzung ist aber ein verlässlicher Simulator, mit dem sich vorgenommene Aktionen bewerten lassen.

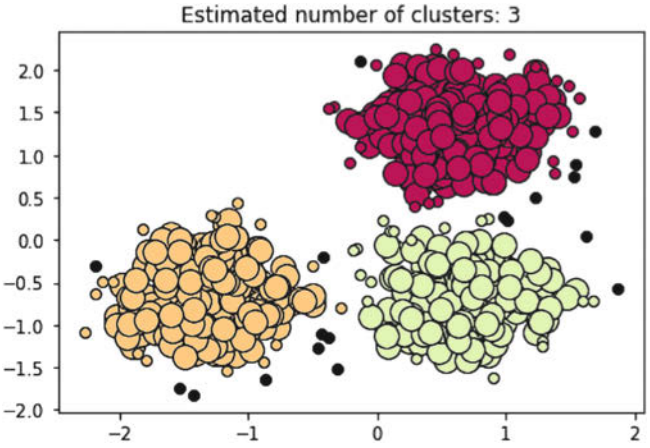
Reinforcement Learning lässt sich daher für viele allgemeine Problemstellungen und auch für das Erstellen der Versicherungsregeln nicht wirklich gut verwenden.

## Unsupervised Machine Learning

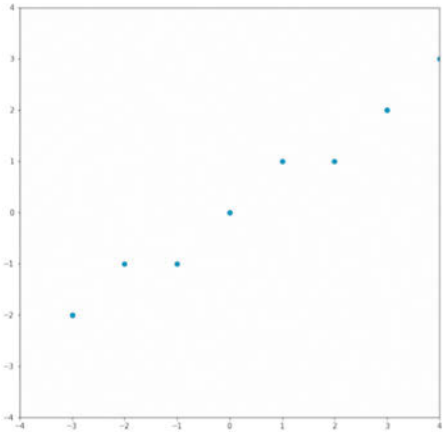
Typischerweise spricht man bei Clustering-Verfahren und der Reduktion von Dimensionen von Unsupervised Machine Learning oder unüberwachtem Lernen.

## Clustering

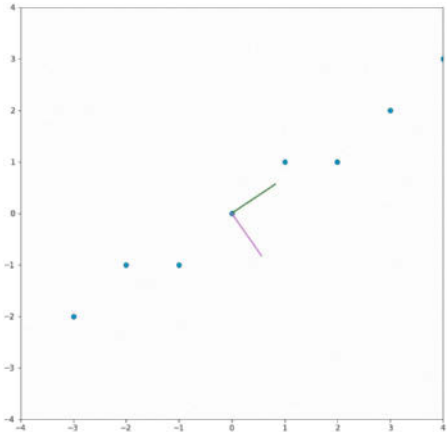
Das Clustering erfordert keine bekannten Klassifikationen. Anders ausgedrückt könnte der in Abbildung 3 gezeigte Plot ein-



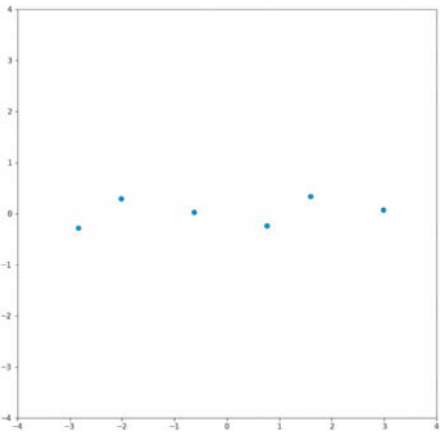

**Das Ergebnis des DBSCAN-Clusterings, Punkte gleicher Farbe gehören zu demselben Cluster, schwarze Punkte sind Ausreißer (Abb. 4).**



**Die ursprünglichen Daten vor der Transformation (Abb. 5)**



**Die neuen Achsen basierend auf der PCA (Abb. 6)**



**Die Daten nach der Reduktion mit PCA (Abb. 7)**

farbig sein. Es gibt somit zwar Kundendaten, aber keine Einschätzung der Schadensklasse. Das System versucht nun herauszufinden, welche Datenpunkte ähnlich und welche Ausreißer sind. Es gibt unterschiedliche Verfahren mit verschiedenen Eigenschaften; ein bewährtes ist DBSCAN [a].

Das Ergebnis eines solchen Clusterings zeigt Abbildung 4. Die Punkte sind vorgegeben und die Farben pro Cluster sowie deren Größen durch das DBSCAN-Clustering entstanden. Schwarze Punkte sind Ausreißer, die großen Punkte bilden den Kern eines Clusters, und die kleinen Punkte sind in der Peripherie eines Clusters, gehören aber noch dazu.

Mit der Python-Bibliothek scikit-learn [b] ist die Anwendung eines DBSCAN-Clustering trivial. In folgendem Python-Code ist *X* das mehrdimensionale Array der Eingabedaten:

```
from sklearn.cluster import DBSCAN

clf = DBSCAN() clf.fit(X)
```

## Dimensionality Reduction

Die Reduktion von vielen Parametern auf wenige, dafür jedoch aussagekräftigere nennt man Dimensionsreduzierung (Dimensionality Reduction). Damit lassen sich unter anderem Daten auf zwei Dimensionen reduzieren, um sie für Menschen passend zu visualisieren. Eine weitere Anwendung ist das Reduzieren vieler Dimensionen auf eine geringere Zahl, um beim späteren Training bessere Ergebnisse zu erzielen. Die Reduktion führt typischerweise zu schnellerem Training und einer weniger präzisen Generalisierung für unbekannte Daten.

Das klassische Verfahren dafür ist die Hauptkomponentenanalyse (Principal Component Analysis, PCA) [c], das ein Koordinatensystem in andere Achsen transformiert. Ein Beispiel für Datenpunkte in zwei Dimensionen zeigt Abbildung 5.

Ein genauer Blick auf das Diagramm mag die Frage provozieren, ob es sich tatsächlich um zwei Dimensionen handelt. Die Punkte bilden eine fast perfekte Linie, und beim Betrachten dieser Linie als neuer Achse würde die zweite Achse, die darauf rechtwinklig steht, fast gar keine Information mehr tragen.

Die neuen Achsen sind in Abbildung 6 zu sehen. Die grüne ist die Achse entlang der Linie und die in Magenta ist dazu orthogonal.

Folgender Code transformiert die Daten wieder mit scikit-learn auf die neuen Achsen:

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
pca.fit(X)
```

Das Ergebnis (s. Abb. 7) zeigt deutlich, dass die y-Achse nicht zwingend erforderlich ist, da sich alle Werte um die Nullachse

AI | Spektrum

DAS FACHPORTAL FÜR KÜNSTLICHE INTELLIGENZ

eBooks /// Fachartikel /// Infografiken /// Poster

Jetzt registrieren und **kostenfreien** Zugriff auf das gesamte AI-Wissen erhalten

ai-spektrum.de

**Listing 2: Fully Connected Neural Network mit Keras**

```
from tensorflow.keras.layers import Dense, Dropout,
                                    BatchNormalization, Activation

model = keras.Sequential()

model.add(Dense(100, input_dim=2))
model.add(Dense(100)
```

herum befinden. Erstere kann somit ohne großen Verlust wegfallen.

## Supervised Machine Learning

Überwachtes Lernen (Supervised Machine Learning) benötigt passende Paare von Ein- und Ausgabedaten. Dabei wird ein Modell mit diesen bekannten Daten in der Hoffnung gefüttert, später die Ausgabe für eine bisher nicht bekannte Eingabe zu erzeugen.

Es findet also eine Generalisierung von bekannten auf bisher unbekannte Datenpaare statt. Voraussetzung ist daher zunächst das Vorhandensein von Daten. Der Erfolg hängt hauptsächlich von Menge und Qualität solcher Daten ab.

Der konkrete Anwendungsfall schreit geradezu nach überwachtem Lernen, da neben den Kundendaten zusätzlich Klassifikationen nach Schadensklasse existieren. Nach der passenden Art des Machine Learning steht die Wahl der Strategie an.

Das für den bisherigen Code verwendete scikit-learn eignet sich als Standard-Bibliothek für klassisches Maschine Learning in Python ebenso für überwachtes Lernen wie für das bisher gezeigte unüberwachte. Zu den bekannten, von scikit-learn gebotenen Verfahren des Supervised ML gehören Decision Trees, Random Forest und Support Vector Machines.

Decision Trees funktionieren sehr ähnlich zu von Hand kodierten Regeln in Form verschachtelter *if-then*-Blöcke, aber das System lernt die Regeln. Random Forests sind eine Kombination aus vielen solchen Decision Trees, die für eine bessere Vorhersage von bisher unbekannten Daten sorgen. Support Vector Machines versuchen anhand von einigen, wenigen Datenpunkten die als Stützpunkte dienen, eine möglichst gute Grenze zwischen den einzelnen Klassen zu ziehen. Dabei lässt sich zwischen einer möglichst glatten Linie und möglichst wenig falsch klassifizierten Punkten abwägen.

Für die fiktive Versicherung taugt in der Tat ein solches klassisches Verfahren.

Darüber hinaus gibt es Alternativen in Form von künstlichen neuronalen Netzen und Deep Learning, die in der jüngeren Vergangenheit Fahrt aufgenommen haben. Dafür bietet sich die verbreitete Bibliothek TensorFlow [d] an. Ein neuronales Netz setzt sich aus einer Menge künstlicher Neuronen zusammen, die sich je nach Art der Daten auf unterschiedliche Weise verschalten lassen.

## Recurrent Neural Networks

Das Besondere an Sequenzen wie Texten oder Börsenkursen ist, dass hintereinander folgende Punkte nicht unabhängig voneinander sind. Wörter bekommen erst im Zusammenhang mit anderen ihren eigentlichen Sinn. Auch zeitliche Verläufe haben oft eine Regelmäßigkeit.

Zum Lernen einer solchen Abfolge muss das System Informationen über die Vergangenheit mitlernen. Die Aufgabe können Recurrent Neural Networks (RNN) erledigen, indem sie ihre Ausgaben bei jedem Zeitschritt wieder als Eingaben nutzen. Die bekannten Ausprägungen sind Long Short-Term Memorys (LSTM) [e] und Gated Recurrent Units (GRU).

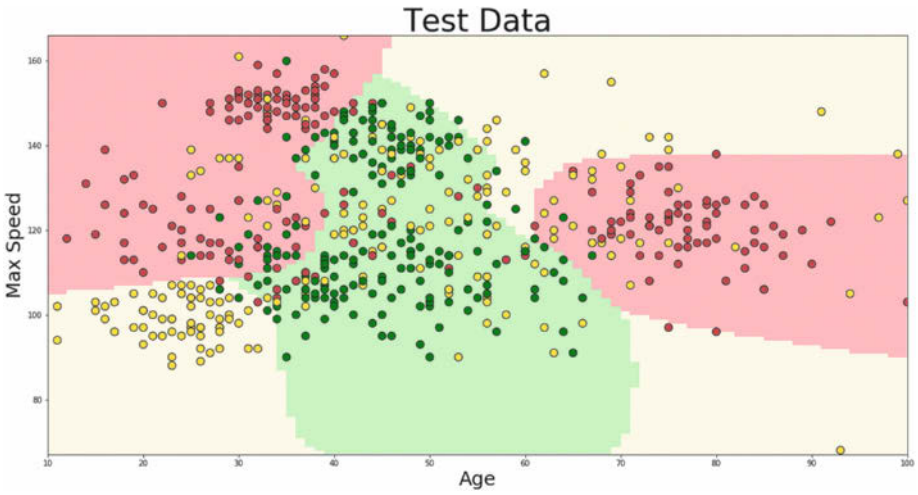
Bilddaten haben die Eigenschaft, dass Bildpunkte, die nebeneinander liegen, in Zusammenhang stehen. Das nutzen Convolutional Neural Networks (CNN) [f] und erlernen Filteroperationen, die lokale Nachbarschaften in Betracht ziehen. CNNs sind jedoch nicht nur auf Bildanalyse beschränkt, sondern kommen unter anderem auch für bestimmte Anwendungsfälle in der Textanalyse zum Einsatz.

## Fully Connected Neural Networks

Die fiktive Versicherung braucht für ihre Bewertung weder Bilddaten noch Sequenzen. Zur Auswertung stehen stattdessen tabellarische Daten bereit. Für deren Verarbeitung haben sich künstliche neuronale Netze mit Fully-connected Layers bewährt. Die künstlichen Neuronen sind dabei in mehreren Schichten organisiert, wobei jedes Neuron einer Schicht mit jedem der nächsten Schicht verbunden ist. Bei zahlreichen Layers spricht man von Deep Learning.

In TensorFlow hilft bei der Anwendung die Keras-API [g]. Listing 2 definiert ein Netz mit zwei sequenziell hintereinanderstehenden Schichten mit jeweils 100 Neuronen. Keras (siehe Artikel "Komfortabler Modellbau" auf S. 64) erlaubt eine Beschreibung auf einer hohen Abstraktionsebene. Es hat als eigenständiges Projekt begonnen, das neben TensorFlow weitere Frameworks für neuronale Netze unterstützt hat. Inzwischen arbeitet der Hauptentwickler für Google, und Keras ist als integraler Bestandteil des vom Internetriesen ins Leben gerufenen TensorFlow verfügbar.

Die erste Schicht für das Versicherungsbeispiel enthält die beiden Pa-




**Ergebnis der Vorhersage mit Keras – dabei fallen vor allem die differenzierten, aber glatten Grenzen zwischen den Klassen auf (Abb. 8).**

rameter Alter und Geschwindigkeit. Eine Vorhersage mit einem solchen Netz kommt auf über 70 Prozent Genauigkeit. Die Darstellung der Vorhersagen (s. Abb. 8) unterscheidet sich deutlich von der Vorhersage auf Grundlage der von Hand kodierten Regeln. Eine ähnliche Form liefern andere klassische Lernverfahren. Sie scheint somit für die Daten besonders geeignet zu sein.

Damit bleibt jedoch die Frage, wie viel die Genauigkeit im praktischen Einsatz wert ist. Die Einordnung geschieht schließlich, um die Kunden zu klassifizieren, deren Unfallverhalten noch unbekannt ist. Die Antwort steckt unter anderem in der Überschrift „Test Data“ in Abbildung 8.

## Die Kunst der Generalisierung

Wie lässt sich eine Genauigkeit für bisher unbekannte Daten erreichen. Dazu dient ein recht simpler Trick: Data Scientists spalten von den ihnen bekannten Daten einen bestimmten Anteil ab, den sie nicht für das Training des neuronalen Netzes, sondern für Tests nutzen. Im konkreten Fallbeispiel dienen 20 Prozent des Gesamtdatensatzes für den Test. Eine andere prozentuale Aufteilung ist freilich ebenso möglich.

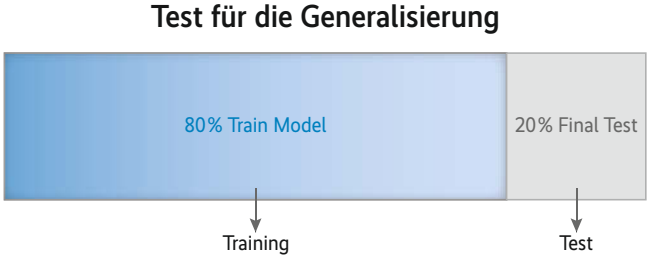
Die Hoffnung ist nun, dass die bekannten Daten grundsätzlich ähnlich zu den unbekannten sind, damit der Test mit den abgezweigten Daten Rückschlüsse auf unbekannte Daten zulässt.

Die Auswertung der fiktiven Versicherungsdaten bietet eine schöne Metrik für die Qualität des Netzes: einen Prozentsatz von richtigen Vorhersagen auf Testdaten. Darüber hinaus liefert sie einen zweiten Wert: den entsprechenden Prozentsatz für die Trainingsdaten.

Wenn die beiden Werte im Idealfall gleich sind, bedeutet das, dass das Generalisieren optimal funktioniert. Das Modell arbeitet genauso gut mit unbekannten Daten wie für bekannte. Wenn das nicht der Fall ist und das Modell besser für bekannte Daten funktioniert, spricht man vom Overfitting. Das bedeutet, dass sich die Data Scientists zu sehr auf die bekannten Daten eingeschossen haben. Je nach Lernstrategie existieren unterschiedliche Techniken, um dem Overfitting entgegenzuwirken. Der grundsätzliche Einsatz von Gegenmaßnahmen heißt Regularisierung.

## Anwendungen

Viele Experten – darunter der Entwickler von Keras – sind der Meinung, dass für die Anwendung von Machine Learning ein




**Ein Teil des Gesamtdatensatzes dient für den Test zur Generalisierung (Abb. 9).**

enormes Potenzial besteht. Fairerweise muss erwähnt werden, dass die Forschung zwar in den letzten Jahren große Fortschritte erzielt hat, aber die Anwendungen noch hinterherhinken. Für die eingangs gestellte Aufgabe der Risikobewertung ist mit einem künstlichen neuronalen Netz ein Ansatz gefunden, der eine gute Genauigkeit verspricht. Hier passen Anwendungsfall und Machine Learning zusammen.

Andere Aufgaben erfordern potenziell andere Ansätze. Zum Abschluss soll ein Überblick einiger real existierender Anwendungen Entscheidungshilfe geben, ob sich der Einsatz von Machine Learning in geplanten Projekten lohnt.

- Empfehlungsdienste: Spotify und Netflix kennen die Anwender besser als sie sich selbst,
- Fraud Detection: Erkennt, wenn eine Kreditkarten-Transaktion oder eine Steuererklärung ungewöhnlich aussieht,
- Predictive Maintenance: KI-Systeme melden, wann man ein Wasserrohr ersetzen soll, bevor es bricht,
- Expertensysteme: Zuordnung von Krankheit und Maßnahmen
- Customer Conversion/Churn: erkennt, ob jemand später Kunde wird oder vorhat zu kündigen,
- Image Recognition: verschafft einen Überblick über eingereichte Dokumente und
- Text Classification: Einordnung, welche Abteilung sich um die E-Mail eines Kunden kümmert.

## Fazit

Machine Learning bietet sich häufig als Alternative zu klassischen Programmieransätzen an. Die Einsatzmöglichkeiten sind vielfältig und hauptsächlich durch passende Daten beschränkt. Die Werkzeuge scikit-learn und TensorFlow sind weit verbreitet und bieten in Kombination Ansätze für jede Art von Machine Learning und Daten an.

Wer sich nach dem Lesen dieses Artikels intensiver mit eigenen Experimenten rund um Machine Learning beschäftigen will, kann passendes Datenmaterial aus der Google-Suchmaschine für Daten [h] beziehen, die für zahlreiche Bereiche Ergebnisse liefert. (rme@ix.de)

### Onlinequellen

| | | |
|---|---|---|
| [a] | DBSCAN | de.wikipedia.org/wiki/DBSCAN |
| [b] | scikit-learn | scikit-learn.org |
| [c] | PCA | de.wikipedia.org/wiki/Hauptkomponentenanalyse |
| [d] | TensorFlow | tensorflow.org |
| [e] | LSTM | de.wikipedia.org/wiki/Long_short-term_memory |
| [f] | CNN | de.wikipedia.org/wiki/Convolutional_Neural_Network |
| [g] | Keras | www.tensorflow.org/guide/keras |
| [h] | Google-Suchmaschine für Daten | toolbox.google.com/datasetsearch |

**Oliver Zeigermann**
hat über Jahrzehnte in vielen unterschiedlichen Sprachen und mit vielen Technologien Software entwickelt. In den letzten Jahren ist er tief in die Analyse großer Datenmengen unter anderem auch mit Techniken des Machine Learning eingestiegen.

*Alle Links: ix.de/ix1815008*

Hardware als Basis intelligenter IT-Management-Strategien

# Mit Big Data und KI zu hochverfügbaren Rechenzentren

**Damit KI-gestützte Systeme selbstständig IT-Umgebungen analysieren, Fehler identifizieren und Probleme lösen können, benötigen sie so viele Informationen wie möglich. Die Kommunikation mit der Hardware spielt dabei eine Schlüsselrolle.**

Unternehmen setzen große Hoffnungen in Big Data, also das Erfassen, Aggregieren, Aufbereiten und Analysieren sehr großer Datenmengen. Das Marktforschungsunternehmen IDC erwartet für 2018 allein in Deutschland einen Umsatz von 6,4 Milliarden Euro mit Hardware, Software und Dienstleistungen für Big-Data-Anwendungen – rund zehn Prozent mehr als im vergangenen Jahr.[1]

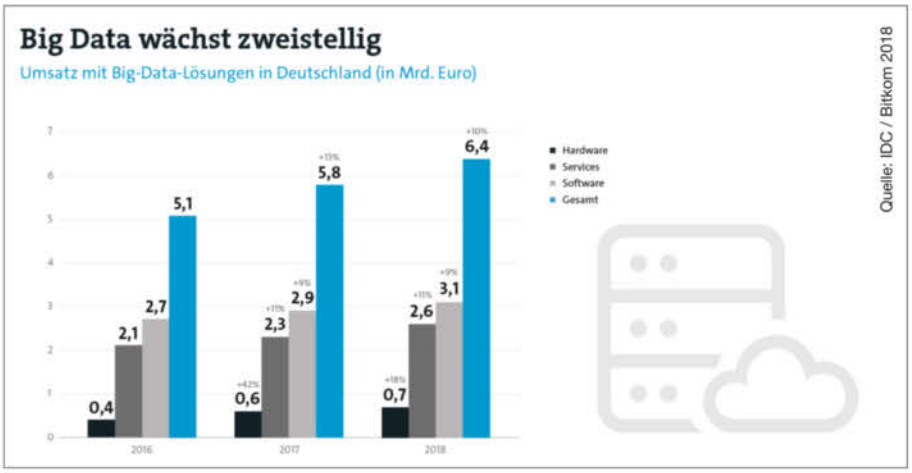
Auf die Begeisterung für Big Data folgt allerdings häufig schnell Ernüchterung. Nach Untersuchungen des Analystenhauses Gartner scheitern rund 85 Prozent aller Big-Data-Projekte.[2] Eine der Hauptschwierigkeiten stellt laut Gartner-Analyst Nick Heudecker die Integration in bestehende Geschäftsprozesse und Anwendungen dar. Andere wesentliche Faktoren sind die Blockadehaltung des Managements und überzogene Erwartungen. Hinzu kommen mangelnde Kenntnisse und Fähigkeiten, die Datenflut zu analysieren und zu interpretieren. Heudecker empfiehlt daher, pragmatisch vorzugehen und mit kleinen überschaubaren Projekten zu starten.

## Voraussetzungen für Big Data

Damit ein Big-Data-Projekt Chancen auf ein Gelingen hat, sollte es einen klar definierten, überschaubaren Umfang, schnelle und leicht nachvollziehbare Erfolge und Unterstützung bei der Datenanalyse bieten. Wie sich diese Anforderungen erfüllen lassen, wird beispielsweise an den HPE Nimble Storage All-Flash Arrays deutlich: Sie erfassen über zahlreiche softwarebasierte Sensoren Daten rund um Zustand und Leistung der Systeme und senden sie an die Cloud-basierte Analyseplattform InfoSight. Die aggregierte Datenmenge ist dabei wirklich „Big Data“: InfoSight sammelt und analysiert in vier Stunden mehr als 100 Milliarden Datenpunkte!

Zu den rund 4.000 Telemetriedaten, die von den Storage-Arrays erfasst und übermittelt werden, gehören beispielsweise Informationen über Durchsatz und Performance, über den Zustand der SSDs oder die aktuelle Datenreduktionsrate. Anders als bei traditionellen Log-Files muss der Kunde die gelieferten Daten nicht selbst auswerten und interpretieren, sondern erhält direkt eine Aussage darüber, welchen Einfluss die aktuellen Werte auf die Performance seiner Systeme haben.

Die Arrays stehen aber natürlich nicht für sich allein, sondern sind in eine umfassende IT-Infrastruktur eingebunden, die sich mit zusätzlichen Plug-ins auf den Nimble-Systemen ebenfalls monitoren lässt. So kann die Software beispielsweise Log-Dateien aus VMware-Umgebungen sammeln und an InfoSight übermitteln. Intelligente Verfahren wie Korrelationsanalysen, Systemmodellierung und Predictive Analytics erlauben es, Ursachen für Performance-Einbrüche zu identifizieren, auch wenn diese nicht vom Storage-System selbst verursacht werden. Dank der intelligenten Analysen und der proaktiven, zum großen Teil automatisierten Fehlerbehebungsverfahren lässt sich eine Verfügbarkeit von mehr als 99,9999 Prozent erreichen – und das nicht nur über die installierte Basis, sondern auch über alle Betriebssystemebenen hinweg.




**Deutsche Firmen investieren verstärkt in Big Data.**

## Beispiele aus der Praxis

Wie sich mithilfe der Hardwaretelemetrie komplexe IT-Probleme lösen lassen, deren Ursachen bisher nur mit großem Aufwand, langen Analysezeiten oder womöglich gar nicht festgestellt werden konnten, zeigen die folgenden Beispiele:

**Performance-Einbrüche durch fehlerhafte Kommunikation:** Ein Kunde musste feststellen, dass bei der Übertragung großer Datenmengen im Bereich von 2 Petabyte (PB) der Datendurchsatz immer wieder von 200 MB/s auf 20 MB/s einbrach. Auf Basis der von den Nimble Arrays gelieferten Daten konnte InfoSight

[1] https://www.bitkom.org/Presse/Presseinformation/Markt-fuer-Big-Data-waechst-in-Deutschland-zweistellig.html
[2] https://www.techrepublic.com/article/85-of-big-data-projects-fail-but-your-developers-can-help-yours-succeed/

einen Fehler im Hypervisor als Ursache identifizieren. Eine falsche Antwort auf ein SCSI-Kommando führte zu einer exzessiven Zunahme von zusätzlichen Schreibbefehlen. Indem das System, das den Fehler verursachte, identifiziert und gesperrt werden konnte, ließ sich der Performance-Einbruch vermeiden. Mehr noch: Mit den Informationen konnten bei weiteren 600 Kunden ähnliche Probleme von vornherein ausgeschlossen werden.

**Fehlerhaft ausgerichtete Schreib-/Lesezugriffe (Unaligned IO):** Um maximale Performance sicherzustellen, sollte der erste logische Block eines Lese- oder Schreibzugriffs am Beginn eines physischen Blocks auf dem Speichermedium liegen. In manchen Fällen kommt es jedoch zu einer fehlerhaften Ausrichtung (Unaligned IO), was die Übertragungsgeschwindigkeit um bis zu 50 Prozent reduzieren kann. Mit herkömmlichen Management-Tools lässt sich der Fehler nicht erkennen. Nimble Storage und InfoSight konnten dagegen bei 200 Kunden potenzielle Alignment -Fehler im Vorfeld feststellen und so einen Performance-Einbruch verhindern.

**„All Paths Down"-Fehler:** Bei großen Lese- oder Schreib-Workloads brach immer wieder die Verbindung zwischen Hypervisor und Speichergerät ab (All Paths Down). Durch eine automatische Korrelationsanalyse kam HPE dem Fehler auf die Spur und informierte den Hersteller des Hypervisor-Systems, der das Problem mithilfe entsprechender Informationen lösen konnte.

**Extreme Latenzen:** Ein Kunde hatte bei geschäftskritischen Applikationen mit extrem hohen Latenzen zu kämpfen. Der Hersteller der betroffenen Netzwerkinfrastruktur versuchte seit sechs Monaten erfolglos, das Problem zu lösen. Mithilfe einer Analyse der kompletten Signalwege durch InfoSight ließ sich der Verursacher schnell identifizieren: Es handelte sich um einen falsch konfigurierten Switch. Nach einer Korrektur der Einstellungen war das Problem behoben.

**Sporadische Latenzeinbrüche:** Immer wieder traten bei einem Kunden Latenzeinbrüche auf, die das betroffene System unbrauchbar machten. Eine manuelle Analyse der beteiligten Komponenten ergab keine erkennbare Ursache. Durch eine Ende-zu-Ende-Korrelationsanalyse der Pfade konnte InfoSight den Verursacher – einen defekten Netzwerkadapter – identifizieren. Der Austausch der Karte behob das Problem sofort.

**Die Sensoren der Nimble Storage Arrays liefen detaillierte Informationen über die Performance der Systeme.**

**Offene Ports:** Gegenüber dem Internet geöffnete Ports werden von Cyber-Kriminellen innerhalb von Minuten durch Brute-Force-Attacken erkannt und für DDoS-Angriffe (Distributed Denial-of-Service) oder den Diebstahl von Daten genutzt. Dank der von den Arrays gelieferten Informationen konnten bei 100 Kunden offene Ports identifiziert und so 400 TB an Daten vor dem Zugriff von außen geschützt werden.

**Ausfall der Klimaanlage:** Selbst äußere Einflüsse, etwa der Ausfall von Hilfssystemen, lassen sich durch die in InfoSight verwendeten intelligenten Korrelationsanalysen erkennen. Im konkreten Fall stellten Nimble Storage und InfoSight den Ausfall der Klimaanlage 36 Minuten früher als jedes andere Überwachungssystem fest.

## Fazit

Unternehmen setzen große Hoffnungen in Big Data, die in der Realität jedoch häufig enttäuscht werden. Die Gründe für das Scheitern von Big-Data-Projekten sind vielfältig – Datenqualität, Datenmenge und Datendichte spielen dabei eine große Rolle. Moderne Systeme wie die Nimble Storage Arrays, die mit einer Vielzahl von Sensoren ausgestattet sind, bilden daher eine wertvolle Basis für eine erfolgreiche Big-Data-Strategie, denn sie liefern zuverlässig, schnell und konsistent eine Vielzahl von Informationen. In Kombination mit fortschrittlichen, auf KI basierenden Analysemethoden lassen sich so Fehler und deren Ursachen in einer Geschwindigkeit und Qualität identifizieren, die bisher nicht vorstellbar waren.

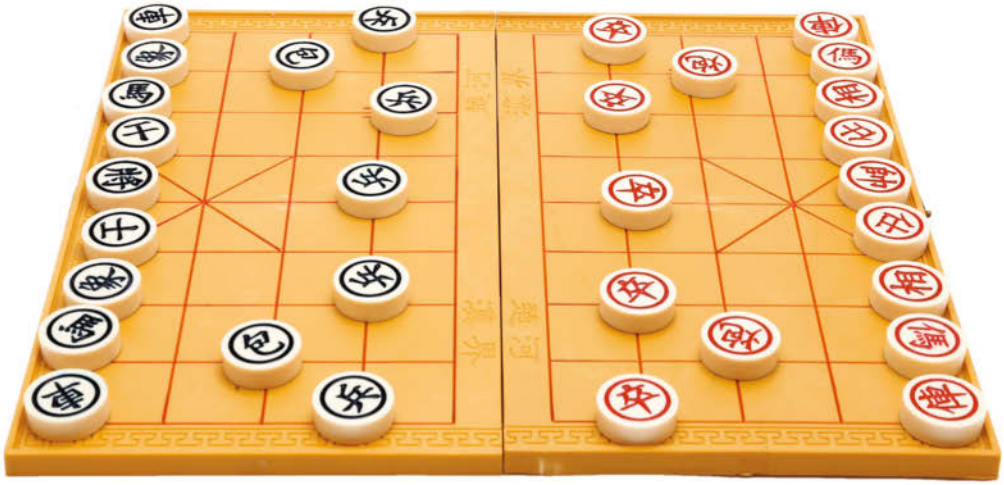
**Dieser Beitrag ist Teil des eBooks „KI: Revolution im Rechenzentrum", das HPE in Kooperation mit Heise Business Services umgesetzt hat. Das e-Book steht Ihnen hier zum Download zur Verfügung:**

Hewlett Packard Enterprise

**https://www.hpe.com/de/de/storage/nimble.html**

Künstliche neuronale Netze in Theorie und Praxis

# Das Gehirn des Rechners



**Marcel Tilly**

Ein Blick hinter die Kulissen offenbart, dass KNNs und Deep Learning weit weniger komplex sind als viele glauben mögen. Das Verständnis für die Arbeitsweise und Methoden hilft dabei, Machine-Learning-Frameworks für eigene Projekte passend einzusetzen.

Künstliche Intelligenz ist in der Realität und in der allgemeinen Wahrnehmung angekommen. Die Erfolge von DeepMind mit AlphaGo, die Berichte zu Bild- und Gesichtserkennung oder smarten Bots sind in aller Munde. Die meisten Ansätze nutzen Machine Learning im Allgemeinen oder Deep Learning im Speziellen. Das mag anfangs nach höchst komplexer Mathematik klingen und etwas Mystisches besitzen, aber eigentlich sind die Grundlagen ziemlich trivial und lassen sich auf ein paar einfache Konzepte herunterbrechen.

Im Grunde geht es bei Deep Learning darum, ein künstliches neuronales Netz (KNN) zu bilden, das aus verschiedenen Schichten mit sogenannten Neuronen besteht. Dabei übernimmt jedes Neuron eine einfache mathematische Berechnung und gibt das Ergebnis an die Neuronen der nächsten Schicht weiter.

Durch die jüngsten Erfolge mag es überraschend klingen, dass die Ideen und Konzepte hinter neuronalen Netzen eigentlich schon relativ alt sind. Ein Blick auf die Geschichte und Konzepte von KNNs, die bereits in den 1940er Jahren ihren Anfang nahm, entmystifiziert die Themen KI, Deep Learning und neuronale Netze. Siehe dazu den Artikel "Am Anfang war das Neuron" auf Seite 21.

Vor allem in den letzten Jahren haben KNNs und Deep Neural Networks (DNNs) immer mehr an Popularität gewonnen und brachten große Erfolge in den Bereichen Bild- und Spracherkennung, Textverständnis und Übersetzung. Gründe dafür sind sicherlich auch die riesige Anzahl an verfügbaren Daten und die immensen, verfügbaren Rechenkapazitäten. Denn zum (An-)Lernen eines DNN mit vielen Layer und Neuronen sind viele Daten von guter Qualität und viel Rechenleistung notwendig – das gab es in den 1940er Jahren halt noch nicht!

## Das Gehirn des Rechners

Immer wieder fällt im Zusammenhang mit künstlichen neuronalen Netzen der Vergleich zum menschlichen Gehirn. Das bedeutet, dass das KNN dessen Funktionsweise oder besser der des Nervensystems nachempfunden ist. Letzteres verarbeitet,