Machine Learning - vor dem ersten Schritt: Wozu? Für wen?

Eine Einführung für Interessierte

Andy Yap; Gregor Schock; Fabio Ferreira - AKKA Automotive;
Jens Bruno Wittek - AKKA Digital

Maschinelles Lernen kann in zahlreichen Anwendungsfeldern Mehrwerte generieren. Neueinsteiger können mit den verwendeten Begriffen nicht immer etwas anfangen und werden teilweise durch „Marketing-Sprache“ verunsichert. Gerade Begriffe aus diesem Themenfeld sind derzeit allgegenwärtig und werden verallgemeinernd benutzt. In diesem Beitrag werden die Grundprinzipien des Machine Learning aus verschiedenen Perspektiven betrachtet und anschaulich erklärt. Es werden realistische Erwartungen an datenbasierte Analyseprojekte vermittelt.

Die Begriffe KÜNSTLICHE INTELLIGENZ (KI) oder auch ARTIFICIAL INTELLGENCE (AI) werden häufig genutzt - bedeuten das gleiche - und fassen alle Verfahren zusammen, die zum Ziel haben, eine Intelligenz in einem technischen System abzubilden. Eine sehr allgemeine Definition von Intelligenz ist die Fähigkeit komplexe Probleme zu lösen [1]. MACHINE LEARNING (ML) [2] ist der Oberbegriff für diejenigen Methoden von AI, welche Algorithmen automatisch durch Nutzung vorhandener Daten verbessern.
Ein wesentliches Ziel von ML ist die Konstruktion eines generalisierenden Systems. Dies wird gebildet, geformt und verbessert auf Basis relevanter Daten. Ein wichtiges ML-Konzept, welches auf einigen Eigenschaften natürlicher Nervensysteme basiert, ist die Nachbildung NEURONALER NETZE (NN). Diese mathematischen Nachbildungen werden als KÜNSTLICHE NEURONALE NETZE (KNN) bezeichnet.

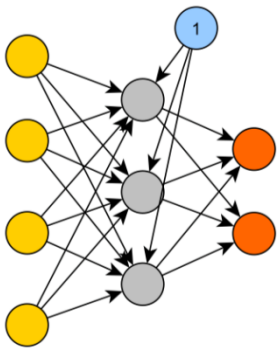

Abbildung 1: Schematische Darstellung eines KNN

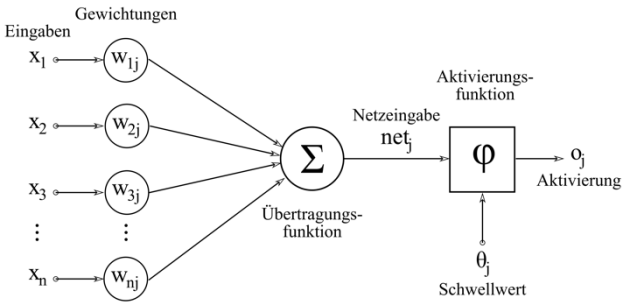

Abbildung. 2: Schematische Darstellung eines künstlichen Neurons mit dem Index j [a]

Anmerkung: KNN ist im Bereich der KI auch die Abkürzung für einen Algorithmus eines Klassifikationsverfahrens [3]. Dieses ist in diesem Beitrag nicht gemeint, veranschaulicht aber die Begriffsvielfalt in diesem Themenfeld. Biologische NN haben mit den hier beschriebenen KNN nur bedingt etwas gemeinsam, da sie lediglich als Vorlage für Aufbau und Funktionsweise dienen. Deshalb gibt es Netze, welche sich

ähnlich wie die biologischen Vorbilder verhalten. Es können aber auch andere Funktionsweisen über KNN abgebildet werden.
Ein KNN (Abbildung 1) besteht aus mehreren Neuronen, die in Form von Schichten oder engl. Layers (Input-, Hidden- und Output-Layer) angeordnet sind. Wir stellen hier das DEEP LEARNING (DL) [4] vor, bei dem die KNN mehrere verdeckte Zwischenschichten (Hidden Layer) enthalten. Es werden dabei drei Arten von Neuronen klassifiziert, die ihrer jeweiligen Schicht zugeordnet werden: Input-, Output- und Hidden-Neuron. Ein künstliches Neuron (Abbildung 2) errechnet einen einzigen Ausgangswert o, den es an alle mit ihm verbundenen Neuronen übergibt. Einer Verbindung zwischen zwei Neuronen ist ein Gewicht w zugeordnet, das die Abschwächung oder Verstärkung eines Signals erlaubt.

Lernen bedeutet sinnvolles Ändern der Gewichte bis ein Abbruchkriterium erreicht ist (Lernen durch Fehler). Dabei wird eine Ausgabe wiederholt berechnet und durch Anpassung der Gewichte optimiert. Abbruchkriterien dieser Lernphase können eine maximale Anzahl an Iterationen, eine gute Lösung (Optimum der Ausgabe) oder eine geringe Veränderung der Netzausgabe sein. Nach Beendigung der Lernphase repräsentieren alle Gewichte eines KNN das gelernte Wissen. Die zuvor erwähnte Tiefe eines KNN bezieht sich auf die Anzahl der verdeckten Schichten, wobei mit heutiger Rechenleistung mehrere hundert Schichten berechenbar sind.

Nutzen und Vergleich mit klassischen Lösungsmethoden
Anwendungsfelder für KNN gibt es vor allem dort, wo die Komplexität physikalischer oder mathematischer Modelle sehr hoch ist oder wo Details wichtig sind, die aber in einer Modellierung nicht beachtet werden können. Wichtige Nachteile von KNN sind der große Rechenaufwand, die große Menge der benötigten Daten sowie das Fehlen einer les- oder verstehbaren Grundstruktur.

Ein wichtiger Vorteil ist die universelle Einsetzbarkeit der KNN, da kein Systemwissen in die Erstellung einfließt. Außerdem können alle inhaltlich relevanten Daten unabhängig von ihrer Qualität (z.B. Rauschen, Auflösung, etc.) direkt verarbeitet werden.
Prinzipiell sind große Datenmengen eine Voraussetzung für die sinnvolle Anwendung von KNN. Um diese aber richtig zu nutzen, ist es sinnvoll ein eng verwandtes Gebiet vorzustellen: DATA MINING (DM). DM ist von den Methoden und Zielen eng mit dem ML verwandt.

Ablauf eines DM Projekts
Eine Möglichkeit für die Durchführung eines DM Projekts [5] ist der „Cross Industry Standard Process for Data Mining“ (Abbildung 3). Dieser zeigt die wesentlichen Schritte eines datenbasierten Projektes. Dabei fließen neue Erkenntnisse iterativ ein, um vorherige Schritte zu verbessern und noch einmal auszuführen.

Verständnis des Problems
Voraussetzung für den Erfolg eines Projekts ist ein umfassendes Wissen über die Fragestellung und die Ziele. Oft gibt es Vorgaben an die Analyse, wie möglichst gute Ergebnisse, eine gute Interpretierbarkeit oder Schnelligkeit. Die inhaltlichen Ziele gilt es in eine zu optimierende Größe zu übersetzen, die möglichst klar definiert oder gut zu messen ist und die Fragestellung möglichst gut repräsentiert.

Verständnis der Daten
Es gilt die Software zu analysieren, die an der Entstehung, Verarbeitung, Speicherung und Verwendung der Daten beteiligt ist oder war. Eventuell gibt es Hinweise darauf, dass Daten für den angedachten Use Case nicht repräsentativ sind oder Hinweise auf fehlerhafte Daten. Für ein erstes Verständnis eignen sich beschreibende statistische Kennzahlen, eine Ausreißeranalyse und explorative Grafiken.

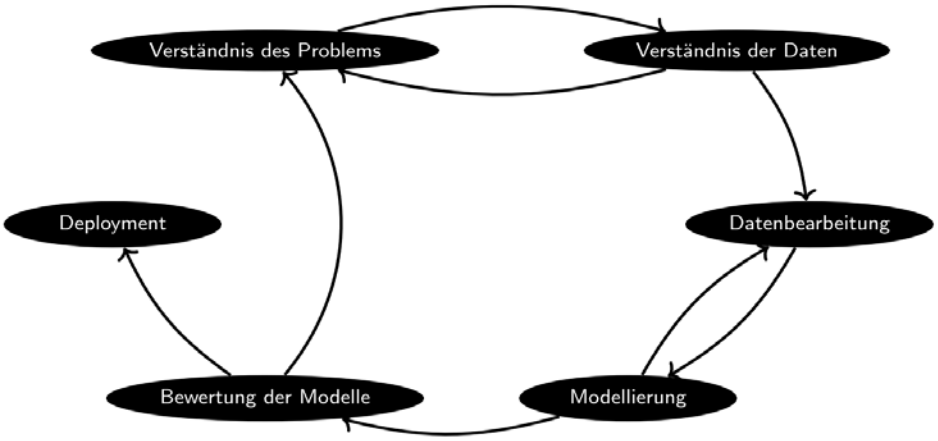

Abbildung 3: Ablauf eines DM Projektes: „Cross Industry Standard Process for Data Mining“ (CRISP-DM)

Datenbearbeitung
Es ist sinnvoll die Daten aufzuteilen (Abbildung 4), damit mit einem Teil der Daten die entwickelten Modelle (wie KNN) getestet und validiert werden können.

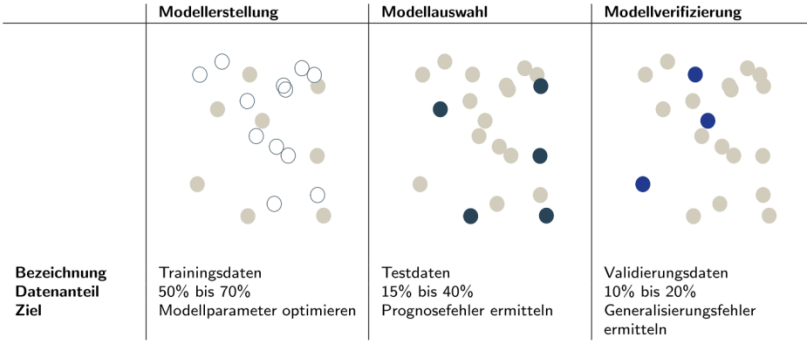

Abbildung 4: Datenaufteilung in Trainings- ,Test- und Validierungsdaten

Die Datenvorverarbeitung nimmt erfahrungsgemäß in vielen Projekten rund 80% der Zeit ein. Hier lassen sich große Verbesserungen der KI Systeme erzielen. Wichtige Schritte sind die Ersetzung fehlender oder falscher Werte (Imputation) oder Zusammenfassung von Variablenkategorien. Oft ist es nötig Messwerte in eine genormte Skala zu transformieren, um unterschiedliche Variablen vergleichbar zu machen. Auch kann es je nach Methode sinnvoll sein, stark korrelierte Variablen oder solche mit zu geringer Aussagekraft zu entfernen oder allgemein die Dimension zu reduzieren. Das Format von Einheiten, Encoding, Zeit- und Datumsvariablen muss immer beachtet werden.

Modellierung
Je nach Eigenschaften der Daten und Ziel der Analyse entscheidet sich, welche Verfahren sich am besten eignen (Abbildung 5). Existiert eine abhängige Variable, die es durch andere Variablen möglichst gut zu modellieren gilt, spricht man vom überwachten Lernen. Es handelt sich um Klassifikationsprobleme (Erkennung von Kunden, defekten Bauteilen, Objekten auf Bildern, etc.) mit abzählbarer abhängiger Variable (meist binär) oder um Regressionsprobleme (Vorhersage von Besucherzahlen, Umsatz, Restlebensdauer, etc.) mit kontinuierlicher abhängiger Variable.
Ohne abhängige Variable (unüberwachtes Lernen) kommt es auf die Zielsetzung an: Mit Clustering kann die Anzahl der Beobachtungen verringert oder in Gruppen eingeteilt werden. Beobachtungen in gleichen Clustern sollen ähnlich zueinander sein (Beispiele: Kundensegmentierung, Erkennung zueinander ähnlicher Texte und Bilder). Außerdem gibt es die Assoziationsanalyse, um Abhängigkeiten zu ermitteln, die innerhalb von Variablen oft zusammen beobachtet werden (Beispiele: Kaufverbundmuster, Empfehlungen für Produkte, Sehenswürdigkeiten oder Musik).

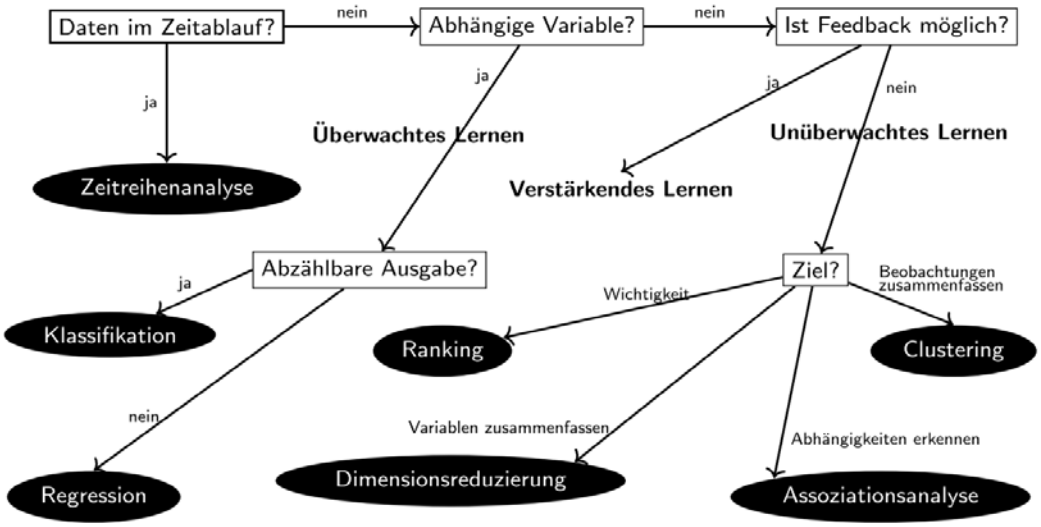

Abbildung 5: Methodenauswahl

Meistens sind Einsatzbereiche des ML in der Klassifikation und Regression, sodass eine abhängige Variable vorhanden ist und der Algorithmus anhand von Beispielen lernen kann. Die Initialisierung von Modellen sollte anhand wissenschaftlicher Kriterien erfolgen, zum Teil können Parameter aus den Daten geschätzt werden. Beim sogenannten Auswendiglernen kann das System schlechter mit unbekannten Daten umgehen (Overfitting).

Bewertung der Modelle
Die Bewertung der Modelle erfolgt nach statistischen Hypothesentests. Für überwachtes Lernen gibt es sinnvolle Klassifikations- und Regressionskennzahlen auf Grundlage der Unterschiede zwischen beobachteten und prognostizierten Testdaten.

Deployment
Sobald das Modell fertig gelernt hat, kann es mit neuen Daten gefüttert werden und liefert meist in Sekundenbruchteilen ein Ergebnis. In der Regel bietet sich eine Pilotphase in einem kleinen Bereich mit quantitativer (Monitoring) und qualitativer (Gespräche mit Anwendern) Bewertung an. Langfristig ist teilweise eine Überset-

zung von Prototypen in schnellere, maschinennahe Programmiersprachen sinnvoll. Ein Modellverfall (Verschlechterung der Ergebnisse mit neuen Daten) ist möglich, wenn sich Rahmenbedingungen ändern. Es empfiehlt sich neue Daten regelmäßig ins Modell aufzunehmen und die Modelle neu zu trainieren.

Beispiel: Objekt-Detektion auf einem Smartphone
Google hat eine latenzoptimierte bildbasierte Objektdetektionsanwendung auf einem Smartphone gezeigt [6]. Eine Objektdetektion (Abbildung 6) beinhaltet in der Regel eine Klassifikation (z.B. Mensch oder Auto) sowie die Lokalisierung der Objekte und deren Hervorhebung durch eine Umrahmung im Bildstream (Bounding Box).

Abbildung 6: Example objection detection results using MobileNet SSD [b]

In diesem Projekt wurden MobileNets [7] genutzt, eine eigens entworfene Familie von KNN zur Lösung von Detektions-, Erkennungs- und Klassifikationsproblemen, die sich besonders gut für die Portierung auf Mobilgeräte eignen, indem die Modellgenauigkeit unter den Kriterien geringer Stromverbrauch, Latenzzeit sowie Speicherverbrauch maximiert wird. Ein Parameter erlaubt dabei die Abwägung zwischen diesen Kriterien und die Anpassung der Modelleigenschaften und Größe an das verfügbare System.

Ein auf einem sehr großen Bilddatensatz vortrainiertes MobileNet wurde hierbei für die Objektdetektionsaufgabe verwendet. Diese Methode wird Transfer Learning genannt und erlaubt es, die Trainingszeit von KNN erheblich zu reduzieren. Um die Latenzzeiten des MobileNets noch weiter zu reduzieren, wurden die Aktivierungen und Gewichte des KNN quantisiert, indem sie von floating-point-basierte auf 8-Bit integer-basierte Inferenz umgestellt wurden [8], dadurch werden bessere Latenzzeiten auf beispielsweise ARM CPUs ermöglicht.

Anstelle von CPUs kann für die Berechnung der KNN auch auf andere Hardware zurückgegriffen werden. Für den Embedded-Gebrauch bietet beispielsweise Nvidia mit der Drive PX-Serie eine GPU-Lösung an, wohingegen Xilinx mit den Zynq Ultrascale+ MPSoC Chips eine FPGA-Lösung im Angebot hat. Der Vorteil dieser Hardwarelösungen besteht darin, dass die unzähligen Rechenoperationen parallel abgearbeitet werden können, wohingegen eine CPU die Aufgaben seriell löst.

Ausblick

Wie das Beispiel von Google zeigt, ist die Verwendung von Deep Learning mit neuronalen Netzwerken als Teilbereich der AI mit zunehmender Rechenleistung immer erschwinglicher und die Steigerung der Lernleistungen erschließt dabei bisher ungenutzte und neue Anwendungsfelder. Für eine gute Generalisierungsleistung dieser vielschichtigen Netze sind repräsentative und vor allem große Datenmengen Voraussetzung. Die Verwendung spezialisierter, vortrainierter Netze verkürzt dabei den sonst benötigten Zeitaufwand enorm.

Aktuell sind viele Themen im Bereich der Produktentwicklung in Bearbeitung, es gilt die Konzepte aus Forschung und Entwicklung serienreif zu machen. Hier kommt die gesamte Bandbreite der Entwicklung von Embedded-Systemen zu Einsatz, beginnend bei der Hardwareauswahl und dem Bedarf hochwertiger Software-Implementierungen.

In Zukunft werden sicherlich noch viele weitere Anwendungsmöglichkeiten erschlossen und weitere Verbesserungen der Performance und der inhaltlichen Leistungen von KI Systemen erzielt. Um das Themenfeld weiter zu erschließen, eignen sich die im Literaturverzeichnis genannten Publikationen gut als Ausgangspunkt.

Literaturverzeichnis

[1] Max Tegmark; “Leben 3.0: Mensch sein im Zeitalter Künstlicher Intelligenz”; Ullstein Verlag; 2017

[2] Christopher Bishop. “Pattern Recognition and Machine Learning”. Springer, 2006.

[3] Evelyn Fix and Joseph L. Hodges Jr. “Discriminatory analysis-nonparametric discrimination: consistency properties”. University of California Berkeley, 1951.

[4] Ian Goodfellow et al., “Deep Learning”. MIT Press, 2016

[5] C. Shearer, “The CRISP-DM model: the new blueprint for data mining”, Journal of Data Warehousing ; vol. 5, no 4: pp 13—22, 2000

[6] Google AI Blog: Posted by Jonathan Huang; “Accelerated Training and Inference with the Tensorflow Object Detection API”, 13.07.2018 [Stand 01.10.2018]. URL: https://ai.googleblog.com/2018/07/accelerated-training-and-inference-with.html

[7] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam. Google Inc.: „MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications“. arXiv preprint arXiv: 1704.04861v1, 2017. URL: https://arxiv.org/pdf/1704.04861.pdf

[8] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, D. Kalenichenko. Google Inc.: “Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference”, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2704-2713. URL: https://arxiv.org/pdf/1712.05877.pdf

Bildquellenverzeichnis

[a] Wikimedia: User Chrislb, Perhelion; “Schematische Darstellung eines künstlichen Neurons mit dem Index j.”, 08.10.2010, 19:52 [Stand 24.09.2018]. URL: https://commons.wikimedia.org/wiki/File:NeuronModel_deutsch.svg

[b] Photo by Juanedc (CC BY 2.0); Figure 6 aus [7]

Autoren

Dr.-Ing. Andy Yap ist Senior Manager Software Solutions bei AKKA Automotive, er verfügt über eine mehr als 20-jährige Erfahrung in den Bereichen Softwareentwicklung und Funktionsentwicklung im Automotive Umfeld. Schwerpunkte sind die Themenfelder Fahrzeugantriebe, Fahrdynamik, Fahrerassistenz und autonomes Fahren.
Kontakt: andy.yap@akka.eu

Fabio Ferreira steht kurz vor seinem Master-Abschluss in Informatik und legt seit drei Jahren seinen fachlichen Fokus auf Machine Learning. Am KIT forscht er am Institut für humanoide Robotik im Bereich "Machine Learning for Robotic Perception and Cognition". Seine Masterarbeit über Deep Reinforcement Learning verfasst er seit Oktober am Stanford AI Lab. Seit 2013 betreut er als Werkstudent bei AKKA Automotive interne Toolentwicklungen.
Kontakt: fabio.ferreira@mbtech-group.com

Jens Bruno Wittek ist seit Januar 2017 Data Scientist bei AKKA Digital und analysiert hauptsächlich Daten aus dem Automobilbereich. Zuvor arbeitete er im Big-Data-Team von Daimler TSS. Er hat einen Master in Statistik und ist Experte in Machine Learning, Statistik und R-Programmierung.
Kontakt: jensbruno.wittek@akka.eu

Dipl.-Ing tech. kyb. Gregor Schock ist Entwicklungsingenieur und Funktionsentwickler bei AKKA. Im Studium beschäftigte er sich mit biologischen und künstlichen Neuronalen Netzen. Seit 1996 erarbeitet er Lösungen im Bereich Datenanalyse, Modellbildung, Simulations- und Regelungstechnik als Funktionsentwickler im automobilen Umfeld.
Kontakt: gregor.schock@mbtech-group.com