

O'REILLY®

2. Auflage  
Aktuell zu  
TensorFlow 2

# Praxiseinstieg Machine Learning mit Scikit-Learn, Keras und TensorFlow

Konzepte, Tools und Techniken  
für intelligente Systeme

powered by



Aurélien Géron

Übersetzung von Kristian Rother  
und Thomas Demmig

Papier  
plus<sup>+</sup>  
PDF.

Zu diesem Buch – sowie zu vielen weiteren O'Reilly-Büchern – können Sie auch das entsprechende E-Book im PDF-Format herunterladen. Werden Sie dazu einfach Mitglied bei oreilly.plus<sup>+</sup>:

[www.oreilly.plus](http://www.oreilly.plus)

2. AUFLAGE

---

# Praxiseinstieg Machine Learning mit Scikit-Learn, Keras und TensorFlow

*Konzepte, Tools und Techniken  
für intelligente Systeme*

*Aurélien Géron*

*Deutsche Übersetzung von  
Kristian Rother & Thomas Demmig*

**O'REILLY®**

Aurélien Géron

Lektorat: Alexandra Follenius

Übersetzung: Kristian Rother, Thomas Demmig

Korrektorat: Sibylle Feldmann, [www.richtiger-text.de](http://www.richtiger-text.de)

Satz: III-satz, [www.drei-satz.de](http://www.drei-satz.de)

Herstellung: Stefanie Weidner

Umschlaggestaltung: Karen Montgomery, Michael Oréal, [www.oreal.de](http://www.oreal.de)

Druck und Bindung: mediaprint solutions GmbH, 33100 Paderborn

Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

ISBN:

Print 978-3-96009-124-0

PDF 978-3-96010-339-4

ePub 978-3-96010-340-0

mobi 978-3-96010-341-7

2. Auflage

Translation Copyright für die deutschsprachige Ausgabe © 2020 dpunkt.verlag GmbH

Wieblinger Weg 17

69123 Heidelberg

Authorized German translation of the English edition of *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition*, ISBN 9781492032649 © 2019 Kiwisoft S.A.S. This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

Dieses Buch erscheint in Kooperation mit O'Reilly Media, Inc. unter dem Imprint »O'REILLY«. O'REILLY ist ein Markenzeichen und eine eingetragene Marke von O'Reilly Media, Inc. und wird mit Einwilligung des Eigentümers verwendet.

*Hinweis:*

Dieses Buch wurde auf PEFC-zertifiziertem Papier aus nachhaltiger Waldwirtschaft gedruckt. Der Umwelt zuliebe verzichten wir zusätzlich auf die Einschweißfolie.



*Schreiben Sie uns:*

Falls Sie Anregungen, Wünsche und Kommentare haben, lassen Sie es uns wissen: [komentar@oreilly.de](mailto:komentar@oreilly.de).

Die vorliegende Publikation ist urheberrechtlich geschützt. Alle Rechte vorbehalten. Die Verwendung der Texte und Abbildungen, auch auszugsweise, ist ohne die schriftliche Zustimmung des Verlags urheberrechtswidrig und daher strafbar. Dies gilt insbesondere für die Vervielfältigung, Übersetzung oder die Verwendung in elektronischen Systemen.

Es wird darauf hingewiesen, dass die im Buch verwendeten Soft- und Hardware-Bezeichnungen sowie Markennamen und Produktbezeichnungen der jeweiligen Firmen im Allgemeinen warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.

Alle Angaben und Programme in diesem Buch wurden mit größter Sorgfalt kontrolliert. Weder Autor noch Verlag noch Übersetzer können jedoch für Schäden haftbar gemacht werden, die in Zusammenhang mit der Verwendung dieses Buches stehen.

5 4 3 2 1 0

<b>Vorwort</b> .....	<b>XVII</b>
----------------------	-------------

---

## **Teil I Die Grundlagen des Machine Learning**

<b>1 Die Machine-Learning-Umgebung</b> .....	<b>3</b>
Was ist Machine Learning? .....	4
Warum wird Machine Learning verwendet? .....	4
Anwendungsbeispiel .....	7
Unterschiedliche Machine-Learning-Systeme .....	9
Überwachtes/unüberwachtes Lernen .....	9
Batch- und Online-Learning .....	16
Instanzbasiertes versus modellbasiertes Lernen .....	18
Die wichtigsten Herausforderungen beim Machine Learning .....	24
Unzureichende Menge an Trainingsdaten .....	24
Nicht repräsentative Trainingsdaten .....	26
Minderwertige Daten .....	27
Irrelevante Merkmale .....	28
Overfitting der Trainingsdaten .....	28
Underfitting der Trainingsdaten .....	30
Zusammenfassung .....	31
Testen und Validieren .....	31
Hyperparameter anpassen und Modellauswahl .....	32
Datendiskrepanz .....	33
Übungen .....	34
<b>2 Ein Machine-Learning-Projekt von A bis Z</b> .....	<b>37</b>
Der Umgang mit realen Daten .....	37
Betrachte das Gesamtbild .....	39
Die Aufgabe abstecken .....	39

Wähle ein Qualitätsmaß aus . . . . .	41
Überprüfe die Annahmen . . . . .	44
Beschaffe die Daten. . . . .	44
Erstelle eine Arbeitsumgebung. . . . .	44
Die Daten herunterladen . . . . .	48
Wirf einen kurzen Blick auf die Datenstruktur . . . . .	49
Erstelle einen Testdatensatz. . . . .	53
Erkunde und visualisiere die Daten, um Erkenntnisse zu gewinnen . . . . .	57
Visualisieren geografischer Daten . . . . .	58
Suche nach Korrelationen . . . . .	60
Experimentieren mit Kombinationen von Merkmalen . . . . .	63
Bereite die Daten für Machine-Learning-Algorithmen vor . . . . .	64
Aufbereiten der Daten . . . . .	64
Bearbeiten von Text und kategorischen Merkmalen . . . . .	67
Eigene Transformer . . . . .	70
Skalieren von Merkmalen . . . . .	71
Pipelines zur Transformation. . . . .	72
Wähle ein Modell aus und trainiere es . . . . .	74
Trainieren und Auswerten auf dem Trainingsdatensatz . . . . .	74
Bessere Auswertung mittels Kreuzvalidierung . . . . .	76
Optimiere das Modell. . . . .	78
Gittersuche. . . . .	78
Zufällige Suche. . . . .	80
Ensemble-Methoden . . . . .	81
Analysiere die besten Modelle und ihre Fehler . . . . .	81
Evaluere das System auf dem Testdatensatz. . . . .	82
Nimm das System in Betrieb, überwache und warte es . . . . .	83
Probieren Sie es aus! . . . . .	86
Übungen . . . . .	87
<b>3 Klassifikation . . . . .</b>	<b>89</b>
MNIST . . . . .	89
Trainieren eines binären Klassifikators. . . . .	91
Qualitätsmaße. . . . .	92
Messen der Genauigkeit über Kreuzvalidierung . . . . .	92
Konfusionsmatrix. . . . .	94
Relevanz und Sensitivität . . . . .	96
Die Wechselbeziehung zwischen Relevanz und Sensitivität . . . . .	97
Die ROC-Kurve . . . . .	100
Klassifikatoren mit mehreren Kategorien . . . . .	103
Fehleranalyse . . . . .	106
Klassifikation mit mehreren Labels. . . . .	109

Klassifikation mit mehreren Ausgaben .....	110
Übungen .....	112
<b>4 Trainieren von Modellen .....</b>	<b>115</b>
Lineare Regression .....	116
Die Normalengleichung .....	118
Komplexität der Berechnung .....	120
Das Gradientenverfahren .....	121
Batch-Gradientenverfahren .....	124
Stochastisches Gradientenverfahren .....	127
Mini-Batch-Gradientenverfahren .....	130
Polynomielle Regression .....	131
Lernkurven .....	133
Regularisierte lineare Modelle .....	137
Ridge-Regression .....	137
Lasso-Regression .....	139
Elastic Net .....	142
Early Stopping .....	143
Logistische Regression .....	144
Abschätzen von Wahrscheinlichkeiten .....	145
Trainieren und Kostenfunktion .....	146
Entscheidungsgrenzen .....	147
Softmax-Regression .....	149
Übungen .....	153
<b>5 Support Vector Machines .....</b>	<b>155</b>
Lineare Klassifikation mit SVMs .....	155
Soft-Margin-Klassifikation .....	156
Nichtlineare SVM-Klassifikation .....	159
Polynomieller Kernel .....	160
Ähnlichkeitsbasierte Merkmale .....	161
Der gaußsche RBF-Kernel .....	162
Komplexität der Berechnung .....	163
SVM-Regression .....	164
Hinter den Kulissen .....	166
Entscheidungsfunktion und Vorhersagen .....	166
Zielfunktionen beim Trainieren .....	167
Quadratische Programme .....	169
Das duale Problem .....	170
Kernel-SVM .....	171
Online-SVMs .....	173
Übungen .....	175

<b>6</b>	<b>Entscheidungsbäume</b> .....	<b>177</b>
	Trainieren und Visualisieren eines Entscheidungsbaums .....	177
	Vorhersagen treffen .....	178
	Schätzen von Wahrscheinlichkeiten für Kategorien .....	181
	Der CART-Trainingsalgorithmus .....	181
	Komplexität der Berechnung .....	182
	Gini-Unreinheit oder Entropie? .....	182
	Hyperparameter zur Regularisierung .....	183
	Regression .....	185
	Instabilität .....	187
	Übungen .....	188
<b>7</b>	<b>Ensemble Learning und Random Forests</b> .....	<b>191</b>
	Abstimmverfahren unter Klassifikatoren .....	192
	Bagging und Pasting .....	195
	Bagging und Pasting in Scikit-Learn .....	196
	Out-of-Bag-Evaluation .....	197
	Zufällige Patches und Subräume .....	198
	Random Forests .....	199
	Extra-Trees .....	200
	Wichtigkeit von Merkmalen .....	200
	Boosting .....	202
	AdaBoost .....	202
	Gradient Boosting .....	205
	Stacking .....	210
	Übungen .....	213
<b>8</b>	<b>Dimensionsreduktion</b> .....	<b>215</b>
	Der Fluch der Dimensionalität .....	216
	Die wichtigsten Ansätze zur Dimensionsreduktion .....	217
	Projektion .....	217
	Manifold Learning .....	219
	Hauptkomponentenzerlegung (PCA) .....	221
	Erhalten der Varianz .....	221
	Hauptkomponenten .....	222
	Die Projektion auf d Dimensionen .....	223
	Verwenden von Scikit-Learn .....	224
	Der Anteil erklärter Varianz .....	224
	Auswählen der richtigen Anzahl Dimensionen .....	225
	PCA als Komprimierungsverfahren .....	226
	Randomisierte PCA .....	227
	Inkrementelle PCA .....	227

Kernel-PCA .....	228
Auswahl eines Kernels und Optimierung der Hyperparameter . . . .	229
LLE .....	231
Weitere Techniken zur Dimensionsreduktion .....	233
Übungen .....	234
<b>9 Techniken des unüberwachten Lernens .....</b>	<b>237</b>
Clustering .....	238
K-Means .....	240
Grenzen von K-Means .....	250
Bildsegmentierung per Clustering .....	251
Vorverarbeitung per Clustering .....	253
Clustering für teilüberwachtes Lernen einsetzen .....	254
DBSCAN .....	257
Andere Clustering-Algorithmen .....	260
Gaußsche Mischverteilung .....	262
Anomalieerkennung mit gaußschen Mischverteilungsmodellen . . .	267
Die Anzahl an Clustern auswählen .....	269
Bayessche gaußsche Mischverteilungsmodelle .....	272
Andere Algorithmen zur Anomalie- und Novelty-Erkennung . . . .	276
Übungen .....	277

---

## Teil II Neuronale Netze und Deep Learning

<b>10 Einführung in künstliche neuronale Netze mit Keras .....</b>	<b>281</b>
Von biologischen zu künstlichen Neuronen .....	282
Biologische Neuronen .....	283
Logische Berechnungen mit Neuronen .....	285
Das Perzeptron .....	286
Mehrschichtiges Perzeptron und Backpropagation .....	290
Regressions-MLPs .....	294
Klassifikations-MLPs .....	295
MLPs mit Keras implementieren .....	297
TensorFlow 2 installieren .....	298
Einen Bildklassifikator mit der Sequential API erstellen .....	299
Ein Regressions-MLP mit der Sequential API erstellen .....	309
Komplexe Modelle mit der Functional API bauen .....	310
Dynamische Modelle mit der Subclassing API bauen .....	315
Ein Modell sichern und wiederherstellen .....	316
Callbacks .....	317
TensorBoard zur Visualisierung verwenden .....	318

Feinabstimmung der Hyperparameter eines neuronalen Netzes . . . . .	322
Anzahl verborgener Schichten . . . . .	326
Anzahl Neuronen pro verborgene Schicht . . . . .	327
Lernrate, Batchgröße und andere Hyperparameter . . . . .	328
Übungen . . . . .	330
<b>11 Trainieren von Deep-Learning-Netzen . . . . .</b>	<b>333</b>
Das Problem schwindender/explodierender Gradienten . . . . .	334
Initialisierung nach Glorot und He . . . . .	335
Nicht sättigende Aktivierungsfunktionen . . . . .	337
Batchnormalisierung . . . . .	341
Gradient Clipping . . . . .	347
Wiederverwenden vortrainierter Schichten . . . . .	348
Transfer Learning mit Keras. . . . .	350
Unüberwachtes Vortrainieren . . . . .	352
Vortrainieren anhand einer Hilfsaufgabe. . . . .	353
Schnellere Optimierer . . . . .	354
Momentum Optimization . . . . .	354
Beschleunigter Gradient nach Nesterov. . . . .	356
AdaGrad. . . . .	357
RMSProp . . . . .	358
Adam-Optimierung . . . . .	359
Scheduling der Lernrate . . . . .	362
Vermeiden von Overfitting durch Regularisierung. . . . .	367
$\ell_1$ - und $\ell_2$ -Regularisierung . . . . .	367
Drop-out . . . . .	368
Monte-Carlo-(MC-)-Drop-out. . . . .	371
Max-Norm-Regularisierung. . . . .	374
Zusammenfassung und praktische Tipps . . . . .	374
Übungen . . . . .	376
<b>12 Eigene Modelle und Training mit TensorFlow . . . . .</b>	<b>379</b>
Ein kurzer Überblick über TensorFlow . . . . .	379
TensorFlow wie NumPy einsetzen . . . . .	383
Tensoren und Operationen . . . . .	383
Tensoren und NumPy . . . . .	385
Typumwandlung . . . . .	385
Variablen . . . . .	386
Andere Datenstrukturen . . . . .	387
Modelle und Trainingsalgorithmen anpassen. . . . .	388
Eigene Verlustfunktion . . . . .	388
Modelle mit eigenen Komponenten sichern und laden . . . . .	389

Eigene Aktivierungsfunktionen, Initialisierer, Regularisierer und Constraints .....	391
Eigene Metriken .....	392
Eigene Schichten .....	395
Eigene Modelle .....	398
Verlustfunktionen und Metriken auf Modell-Internen basieren lassen. ....	400
Gradienten per Autodiff berechnen .....	402
Eigene Trainingsschleifen .....	406
Funktionen und Graphen in TensorFlow .....	409
AutoGraph und Tracing .....	411
Regeln für TF Functions .....	412
Übungen .....	414
<b>13 Daten mit TensorFlow laden und vorverarbeiten .....</b>	<b>417</b>
Die Data-API .....	418
Transformationen verketteten .....	419
Daten durchmischen .....	420
Daten vorverarbeiten .....	423
Alles zusammenbringen .....	424
Prefetching .....	425
Datasets mit tf.keras verwenden .....	427
Das TFRecord-Format .....	428
Komprimierte TFRecord-Dateien .....	429
Eine kurze Einführung in Protocol Buffer .....	429
TensorFlow-Protobufs .....	431
Examples laden und parsen .....	432
Listen von Listen mit dem SequenceExample-Protobuf verarbeiten .....	433
Die Eingabemerkmale vorverarbeiten .....	434
Kategorische Merkmale mit One-Hot-Vektoren codieren .....	435
Kategorische Merkmale mit Embeddings codieren .....	437
Vorverarbeitungsschichten von Keras .....	441
TF Transform .....	443
Das TensorFlow-Datasets-(TFDS-)Projekt .....	445
Übungen .....	446
<b>14 Deep Computer Vision mit Convolutional Neural Networks .....</b>	<b>449</b>
Der Aufbau des visuellen Cortex .....	450
Convolutional Layers .....	451
Filter .....	453
Stapeln mehrerer Feature Maps .....	454

Implementierung in TensorFlow . . . . .	456
Speicherbedarf . . . . .	459
Pooling Layers . . . . .	460
Implementierung in TensorFlow . . . . .	462
Architekturen von CNNs . . . . .	464
LeNet-5 . . . . .	466
AlexNet . . . . .	467
GoogLeNet . . . . .	470
VGGNet . . . . .	473
ResNet . . . . .	474
Xception . . . . .	477
SENet . . . . .	479
Ein ResNet-34-CNN mit Keras implementieren . . . . .	481
Vortrainierte Modelle aus Keras einsetzen . . . . .	482
Vortrainierte Modelle für das Transfer Learning . . . . .	484
Klassifikation und Lokalisierung . . . . .	487
Objekterkennung . . . . .	488
Fully Convolutional Networks . . . . .	490
You Only Look Once (YOLO) . . . . .	492
Semantische Segmentierung . . . . .	495
Übungen . . . . .	499
<b>15 Verarbeiten von Sequenzen mit RNNs und CNNs . . . . .</b>	<b>501</b>
Rekurrente Neuronen und Schichten . . . . .	502
Gedächtniszellen . . . . .	504
Ein- und Ausgabesequenzen . . . . .	505
RNNs trainieren . . . . .	506
Eine Zeitserie vorhersagen . . . . .	507
Grundlegende Metriken . . . . .	508
Ein einfaches RNN implementieren . . . . .	509
Deep RNNs . . . . .	510
Mehrere Zeitschritte vorhersagen . . . . .	512
Arbeit mit langen Sequenzen . . . . .	515
Gegen instabile Gradienten kämpfen . . . . .	516
Das Problem des Kurzzeitgedächtnisses . . . . .	518
Übungen . . . . .	527
<b>16 Natürliche Sprachverarbeitung mit RNNs und Attention . . . . .</b>	<b>529</b>
Shakespearsche Texte mit einem Character-RNN erzeugen . . . . .	530
Den Trainingsdatensatz erstellen . . . . .	531
Wie ein sequenzieller Datensatz aufgeteilt wird . . . . .	532
Den sequenziellen Datensatz in mehrere Fenster unterteilen . . . . .	533

Das Char-RNN-Modell bauen und trainieren . . . . .	535
Das Char-RNN-Modell verwenden . . . . .	535
Einen gefälschten Shakespeare-Text erzeugen . . . . .	536
Zustandsbehaftetes RNN . . . . .	537
Sentimentanalyse . . . . .	539
Maskieren . . . . .	543
Vortrainierte Embeddings wiederverwenden . . . . .	545
Ein Encoder-Decoder-Netzwerk für die neuronale maschinelle	
Übersetzung . . . . .	547
Bidirektionale RNNs. . . . .	550
Beam Search . . . . .	551
Attention-Mechanismen . . . . .	553
Visuelle Attention . . . . .	556
Attention Is All You Need: Die Transformer-Architektur . . . . .	558
Aktuelle Entwicklungen bei Sprachmodellen . . . . .	566
Übungen . . . . .	568
<b>17 Representation Learning und Generative Learning mit Autoencodern</b>	
<b>und GANs. . . . .</b>	<b>571</b>
Effiziente Repräsentation von Daten . . . . .	572
Hauptkomponentenzerlegung mit einem unvollständigen	
linearen Autoencoder . . . . .	574
Stacked Autoencoder. . . . .	575
Einen Stacked Autoencoder mit Keras implementieren . . . . .	576
Visualisieren der Rekonstruktionen . . . . .	577
Den Fashion-MNIST-Datensatz visualisieren. . . . .	578
Unüberwachtes Vortrainieren mit Stacked Autoencoder . . . . .	579
Kopplung von Gewichten. . . . .	580
Trainieren mehrerer Autoencoder nacheinander . . . . .	582
Convolutional Autoencoder . . . . .	583
Rekurrente Autoencoder . . . . .	584
Denoising Autoencoder. . . . .	584
Sparse Autoencoder. . . . .	586
Variational Autoencoder . . . . .	589
Fashion-MNIST-Bilder erzeugen . . . . .	593
Generative Adversarial Networks . . . . .	595
Schwierigkeiten beim Trainieren von GANs . . . . .	599
Deep Convolutional GANs. . . . .	601
Progressive wachsende GANs. . . . .	604
StyleGANs . . . . .	607
Übungen . . . . .	610

<b>18 Reinforcement Learning</b> .....	<b>611</b>
Lernen zum Optimieren von Belohnungen. ....	612
Suche nach Policies. ....	613
Einführung in OpenAI Gym .....	615
Neuronale Netze als Policies. ....	619
Auswerten von Aktionen: Das Credit-Assignment-Problem .....	621
Policy-Gradienten .....	622
Markov-Entscheidungsprozesse .....	627
Temporal Difference Learning .....	631
Q-Learning .....	632
Erkundungspolicies .....	634
Approximatives Q-Learning und Deep-Q-Learning .....	634
Deep-Q-Learning implementieren .....	636
Deep-Q-Learning-Varianten .....	640
Feste Q-Wert-Ziele. ....	640
Double DQN .....	641
Priorisiertes Experience Replay .....	642
Dueling DQN. ....	643
Die TF-Agents-Bibliothek. ....	644
TF-Agents installieren .....	645
TF-Agents-Umgebungen .....	645
Umgebungsspezifikationen .....	646
Umgebungswrapper und Atari-Vorverarbeitung. ....	647
Trainingsarchitektur .....	650
Deep-Q-Netz erstellen .....	652
DQN-Agenten erstellen .....	654
Replay Buffer und Beobachter erstellen. ....	655
Trainingsmetriken erstellen .....	657
Collect-Fahrer erstellen .....	658
Dataset erstellen .....	659
Trainingsschleife erstellen .....	662
Überblick über beliebte RL-Algorithmen .....	664
Übungen .....	666
<b>19 TensorFlow-Modelle skalierbar trainieren und deployen</b> .....	<b>667</b>
Ein TensorFlow-Modell ausführen. ....	668
TensorFlow Serving verwenden. ....	668
Einen Vorhersageservice auf der GCP AI Platform erstellen .....	677
Den Vorhersageservice verwenden. ....	682
Ein Modell auf ein Mobile oder Embedded Device deployen. ....	685
Mit GPUs die Berechnungen beschleunigen. ....	689
Sich eine eigene GPU zulegen .....	690

Eine mit GPU ausgestattete virtuelle Maschine einsetzen . . . . .	693
Colaboratory . . . . .	694
Das GPU-RAM verwalten . . . . .	695
Operationen und Variablen auf Devices verteilen . . . . .	698
Paralleles Ausführen auf mehreren Devices . . . . .	700
Modelle auf mehreren Devices trainieren . . . . .	702
Parallelisierte Modelle . . . . .	703
Parallelisierte Daten . . . . .	705
Mit der Distribution Strategies API auf mehreren Devices trainieren . . . . .	710
Ein Modell in einem TensorFlow-Cluster trainieren . . . . .	712
Große Trainingsjobs auf der Google Cloud AI Platform ausführen . . . . .	715
Black Box Hyperparameter Tuning auf der AI Platform . . . . .	717
Übungen . . . . .	718
Vielen Dank! . . . . .	719
<b>A Lösungen zu den Übungsaufgaben . . . . .</b>	<b>721</b>
<b>B Checkliste für Machine-Learning-Projekte . . . . .</b>	<b>759</b>
<b>C Das duale Problem bei SVMs . . . . .</b>	<b>765</b>
<b>D Autodiff . . . . .</b>	<b>769</b>
<b>E Weitere verbreitete Architekturen neuronaler Netze . . . . .</b>	<b>777</b>
<b>F Spezielle Datenstrukturen . . . . .</b>	<b>787</b>
<b>G TensorFlow-Graphen . . . . .</b>	<b>795</b>
<b>Index . . . . .</b>	<b>805</b>



## Der Machine-Learning-Tsunami

Im Jahr 2006 erschien ein Artikel (<https://homl.info/136>) von Geoffrey Hinton et al.,<sup>1</sup> in dem vorgestellt wurde, wie sich ein neuronales Netz zum Erkennen handgeschriebener Ziffern mit ausgezeichneter Genauigkeit (> 98%) trainieren lässt. Ein Deep Neural Network ist ein (sehr) vereinfachtes Modell unseres zerebralen Kortex, und es besteht aus einer Folge von Schichten mit künstlichen Neuronen. Die Autoren nannten diese Technik »Deep Learning«. Zu dieser Zeit wurde das Trainieren eines Deep-Learning-Netzes im Allgemeinen als unmöglich angesehen,<sup>2</sup> und die meisten Forscher hatten die Idee in den 1990ern aufgegeben. Dieser Artikel ließ das Interesse der wissenschaftlichen Gemeinde wieder aufleben, und schon nach kurzer Zeit zeigten weitere Artikel, dass Deep Learning nicht nur möglich war, sondern umwerfende Dinge vollbringen konnte, zu denen kein anderes Machine-Learning-(ML-)Verfahren auch nur annähernd in der Lage war (mithilfe enormer Rechenleistung und riesiger Datenmengen). Dieser Enthusiasmus breitete sich schnell auf weitere Teilgebiete des Machine Learning aus.

Zehn Jahre später hat Machine Learning ganze Industriezweige erobert: Es ist zu einem Herzstück heutiger Spitzentechnologien geworden und dient dem Ranking von Suchergebnissen im Web, kümmert sich um die Spracherkennung Ihres Smartphones, gibt Empfehlungen für Videos und schlägt den Weltmeister im Brettspiel Go. Über kurz oder lang wird ML vermutlich auch Ihr Auto steuern.

---

1 Geoffrey Hinton et al., »A Fast Learning Algorithm for Deep Belief Nets«, *Neural Computation* 18 (2006): 1527–1554.

2 Obwohl die Konvolutionsnetze von Yann Lecun bei der Bilderkennung seit den 1990ern gut funktioniert hatten, auch wenn sie nicht allgemein anwendbar waren.

# Machine Learning in Ihren Projekten

Deshalb interessieren Sie sich natürlich auch für Machine Learning und möchten an der Party teilnehmen!

Womöglich möchten Sie Ihrem selbst gebauten Roboter einen eigenen Denkapparat geben? Ihn Gesichter erkennen lassen? Oder lernen lassen, herumzulaufen?

Oder vielleicht besitzt Ihr Unternehmen Unmengen an Daten (Logdateien, Finanzdaten, Produktionsdaten, Sensordaten, Hotline-Statistiken, Personalstatistiken und so weiter), und Sie könnten vermutlich einige verborgene Schätze heben, wenn Sie nur wüssten, wo Sie danach suchen müssten, beispielsweise:

- Kundensegmente finden und für jede Gruppe die beste Marketingstrategie entwickeln.
- Jedem Kunden anhand des Kaufverhaltens ähnlicher Kunden Produktempfehlungen geben.
- Betrügerische Transaktionen mit hoher Wahrscheinlichkeit erkennen.
- Den Unternehmensgewinn im nächsten Jahr vorhersagen.

Was immer der Grund ist, Sie haben beschlossen, Machine Learning zu erlernen und in Ihren Projekten umzusetzen. Eine ausgezeichnete Idee!

## Ziel und Ansatz

Dieses Buch geht davon aus, dass Sie noch so gut wie nichts über Machine Learning wissen. Unser Ziel ist es, Ihnen die Grundbegriffe, ein Grundverständnis und die Werkzeuge an die Hand zu geben, mit denen Sie Programme zum *Lernen aus Daten* entwickeln können.

Wir werden eine Vielzahl von Techniken besprechen, von den einfachsten und am häufigsten eingesetzten (wie der linearen Regression) bis zu einigen Deep-Learning-Verfahren, die regelmäßig Wettbewerbe gewinnen.

Anstatt eigene Übungsversionen jedes Algorithmus zu entwickeln, werden wir dazu für den Produktionsbetrieb geschaffene Python-Frameworks verwenden:

- Scikit-Learn (<http://scikit-learn.org/>) ist sehr einfach zu verwenden, enthält aber effiziente Implementierungen vieler Machine-Learning-Algorithmen. Damit ist es ein großartiger Ausgangspunkt, um Machine Learning zu erlernen.
- TensorFlow (<http://tensorflow.org/>) ist eine komplexere Bibliothek für verteiltes Rechnen. Mit ihr können Sie sehr große neuronale Netze effizient trainieren und ausführen, indem Sie die Berechnungen auf bis zu Hunderte von Servern mit mehreren GPUs (*Graphics Processing Units*) verlagern. TensorFlow (TF) wurde von Google entwickelt und läuft in vielen großflächigen Machine-Learning-Anwendungen. Die Bibliothek wurde im November 2015 als Open Source veröffentlicht.

- Keras (<https://keras.io/>) ist eine High-Level-Deep-Learning-API, die das Trainieren und Ausführen neuronaler Netze sehr einfach macht. Sie kann auf TensorFlow, Theano oder Microsoft Cognitive Toolkit (früher bekannt als CNTK) aufsetzen. TensorFlow bringt seine eigene Implementierung dieser API namens *tf.keras* mit, die einige der fortgeschritteneren TensorFlow-Features unterstützt (zum Beispiel die Möglichkeit, Daten effizient zu laden).

Dieses Buch verfolgt einen praxisorientierten Ansatz, bei dem Sie ein intuitives Verständnis von Machine Learning entwickeln, indem Sie sich mit konkreten Beispielen und ein klein wenig Theorie beschäftigen. Auch wenn Sie dieses Buch lesen können, ohne Ihren Laptop in die Hand zu nehmen, empfehlen wir Ihnen, mit den als Jupyter-Notebooks unter <https://github.com/ageron/handson-ml2> verfügbaren Codebeispielen herumzuxperimentieren.

## Voraussetzungen

Dieses Buch geht davon aus, dass Sie ein wenig Programmiererfahrung mit Python haben und dass Sie mit den wichtigsten wissenschaftlichen Bibliotheken in Python vertraut sind, insbesondere mit NumPy (<http://numpy.org/>), pandas (<http://pandas.pydata.org/>) und Matplotlib (<http://matplotlib.org/>).

Wenn Sie sich dafür interessieren, was hinter den Kulissen passiert, sollten Sie ein Grundverständnis von Oberstufenmathematik haben (Analysis, lineare Algebra, Wahrscheinlichkeiten und Statistik).

Sollten Sie Python noch nicht kennen, ist <http://learnpython.org/> ein ausgezeichnete Ausgangspunkt. Das offizielle Tutorial auf [python.org](https://docs.python.org/3/tutorial/) (<https://docs.python.org/3/tutorial/>) ist ebenfalls recht gut.

Falls Sie Jupyter noch nie verwendet haben, führt Sie Kapitel 2 durch die Installation und die Grundlagen: Es ist ein leistungsfähiges Werkzeug in Ihrem Werkzeugkasten.

Und für den Fall, dass Sie mit den wissenschaftlichen Bibliotheken für Python nicht vertraut sind, beinhalten die mitgelieferten Jupyter-Notebooks einige Tutorials. Es gibt dort auch ein kurzes Mathematiktutorial über lineare Algebra.

## Wegweiser durch dieses Buch

Dieses Buch ist in zwei Teile aufgeteilt. Teil I behandelt folgende Themen:

- Was ist Machine Learning? Welche Aufgaben lassen sich damit lösen? Welches sind die wichtigsten Kategorien und Grundbegriffe von Machine-Learning-Systemen?
- Die Schritte in einem typischen Machine-Learning-Projekt.
- Lernen durch Anpassen eines Modells an Daten.

- Optimieren einer Kostenfunktion.
- Bearbeiten, Säubern und Vorbereiten von Daten.
- Merkmale auswählen und entwickeln.
- Ein Modell auswählen und dessen Hyperparameter über Kreuzvalidierung optimieren.
- Die Herausforderungen beim Machine Learning, insbesondere Underfitting und Overfitting (das Gleichgewicht zwischen Bias und Varianz).
- Die verbreitetsten Lernalgorithmen: lineare und polynomielle Regression, logistische Regression, k-nächste Nachbarn, Support Vector Machines, Entscheidungsbäume, Random Forests und Ensemble-Methoden.
- Dimensionsreduktion der Trainingsdaten, um dem »Fluch der Dimensionalität« etwas entgegenzusetzen.
- Andere Techniken des unüberwachten Lernens, unter anderem Clustering, Dichteabschätzung und Anomalieerkennung.

Teil II widmet sich diesen Themen:

- Was sind neuronale Netze? Wofür sind sie geeignet?
- Erstellen und Trainieren neuronaler Netze mit TensorFlow und Keras.
- Die wichtigsten Architekturen neuronaler Netze: Feed-Forward-Netze für Tabellendaten, Convolutional Neural Networks zur Bilderkennung, rekurrente Netze und Long-Short-Term-Memory-(LSTM-)Netze zur Sequenzverarbeitung, Encoder/Decoder und Transformer für die Sprachverarbeitung, Autoencoder und Generative Adversarial Networks (GANs) zum generativen Lernen.
- Techniken zum Trainieren von Deep-Learning-Netzen.
- Wie man einen Agenten erstellt (zum Beispiel einen Bot in einem Spiel), der durch Versuch und Irrtum gute Strategien erlernt und dabei Reinforcement Learning einsetzt.
- Effizientes Laden und Vorverarbeiten großer Datenmengen.
- Trainieren und Deployen von TensorFlow-Modellen im großen Maßstab.

Der erste Teil baut vor allem auf Scikit-Learn auf, der zweite Teil verwendet TensorFlow.



Springen Sie nicht zu schnell ins tiefe Wasser: Auch wenn Deep Learning zweifelsohne eines der aufregendsten Teilgebiete des Machine Learning ist, sollten Sie zuerst Erfahrungen mit den Grundlagen sammeln. Außerdem lassen sich die meisten Aufgabenstellungen recht gut mit einfacheren Techniken wie Random Forests und Ensemble-Methoden lösen (die in Teil I besprochen werden). Deep Learning ist am besten für komplexe Aufgaben wie Bilderkennung, Spracherkennung und Sprachverarbeitung geeignet, vorausgesetzt, Sie haben genug Daten und Geduld.

# Änderungen in der zweiten Auflage

Diese zweite Auflage hat sechs zentrale Ziele:

1. Die Behandlung zusätzlicher ML-Themen: weitere Techniken zum unüberwachten Lernen (unter anderem Clustering, Anomalieerkennung, Dichteabschätzung und Mischmodelle), zusätzliche Techniken zum Trainieren von Deep Networks (einschließlich sich selbst normalisierender Netze), weitere Techniken der Bilderkennung (unter anderem Xception, SNet, Objekterkennung mit YOLO und semantische Segmentierung mit R-CNN), das Verarbeiten von Sequenzen mit Convolutional Neural Networks (CNNs, einschließlich WaveNet), Verarbeitung natürlicher Sprache mit rekurrenten neuronalen Netzen (RNNs), CNNs und Transformers, GANs.
2. Zusätzliche Bibliotheken und APIs (Keras, die Data-API, TF-Agents für das Reinforcement Learning) sowie das Trainieren und Deployen von TF-Modellen im großen Maßstab mithilfe der Distribution Strategies API, TF Serving und Google Cloud AI Platform; zudem eine kurze Vorstellung von TF Transform, TFLite, TF Addons/Seq2Seq und TensorFlow.js.
3. Vorstellen einiger der neuesten Forschungsergebnisse aus dem Deep Learning.
4. Migrieren aller TensorFlow-Kapitel nach TensorFlow 2 und Verwenden der TensorFlow-Implementierung der Keras-API (tf.keras), wann immer das möglich ist.
5. Aktualisieren der Codebeispiele auf die neuesten Versionen von Scikit-Learn, NumPy, pandas, Matplotlib und anderer Bibliotheken.
6. Anpassen einiger Abschnitte zum besseren Verständnis und Beheben von Fehlern dank sehr vieler Rückmeldungen von Lesern.

Es wurden ein paar Kapitel hinzugefügt, andere wurden umgeschrieben, und ein paar wurden neu angeordnet. Unter <https://homl.info/changes2> finden Sie detailliertere Angaben darüber, was sich in der zweiten Auflage geändert hat.

## Ressourcen im Netz

Es gibt viele ausgezeichnete Ressourcen, mit deren Hilfe sich Machine Learning erlernen lässt. Der ML-Kurs auf Coursera (<https://homl.info/ngcourse>) von Andrew Ng ist faszinierend, auch wenn er einen beträchtlichen Zeitaufwand bedeutet (in Monaten).

Darüber hinaus finden Sie viele interessante Webseiten über Machine Learning, darunter natürlich den ausgezeichneten User Guide (<https://homl.info/skdoc>) von Scikit-Learn. Auch Dataquest (<https://www.dataquest.io/>), das sehr ansprechende Tutorials und ML-Blogs bietet, sowie die auf Quora (<https://homl.info/1>) aufgeführ-

ten ML-Blogs könnten Ihnen gefallen. Schließlich sind auf der Deep-Learning-Website (<http://deeplearning.net/>) Ressourcen aufgezählt, mit denen Sie mehr lernen können.

Natürlich bieten auch viele andere Bücher eine Einführung in Machine Learning, insbesondere:

- Joel Grus, *Einführung in Data Science: Grundprinzipien der Datenanalyse mit Python* (<https://www.oreilly.de/buecher/13335/9783960091233-einf%C3%BChrung-in-data-science.html>) (O'Reilly). Dieses Buch stellt die Grundlagen von Machine Learning vor und implementiert die wichtigsten Algorithmen in reinem Python (von null auf).
- Stephen Marsland, *Machine Learning: An Algorithmic Perspective* (Chapman & Hall). Dieses Buch ist eine großartige Einführung in Machine Learning, die viele Themen ausführlich behandelt. Es enthält Codebeispiele in Python (ebenfalls von null auf, aber mit NumPy).
- Sebastian Raschka, *Machine Learning mit Python und Scikit-Learn und TensorFlow: Das umfassende Praxis-Handbuch für Data Science, Predictive Analytics und Deep Learning* (mitp Professional). Eine weitere ausgezeichnete Einführung in Machine Learning. Dieses Buch konzentriert sich auf Open-Source-Bibliotheken in Python (Pylearn 2 und Theano).
- François Chollet, *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek* (mitp Professional). Ein sehr praxisnahes Buch, das klar und präzise viele Themen behandelt – wie Sie es vom Autor der ausgezeichneten Keras-Bibliothek erwarten können. Es zieht Codebeispiele der mathematischen Theorie vor.
- Andriy Burkov, *Machine Learning kompakt: Alles, was Sie wissen müssen* (mitp Professional). Dieses sehr kurze Buch behandelt ein beeindruckendes Themenspektrum gut verständlich, scheut dabei aber nicht vor mathematischen Gleichungen zurück.
- Yaser S. Abu-Mostafa, Malik Magdon-Ismail und Hsuan-Tien Lin, *Learning from Data* (AMLBook). Als eher theoretische Abhandlung von ML enthält dieses Buch sehr tiefgehende Erkenntnisse, insbesondere zum Gleichgewicht zwischen Bias und Varianz (siehe Kapitel 4).
- Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach, 3rd Edition* (Pearson). Dieses ausgezeichnete (und umfangreiche) Buch deckt eine unglaubliche Stoffmenge ab, darunter Machine Learning. Es hilft dabei, ML in einem breiteren Kontext zu betrachten.

Eine gute Möglichkeit zum Lernen sind schließlich Webseiten mit ML-Wettbewerben wie Kaggle.com (<https://www.kaggle.com/>). Dort können Sie Ihre Fähigkeiten an echten Aufgaben üben und Hilfe und Tipps von einigen der besten ML-Profis erhalten.

# In diesem Buch verwendete Konventionen

Die folgenden typografischen Konventionen werden in diesem Buch verwendet:

## *Kursiv*

Kennzeichnet neue Begriffe, URLs, E-Mail-Adressen, Dateinamen und Dateierendungen.

## Konstante Zeichenbreite

Wird für Programmlistings und für Programmelemente in Textabschnitten wie Namen von Variablen und Funktionen, Datenbanken, Datentypen, Umgebungsvariablen, Anweisungen und Schlüsselwörter verwendet.

## **Konstante Zeichenbreite, fett**

Kennzeichnet Befehle oder anderen Text, den der Nutzer wörtlich eingeben sollte.

## *Konstante Zeichenbreite, kursiv*

Kennzeichnet Text, den der Nutzer je nach Kontext durch entsprechende Werte ersetzen sollte.



Dieses Symbol steht für einen Tipp oder eine Empfehlung.



Dieses Symbol steht für einen allgemeinen Hinweis.



Dieses Symbol steht für eine Warnung oder erhöhte Aufmerksamkeit.

## Codebeispiele

Es gibt eine Reihe von Jupyter-Notebooks voll mit zusätzlichem Material, wie Codebeispielen und Übungen, die zum Herunterladen unter <https://github.com/ageron/handson-ml2> bereitstehen.

Einige der Codebeispiele im Buch lassen sich wiederholende Abschnitte oder Details weg, die offensichtlich sind oder nichts mit Machine Learning zu tun haben. Das sorgt dafür, dass Sie sich auf die wichtigen Teile des Codes konzentrieren können, und spart Platz, um mehr Themen unterzubringen. Wollen Sie sich die vollständigen Codebeispiele betrachten, finden Sie diese in den Jupyter-Notebooks.

Geben die Codebeispiele etwas aus, wird dies mit Python-Prompts (>>> und ...) wie in einer Python-Shell dargestellt, um den Code deutlich von den Ausgaben trennen zu können. So definiert beispielsweise folgender Code die Funktion `square()`, rechnet dann damit und gibt das Quadrat von 3 aus:

```
>>> def square(x):
...     return x ** 2
...
>>> result = square(3)
>>> result
9
```

Gibt Code nichts aus, werden keine Prompts verwendet. Aber das Ergebnis wird manchmal als Kommentar angegeben, wie zum Beispiel hier:

```
def square(x):
    return x ** 2

result = square(3) # Ergebnis ist 9
```

## Verwenden von Codebeispielen

Dieses Buch ist dazu da, Ihnen beim Erledigen Ihrer Arbeit zu helfen. Im Allgemeinen dürfen Sie die Codebeispiele aus diesem Buch in Ihren eigenen Programmen und der dazugehörigen Dokumentation verwenden. Sie müssen uns dazu nicht um Erlaubnis fragen, solange Sie nicht einen beträchtlichen Teil des Codes reproduzieren. Beispielsweise benötigen Sie keine Erlaubnis, um ein Programm zu schreiben, in dem mehrere Codefragmente aus diesem Buch vorkommen. Wollen Sie dagegen eine CD-ROM mit Beispielen aus Büchern von O'Reilly verkaufen oder verteilen, benötigen Sie eine Erlaubnis. Eine Frage zu beantworten, indem Sie aus diesem Buch zitieren und ein Codebeispiel wiedergeben, benötigt keine Erlaubnis. Eine beträchtliche Menge Beispielcode aus diesem Buch in die Dokumentation Ihres Produkts aufzunehmen, bedarf hingegen einer Erlaubnis.

Wir freuen uns über Zitate, verlangen diese aber nicht. Ein Zitat enthält Titel, Autor, Verlag und ISBN. Beispiel: »*Praxiseinstieg Machine Learning mit Scikit-Learn, Keras und TensorFlow* von Aurélien Géron (O'Reilly). Copyright 2020, ISBN 978-3-96009-124-0.«

Wenn Sie glauben, dass Ihre Verwendung von Codebeispielen über die übliche Nutzung hinausgeht oder außerhalb der oben vorgestellten Nutzungsbedingungen liegt, kontaktieren Sie uns bitte unter [komentar@oreilly.de](mailto:komentar@oreilly.de).

## Danksagungen

In meinen wildesten Träumen hätte ich mir niemals vorgestellt, dass die erste Auflage dieses Buchs solch eine Verbreitung finden würde. Ich habe so viele Nachrich-

ten von Lesern erhalten – oft mit Fragen, manche mit freundlichen Hinweisen auf Fehler und die meisten mit ermutigenden Worten. Ich kann gar nicht sagen, wie dankbar ich all diesen Lesern für ihre Unterstützung bin. Vielen, vielen Dank! Scheuen Sie sich nicht, sich auf GitHub (<https://homl.info/issues2>) zu melden, wenn Sie Fehler in den Codebeispielen finden (oder einfach etwas fragen wollen) oder um auf Fehler im Text aufmerksam (<https://homl.info/errata2>) zu machen. Manche Leser haben mir auch geschrieben, wie dieses Buch ihnen dabei geholfen hat, ihren ersten Job zu bekommen oder ein konkretes Problem zu lösen, an dem sie gearbeitet haben. Ich finde ein solches Feedback unglaublich motivierend. Hat Ihnen dieses Buch geholfen, würde ich mich freuen, wenn Sie mir Ihre Geschichte erzählen würden – entweder privat (zum Beispiel über LinkedIn (<https://www.linkedin.com/in/aurelien-geron/>)) oder öffentlich (beispielsweise in einem Tweet oder über ein Amazon-Review (<https://homl.info/amazon2>)).

Ich bin all den fantastischen Menschen unglaublich dankbar, die in ihrem geschäftigen Leben die Zeit gefunden haben, mein Buch im Detail gegenzulesen. Insbesondere möchte ich François Chollet dafür danken, dass er alle Kapitel zu Keras und TensorFlow kontrolliert und mir großartiges und detailliertes Feedback gegeben hat. Da Keras eine meiner wichtigsten Ergänzungen dieser zweiten Auflage ist, war die Review durch den Autor unbezahlbar. Ich empfehle Ihnen François' Buch *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek* (mitp Professional): Es bietet die Präzision, Klarheit und Tiefe, die auch die Keras-Bibliothek selbst besitzt. Besonderer Dank geht ebenfalls an Ankur Patel, der jedes Kapitel dieser zweiten Auflage begutachtet und mir ausgezeichnetes Feedback gegeben hat, insbesondere zu Kapitel 9, das sich um unüberwachtes Lernen dreht. Er könnte glatt ein ganzes Buch zu dem Thema schreiben ... Moment mal, das hat er ja! Schauen Sie sich mal *Praxisbuch Unsupervised Learning: Machine-Learning-Anwendungen für ungelabelte Daten mit Python programmieren* (<https://www.oreilly.de/buecher/13534/9783960091271-praxisbuch-unsupervised-learning.html>) (O'Reilly) an. Ein großes Dankeschön auch an Olzhas Akpambetov, der alle Kapitel im zweiten Teil des Buchs begutachtet, sehr viel Code getestet und viele großartige Verbesserungsvorschläge gemacht hat. Ich bin dankbar, dass Mark Daoust, Jon Krohn, Dominic Monn und Josh Patterson den zweiten Teil des Buchs so genau begutachtet und ihre Erfahrungen eingebracht haben. Sie ließen keinen Stein auf dem anderen und lieferten ausgezeichnete Hinweise.

Beim Schreiben dieser zweiten Auflage hatte ich das Glück, sehr viel Hilfe von Mitgliedern des TensorFlow-Teams zu bekommen, insbesondere von Martin Wicke, der unermüdlich Dutzende meiner Fragen beantwortet und den Rest an die richtigen Leute weitergeleitet hat, unter anderen an Karmel Allison, Paige Bailey, Eugene Brevdo, William Chargin, Daniel »Wolff« Dobson, Nick Felt, Bruce Fontaine, Goldie Gadde, Sandeep Gupta, Priya Gupta, Kevin Haas, Konstantinos Katsiapis, Viacheslav Kovalevskyi, Allen Lavoie, Clemens Mewald, Dan Moldovan, Sean Morgan, Tom O'Malley, Alexandre Passos, André Susano Pinto, Anthony Plataniotis, Oscar Ramirez, Anna Revinskaya, Saurabh Saxena, Ryan Sepassi, Jiri Simsa,

Xiaodan Song, Christina Sorokin, Dustin Tran, Todd Wang, Pete Warden (der auch die erste Auflage begutachtet hat), Edd Wilder-James und Yuefeng Zhou, die alle außerordentlich hilfreich waren. Ein großer Dank an euch alle und auch an alle anderen Mitglieder des TensorFlow-Teams – nicht nur für eure Hilfe, sondern auch dafür, dass ihr solch eine tolle Bibliothek geschaffen habt. Ein besonderer Dank geht an Irene Giannoumis und Robert Crowe vom TFCX-Team, die die Kapitel 13 und 19 im Detail durchgearbeitet haben.

Ich danke auch den fantastischen Menschen bei O'Reilly, insbesondere Nicole Taché für ihr aufschlussreiches, immer freundliches, ermutigendes und hilfreiches Feedback. Eine bessere Lektorin hätte ich mir nicht vorstellen können. Ein großer Dank geht an Michele Cronin, die zu Beginn dieser zweiten Auflage sehr hilfreich (und geduldig) war, und an Kristen Brown, Production Editor für die zweite Auflage, die sie auf allen Schritten begleitet hat (sie hat auch Korrekturen und Aktualisierungen jedes Nachdrucks der ersten Auflage koordiniert). Ich danke Rachel Monaghan und Amanda Kersey für ihr umfassendes Copyediting (der ersten bzw. zweiten Auflage) und Johnny O'Toole, der die Beziehung zu Amazon gemanagt und viele meiner Fragen beantwortet hat. Dank geht an Marie Beaugureau, Ben Lorica, Mike Loukides und Laurel Ruma dafür, dass sie an dieses Projekt geglaubt und mir geholfen haben, den Rahmen abzustecken. Ich danke Matt Hacker und dem gesamten Atlas-Team für das Beantworten aller meiner technischen Fragen zu Formatierung, AsciiDoc und LaTeX sowie Nick Adams, Rebecca Demarest, Rachel Head, Judith McConville, Helen Monroe, Karen Montgomery, Rachel Roumeliotis und allen bei O'Reilly, die zu diesem Buch beigetragen haben.

Ich möchte auch meinen früheren Kollegen bei Google danken, insbesondere dem Team zur Klassifikation von YouTube-Videos, von denen ich sehr viel über Machine Learning gelernt habe. Ohne sie hätte ich die erste Auflage niemals starten können. Besonderer Dank gebührt meinen persönlichen ML-Gurus: Clément Courbet, Julien Dubois, Mathias Kende, Daniel Kitachewsky, James Pack, Alexander Pak, Anosh Raj, Vitor Sessak, Wiktor Tomczak, Ingrid von Glehn und Rich Washington. Und danke an alle anderen, mit denen ich bei YouTube und in den großartigen Google-Forschungsteams in Mountain View zusammengearbeitet habe. Vielen Dank auch an Martin Andrews, Sam Witteveen und Jason Zaman, dass sie mich in ihre Google-Developer-Experts-Gruppe in Singapur aufgenommen haben – mit freundlicher Unterstützung durch Soonson Kwon – und für all die tollen Diskussionen über Deep Learning und TensorFlow. Jeder, der in Singapur an Deep Learning interessiert ist, sollte auf jeden Fall das Deep Learning Singapore Meetup (<https://homl.info/meetupsg>) besuchen. Jason verdient besonderen Dank für sein TFLite-Know-how, das in Kapitel 19 eingeflossen ist!

Nie werde ich all die netten Leute vergessen, die die erste Auflage dieses Buchs Korrektur gelesen haben, unter anderem David Andrzejewski, Lukas Biewald, Justin Francis, Vincent Guillbeau, Eddy Hung, Karim Matrah, Grégoire Mesnil, Salim Sémaoune, Iain Smears, Michel Tessier, Ingrid von Glehn, Pete Warden und natür-

lich mein lieber Bruder Sylvain. Ein besonderer Dank geht an Haesun Park, der mir sehr viel ausgezeichnetes Feedback gab und viele Fehler fand, während er die koreanische Übersetzung der ersten Auflage schrieb. Er hat auch die Jupyter-Notebooks ins Koreanische übersetzt, nicht zu vergessen die Dokumentation von TensorFlow. Ich spreche kein Koreanisch, aber in Anbetracht der Qualität seines Feedbacks müssen alle seine Übersetzungen wirklich ausgezeichnet sein. Darüber hinaus hat Haesun freundlicherweise ein paar der Lösungen zu den Übungen in dieser zweiten Auflage beigetragen.

Schließlich bin ich meiner geliebten Frau Emmanuelle und unseren drei wunderbaren Kindern Alexandre, Rémi und Gabrielle unendlich dafür dankbar, dass sie mich zur Arbeit an diesem Buch ermutigt haben. Auch danke ich ihnen für ihre unersättliche Neugier: Indem ich einige der schwierigsten Konzepte in diesem Buch meiner Frau und meinen Kindern erklärt habe, konnte ich meine Gedanken ordnen und viele Teile direkt verbessern. Und sie haben mir sogar Kekse und Kaffee vorbeigebracht. Was kann man sich noch mehr wünschen?



# Die Grundlagen des Machine Learning

