

Inhaltsverzeichnis

	Über die Autoren	17
	Über die Korrektoren	19
	Über den Fachkorrektor der deutschen Ausgabe	20
	Einleitung	21
	Einstieg in Machine Learning	21
	Zum Inhalt des Buches	23
	Was Sie benötigen	26
	Codebeispiele herunterladen	26
	Konventionen im Buch	26
1	Wie Computer aus Daten lernen können	29
1.1	Intelligente Maschinen, die Daten in Wissen verwandeln	29
1.2	Die drei Arten des Machine Learnings	30
1.2.1	Mit überwachtem Lernen Vorhersagen treffen	31
1.2.2	Interaktive Aufgaben durch Reinforcement Learning lösen	34
1.2.3	Durch unüberwachtes Lernen verborgene Strukturen erkennen	35
1.3	Grundlegende Terminologie und Notation	36
1.3.1	Im Buch verwendete Notation und Konventionen	37
1.3.2	Terminologie	38
1.4	Entwicklung eines Systems für das Machine Learning	39
1.4.1	Vorverarbeitung: Daten in Form bringen	40
1.4.2	Trainieren und Auswählen eines Vorhersagemodells	40
1.4.3	Bewertung von Modellen und Vorhersage anhand unbekannter Dateninstanzen	41
1.5	Machine Learning mit Python	42
1.5.1	Python und Python-Pakete installieren	42
1.5.2	Verwendung der Python-Distribution Anaconda	43
1.5.3	Pakete für wissenschaftliches Rechnen, Data Science und Machine Learning	43
1.6	Zusammenfassung	44

2	Lernalgorithmen für die Klassifikation trainieren.	45
2.1	Künstliche Neuronen: Ein kurzer Blick auf die Anfänge des Machine Learnings	45
2.1.1	Formale Definition eines künstlichen Neurons	46
2.1.2	Die Perzeptron-Lernregel.	48
2.2	Implementierung eines Perzeptron-Lernalgorithmus in Python . . .	51
2.2.1	Eine objektorientierte Perzeptron-API	51
2.2.2	Trainieren eines Perzeptron-Modells mit der Iris-Datensammlung	55
2.3	Adaptive lineare Neuronen und die Konvergenz des Lernens	61
2.3.1	Straffunktionen mit dem Gradientenabstiegsverfahren minimieren	62
2.3.2	Implementierung eines adaptiven linearen Neurons in Python	64
2.3.3	Verbesserung des Gradientenabstiegsverfahrens durch Merkmalstandardisierung	69
2.3.4	Large Scale Machine Learning und stochastisches Gradientenabstiegsverfahren.	71
2.4	Zusammenfassung	77
3	Machine-Learning-Klassifikatoren mit scikit-learn verwenden	79
3.1	Auswahl eines Klassifikationsalgorithmus	79
3.2	Erste Schritte mit scikit-learn: Trainieren eines Perzeptrons.	80
3.3	Klassenwahrscheinlichkeiten durch logistische Regression modellieren	86
3.3.1	Logistische Regression und bedingte Wahrscheinlichkeiten.	87
3.3.2	Gewichte der logistischen Straffunktion ermitteln	91
3.3.3	Konvertieren einer Adaline-Implementierung in einen Algorithmus für eine logistische Regression	93
3.3.4	Trainieren eines logistischen Regressionsmodells mit scikit-learn.	98
3.3.5	Überanpassung durch Regularisierung verhindern	101
3.4	Maximum-Margin-Klassifikation mit Support Vector Machines.	104
3.4.1	Maximierung des Randbereichs	105
3.4.2	Handhabung des nicht linear trennbaren Falls mit Schlupfvariablen.	106
3.4.3	Alternative Implementierungen in scikit-learn	108

3.5	Nichtlineare Aufgaben mit einer Kernel-SVM lösen	109
3.5.1	Kernel-Methoden für linear nicht trennbare Daten.	109
3.5.2	Mit dem Kernel-Trick Hyperebenen in höherdimensionalen Räumen finden.	111
3.6	Lernen mit Entscheidungsbäumen	115
3.6.1	Maximierung des Informationsgewinns: Daten ausreizen	116
3.6.2	Konstruktion eines Entscheidungsbaums	120
3.6.3	Mehrere Entscheidungsbäume zu einem Random Forest kombinieren	124
3.7	k-Nearest-Neighbors: Ein Lazy-Learning-Algorithmus.	127
3.8	Zusammenfassung	130
4	Gut geeignete Trainingsdatenmengen: Datenvorverarbeitung	133
4.1	Umgang mit fehlenden Daten	133
4.1.1	Fehlende Werte in Tabellendaten	134
4.1.2	Instanzen oder Merkmale mit fehlenden Daten entfernen.	135
4.1.3	Fehlende Werte ergänzen	136
4.1.4	Die Schätzer-API von scikit-learn	137
4.2	Handhabung kategorialer Daten	138
4.2.1	Codierung kategorialer Daten mit pandas	139
4.2.2	Zuweisung von ordinalen Merkmalen	139
4.2.3	Codierung der Klassenbezeichnungen.	140
4.2.4	One-hot-Codierung der nominalen Merkmale.	141
4.3	Aufteilung einer Datensammlung in Trainings- und Testdaten	145
4.4	Anpassung der Merkmale.	148
4.5	Auswahl aussagekräftiger Merkmale.	150
4.5.1	L1- und L2-Regularisierung als Straffunktionen	151
4.5.2	Geometrische Interpretation der L2-Regularisierung.	151
4.5.3	Dünn besetzte Lösungen mit L1-Regularisierung	153
4.5.4	Algorithmen zur sequenziellen Auswahl von Merkmalen	157
4.6	Beurteilung der Bedeutung von Merkmalen mit Random Forests.	164
4.7	Zusammenfassung	167
5	Datenkomprimierung durch Dimensionsreduktion	169
5.1	Unüberwachte Dimensionsreduktion durch	

	Hauptkomponentenanalyse	169
5.1.1	Schritte bei der Hauptkomponentenanalyse	170
5.1.2	Schrittweise Extraktion der Hauptkomponenten	171
5.1.3	Totale Varianz und erklärte Varianz	174
5.1.4	Merkmalstransformation	176
5.1.5	Hauptkomponentenanalyse mit scikit-learn	179
5.2	Überwachte Datenkomprimierung durch lineare Diskriminanzanalyse	183
5.2.1	Hauptkomponentenanalyse kontra lineare Diskriminanzanalyse	183
5.2.2	Die interne Funktionsweise der linearen Diskriminanzanalyse	184
5.2.3	Berechnung der Streumatrizen	185
5.2.4	Auswahl linearer Diskriminanten für den neuen Merkmalsunterraum	187
5.2.5	Projektion in den neuen Merkmalsraum	190
5.2.6	LDA mit scikit-learn	191
5.3	Kernel-Hauptkomponentenanalyse für nichtlineare Zuordnungen verwenden	193
5.3.1	Kernel-Funktionen und der Kernel-Trick	194
5.3.2	Implementierung einer Kernel-Hauptkomponentenanalyse in Python	198
5.3.3	Projizieren neuer Datenpunkte	206
5.3.4	Kernel-Hauptkomponentenanalyse mit scikit-learn	210
5.4	Zusammenfassung	211
6	Bewährte Verfahren zur Modellbewertung und Hyperparameter-Optimierung	213
6.1	Arbeitsabläufe mit Pipelines optimieren	213
6.1.1	Die Wisconsin-Brustkrebs-Datensammlung	213
6.1.2	Transformer und Schätzer in einer Pipeline kombinieren	215
6.2	Beurteilung des Modells durch k-fache Kreuzvalidierung	217
6.2.1	Holdout-Methode	218
6.2.2	k-fache Kreuzvalidierung	219
6.3	Algorithmen mit Lern- und Validierungskurven debuggen	223
6.3.1	Probleme mit Bias und Varianz anhand von Lernkurven erkennen	224

6.3.2	Überanpassung und Unteranpassung anhand von Validierungskurven erkennen	227
6.4	Feinabstimmung eines Lernmodells durch Grid Search	229
6.4.1	Optimierung der Hyperparameter durch Grid Search	230
6.4.2	Algorithmenauswahl durch verschachtelte Kreuzvalidierung	232
6.5	Verschiedene Kriterien zur Leistungsbewertung	234
6.5.1	Interpretation einer Verwechslungsmatrix.	234
6.5.2	Optimierung der Genauigkeit und der Trefferquote eines Klassifikationsmodells	236
6.5.3	Receiver-Operating-Characteristic-Diagramme	238
6.5.4	Bewertungskriterien für Mehrklassen-Klassifikationen	241
6.6	Handhabung unausgewogener Klassenverteilung	242
6.7	Zusammenfassung	245
7	Kombination verschiedener Modelle für das Ensemble Learning. . .	247
7.1	Ensemble Learning	247
7.2	Klassifikatoren durch Mehrheitsentscheidung kombinieren.	251
7.2.1	Implementierung eines einfachen Mehrheitsentscheidungs-Klassifikators	251
7.2.2	Vorhersagen nach dem Mehrheitsentscheidungsprinzip treffen	258
7.3	Bewertung und Abstimmung des Klassifikator-Ensembles.	261
7.4	Bagging: Klassifikator-Ensembles anhand von Bootstrap-Stichproben entwickeln.	268
7.4.1	Bagging kurz zusammengefasst	269
7.4.2	Klassifikation der Wein-Datensammlung durch Bagging.	270
7.5	Schwache Klassifikatoren durch adaptives Boosting verbessern	274
7.5.1	Funktionsweise des Boostings	274
7.5.2	AdaBoost mit scikit-learn anwenden	278
7.6	Zusammenfassung	282
8	Machine Learning zur Analyse von Stimmungslagen nutzen.	283
8.1	Die IMDb-Filmdatenbank.	283
8.1.1	Herunterladen der Datensammlung	284
8.1.2	Vorverarbeiten der Filmbewertungsdaten	284
8.2	Das Bag-of-words-Modell	286
8.2.1	Wörter in Merkmalsvektoren umwandeln	287
8.2.2	Beurteilung der Wortrelevanz durch das Tf-idf-Maß	289

8.2.3	Textdaten bereinigen	291
8.2.4	Dokumente in Tokens zerlegen.	293
8.3	Ein logistisches Regressionsmodell für die Dokumentklassifikation trainieren	295
8.4	Verarbeitung großer Datenmengen: Online-Algorithmen und Out-of-Core Learning.	298
8.5	Topic Modeling mit latenter Dirichlet-Allokation	302
8.5.1	Aufteilung von Texten mit der LDA	303
8.5.2	LDA mit scikit-learn	303
8.6	Zusammenfassung	307
9	Einbettung eines Machine-Learning-Modells in eine Webanwendung	309
9.1	Serialisierung angepasster Schätzer mit scikit-learn.	309
9.2	Einrichtung einer SQLite-Datenbank zum Speichern von Daten . . .	313
9.3	Entwicklung einer Webanwendung mit Flask.	315
9.3.1	Die erste Webanwendung mit Flask	316
9.3.2	Formularvalidierung und -ausgabe	318
9.4	Der Filmbewertungsklassifikator als Webanwendung	324
9.4.1	Dateien und Ordner – die Verzeichnisstruktur	326
9.4.2	Implementierung der Hauptanwendung app.py	326
9.4.3	Einrichtung des Bewertungsformulars.	329
9.4.4	Eine Vorlage für die Ergebnisseite erstellen.	330
9.5	Einrichtung der Webanwendung auf einem öffentlich zugänglichen Webserver	333
9.5.1	Erstellen eines Benutzerkontos bei PythonAnywhere	333
9.5.2	Hochladen der Filmbewertungsanwendung	334
9.5.3	Updates des Filmbewertungsklassifikators	335
9.6	Zusammenfassung	338
10	Vorhersage stetiger Zielvariablen durch Regressionsanalyse.	339
10.1	Lineare Regression.	339
10.1.1	Ein einfaches lineares Regressionsmodell	340
10.1.2	Multiple lineare Regression	341
10.2	Die Boston-Housing-Datensammlung.	342
10.2.1	Einlesen der Datenmenge in einen DataFrame	342
10.2.2	Visualisierung der wichtigen Eigenschaften einer Datenmenge	344

10.2.3	Zusammenhänge anhand der Korrelationsmatrix erkennen	346
10.3	Implementierung eines linearen Regressionsmodells mit der Methode der kleinsten Quadrate	348
10.3.1	Berechnung der Regressionsparameter mit dem Gradientenabstiegsverfahren.	349
10.3.2	Schätzung der Koeffizienten eines Regressionsmodells mit scikit-learn	353
10.4	Anpassung eines robusten Regressionsmodells mit dem RANSAC-Algorithmus	355
10.5	Bewertung der Leistung linearer Regressionsmodelle	358
10.6	Regularisierungsverfahren für die Regression einsetzen.	361
10.7	Polynomiale Regression: Umwandeln einer linearen Regression in eine Kurve	363
10.7.1	Hinzufügen polynomialer Terme mit scikit-learn.	364
10.7.2	Modellierung nichtlinearer Zusammenhänge in der Boston-Housing-Datensammlung	366
10.8	Handhabung nichtlinearer Beziehungen mit Random Forests.	369
10.8.1	Entscheidungsbaum-Regression.	370
10.8.2	Random-Forest-Regression	371
10.9	Zusammenfassung	375
11	Verwendung von Daten ohne Label: Clusteranalyse.	377
11.1	Gruppierung von Objekten nach Ähnlichkeit mit dem k-Means-Algorithmus	377
11.1.1	k-Means-Clustering mit scikit-learn	378
11.1.2	Der k-Means++-Algorithmus.	383
11.1.3	»Crisp« und »soft« Clustering.	384
11.1.4	Die optimale Anzahl der Cluster mit dem Ellenbogenkriterium ermitteln	386
11.1.5	Quantifizierung der Clustering-Güte mit Silhouettendiagrammen	388
11.2	Cluster als hierarchischen Baum organisieren	393
11.2.1	Gruppierung von Clustern	393
11.2.2	Hierarchisches Clustering mittels einer Distanzmatrix	395
11.2.3	Dendrogramme und Heatmaps verknüpfen	399
11.2.4	Agglomeratives Clustering mit scikit-learn	401
11.3	Bereiche hoher Dichte mit DBSCAN ermitteln	402
11.4	Zusammenfassung	407

12	Implementierung eines künstlichen neuronalen Netzes	409
12.1	Modellierung komplexer Funktionen mit künstlichen neuronalen Netzen	409
12.1.1	Einschichtige neuronale Netze	411
12.1.2	Mehrschichtige neuronale Netzarchitektur	413
12.1.3	Aktivierung eines neuronalen Netzes durch Vorwärtspropagation	416
12.2	Klassifikation handgeschriebener Ziffern	418
12.2.1	Die MNIST-Datensammlung.	419
12.2.2	Implementierung eines mehrschichtigen Perzeptrons.	426
12.3	Trainieren eines künstlichen neuronalen Netzes	438
12.3.1	Berechnung der logistischen Straffunktion	439
12.3.2	Ein Gespür für die Backpropagation entwickeln	441
12.3.3	Trainieren neuronaler Netze durch Backpropagation	443
12.4	Konvergenz in neuronalen Netzen.	447
12.5	Abschließende Bemerkungen zur Implementierung neuronaler Netze	448
12.6	Zusammenfassung	448
13	Parallelisierung des Trainings neuronaler Netze mit TensorFlow . .	451
13.1	TensorFlow und Trainingsleistung	451
13.1.1	Herausforderungen	451
13.1.2	Was genau ist TensorFlow?	453
13.1.3	TensorFlow erlernen	454
13.2	Erste Schritte mit TensorFlow	455
13.2.1	TensorFlow installieren	455
13.2.2	Tensoren in TensorFlow erstellen.	456
13.2.3	Datentyp und Format eines Tensors ändern	457
13.2.4	Anwendung mathematischer Operationen auf Tensoren.	458
13.2.5	Tensoren aufteilen, stapeln und verknüpfen	460
13.3	Eingabe-Pipelines mit tf.data erstellen – die Dataset-API von TensorFlow.	462
13.3.1	Ein TensorFlow-Dataset anhand vorhandener Tensoren erstellen	462
13.3.2	Zwei Tensoren zu einer Datenmenge vereinen.	463
13.3.3	Durchmischen, Batch erstellen und wiederholen	465
13.3.4	Erstellen einer Datenmenge anhand lokal gespeicherter Dateien	468
13.3.5	Zugriff auf die Datenmengen der tensorflow_datasets-Bibliothek	472

13.4	Entwicklung eines NN-Modells mit TensorFlow	478
13.4.1	Die Keras-API (tf.keras) von TensorFlow	478
13.4.2	Entwicklung eines linearen Regressionsmodells	479
13.4.3	Trainieren des Modells mit den Methoden .compile() und .fit()	484
13.4.4	Entwicklung eines mehrschichtigen Perzeptrons zur Klassifikation der Iris-Datensammlung	485
13.4.5	Bewertung des trainierten Modells mit der Testdatenmenge	490
13.4.6	Das trainierte Modell speichern und einlesen	490
13.5	Auswahl der Aktivierungsfunktionen mehrschichtiger neuronaler Netze	491
13.5.1	Die logistische Funktion kurz zusammengefasst	492
13.5.2	Wahrscheinlichkeiten bei der Mehrklassen-Klassifikation mit der softmax-Funktion schätzen	494
13.5.3	Verbreiterung des Ausgabespektrums mittels Tangens hyperbolicus	495
13.5.4	Aktivierung mittels ReLU	498
13.6	Zusammenfassung	499
14	Die Funktionsweise von TensorFlow im Detail	501
14.1	Grundlegende Merkmale von TensorFlow	502
14.2	TensorFlows Berechnungsgraphen: Migration nach TensorFlow v2	503
14.2.1	Funktionsweise von Berechnungsgraphen	503
14.2.2	Erstellen eines Graphen in TensorFlow v1.x	504
14.2.3	Migration eines Graphen nach TensorFlow v2	505
14.2.4	Eingabedaten einlesen mit TensorFlow v1.x	506
14.2.5	Eingabedaten einlesen mit TensorFlow v2.	506
14.2.6	Beschleunigung von Berechnungen mit Funktionsdekoratoren	507
14.3	TensorFlows Variablenobjekte zum Speichern und Aktualisieren von Modellparametern	509
14.4	Gradientenberechnung durch automatisches Differenzieren und GradientTape	514
14.4.1	Berechnung der Gradienten der Verlustfunktion bezüglich trainierbarer Variablen	514
14.4.2	Berechnung der Gradienten bezüglich nicht trainierbarer Tensoren	516

14.4.3	Ressourcen für mehrfache Gradientenberechnung erhalten	516
14.5	Vereinfachung der Implementierung gebräuchlicher Architekturen mit der Keras-API	517
14.5.1	Lösen einer XOR-Klassifikationsaufgabe	521
14.5.2	Flexiblere Modellerstellung mit Keras' funktionaler API	526
14.5.3	Modelle mit Keras' »Model«-Klasse implementieren	528
14.5.4	Benutzerdefinierte Keras-Schichten	529
14.6	TensorFlows Schätzer	533
14.6.1	Merkmalsspalten	534
14.6.2	Machine Learning mit vorgefertigten Schätzern	538
14.6.3	Klassifikation handgeschriebener Ziffern mit Schätzern	543
14.6.4	Benutzerdefinierte Schätzer anhand eines Keras-Modells erstellen	545
14.7	Zusammenfassung	548
15	Bildklassifikation mit Deep Convolutional Neural Networks	549
15.1	Bausteine von Convolutional Neural Networks	549
15.1.1	CNNs und Merkmalshierarchie	550
15.1.2	Diskrete Faltungen	552
15.1.3	Subsampling	561
15.2	Implementierung eines CNNs	563
15.2.1	Verwendung mehrerer Eingabe- oder Farbkanäle	563
15.2.2	Regularisierung eines neuronalen Netzes mit Dropout	566
15.2.3	Verlustfunktionen für Klassifikationen	570
15.3	Implementierung eines tiefen CNNs mit TensorFlow	572
15.3.1	Die mehrschichtige CNN-Architektur	573
15.3.2	Einlesen und Vorverarbeiten der Daten	574
15.3.3	Implementierung eines CNNs mit TensorFlows Keras-API	575
15.4	Klassifikation des Geschlechts anhand von Porträtfotos mit einem CNN	582
15.4.1	Einlesen der CelebA-Datenmenge	582
15.4.2	Bildtransformation und Datenaugmentation	583
15.4.3	Training eines CNN-Klassifikators	590
15.5	Zusammenfassung	596
16	Modellierung sequenzieller Daten durch rekurrente neuronale Netze	597
16.1	Sequenzielle Daten	597

16.1.1	Modellierung sequenzieller Daten: Die Reihenfolge ist von Bedeutung	598
16.1.2	Repräsentierung von Sequenzen	598
16.1.3	Verschiedene Kategorien der Sequenzmodellierung.	599
16.2	Sequenzmodellierung mit RNNs	601
16.2.1	Struktur und Ablauf eines RNNs	601
16.2.2	Aktivierungen eines RNNs berechnen	603
16.2.3	Rückkopplung mit der verdeckten Schicht oder der Ausgabeschicht.	606
16.2.4	Probleme bei der Erkennung weitreichender Interaktionen	609
16.2.5	LSTM-Speicherzellen.	610
16.3	Implementierung von RNNs zur Sequenzmodellierung mit TensorFlow.	612
16.3.1	Projekt 1: Vorhersage der Stimmungslage von IMDB-Filmbewertungen	612
16.3.2	Projekt 2: Sprachmodellierung durch Zeichen mit TensorFlow	629
16.4	Sprache mit dem Transformer-Modell verstehen	642
16.4.1	Der Mechanismus der Selbst-Aufmerksamkeit.	643
16.4.2	Multi-Head-Attention und Transformer-Block	646
16.5	Zusammenfassung	647
17	Synthetisieren neuer Daten mit Generative Adversarial Networks	649
17.1	Einführung in GANs.	649
17.1.1	Autoencoder	650
17.1.2	Generative Modelle zum Synthetisieren neuer Daten.	652
17.1.3	Mit GANs neue Beispiele erzeugen	654
17.1.4	Die Verlustfunktion des Generator- und Diskriminator-Netzes in einem GAN-Modell	655
17.2	Ein GAN von Grund auf implementieren	658
17.2.1	GAN-Modelle mit Google Colab trainieren	658
17.2.2	Implementierung der Generator- und Diskriminator-Netze.	661
17.2.3	Definition der Trainingsdatenmenge	665
17.2.4	Trainieren des GAN-Modells.	667
17.3	Verbesserung der Qualität synthetisierter Bilder durch Convolutional GAN und Wasserstein-GAN	676

17.3.1	Transponierte Faltung	677
17.3.2	Batchnormierung	678
17.3.3	Implementierung des Generators und des Diskriminators	681
17.3.4	Maße für den Unterschied zwischen zwei Verteilungen.	688
17.3.5	Verwendung der EM-Distanz in der Praxis	691
17.3.6	Strafterm	692
17.3.7	Implementierung von WGAN-GP zum Trainieren des DCGAN-Modells.	693
17.3.8	Zusammenbrechen des Verfahrens	697
17.4	Weitere GAN-Anwendungen	699
17.5	Zusammenfassung	700
18	Entscheidungsfindung in komplexen Umgebungen per Reinforcement Learning	701
18.1	Einführung: Aus Erfahrung lernen	702
18.1.1	Reinforcement Learning	702
18.1.2	Definition der Agent-Umgebung-Schnittstelle für ein Reinforcement-Learning-System.	704
18.2	Theoretische Grundlagen des RLs	705
18.2.1	Markov-Entscheidungsprozesse	705
18.2.2	Mathematische Formulierung von Markov- Entscheidungsprozessen	706
18.2.3	RL-Terminologie: Return, Policy und Wertfunktion	710
18.2.4	Dynamische Programmierung und Bellman-Gleichung.	714
18.3	Reinforcement-Learning-Algorithmen.	715
18.3.1	Dynamische Programmierung	715
18.3.2	Reinforcement Learning mit Monte-Carlo-Algorithmen.	718
18.3.3	Temporal Difference Learning	720
18.4	Implementierung eines RL-Algorithmus.	723
18.4.1	OpenAI Gym.	723
18.4.2	Lösung der Grid-World-Aufgabe mit Q-Learning	734
18.4.3	Ein Blick auf Deep Q-Learning	739
18.5	Zusammenfassung und Schlusswort.	747
	Stichwortverzeichnis	751