

Inhaltsverzeichnis

	Über die Autoren	15
	Über die Korrektoren	17
	Einleitung	19
1	Wie Computer aus Daten lernen können	25
1.1	Intelligente Maschinen, die Daten in Wissen verwandeln	25
1.2	Die drei Arten des Machine Learnings	26
1.2.1	Mit überwachtem Lernen Vorhersagen treffen	26
1.2.2	Interaktive Aufgaben durch verstärkendes Lernen lösen ...	29
1.2.3	Durch unüberwachtes Lernen verborgene Strukturen erkennen	31
1.3	Grundlegende Terminologie und Notation	32
1.4	Entwicklung eines Systems für das Machine Learning	34
1.4.1	Vorverarbeitung: Daten in Form bringen	35
1.4.2	Trainieren und Auswählen eines Vorhersagemodells	36
1.4.3	Bewertung von Modellen und Vorhersage anhand unbekannter Dateninstanzen	37
1.5	Machine Learning mit Python	37
1.5.1	Python-Pakete installieren	37
1.5.2	Verwendung der Python-Distribution Anaconda	38
1.5.3	Pakete für wissenschaftliches Rechnen, Data Science und Machine Learning	38
1.6	Zusammenfassung	39
2	Lernalgorithmen für die Klassifizierung trainieren	41
2.1	Künstliche Neuronen: Ein kurzer Blick auf die Anfänge des Machine Learnings	41
2.1.1	Formale Definition eines künstlichen Neurons	42
2.1.2	Die Perzeptron-Lernregel	44
2.2	Implementierung eines Perzeptron-Lernalgorithmus in Python ...	47
2.2.1	Eine objektorientierte Perzeptron-API	47
2.2.2	Trainieren eines Perzeptron-Modells auf die Iris-Datensammlung	51

2.3	Adaptive lineare Neuronen und die Konvergenz des Lernens	56
2.3.1	Straffunktionen mit dem Gradientenabstiegsverfahren minimieren	57
2.3.2	Implementierung eines adaptiven linearen Neurons in Python	59
2.3.3	Verbesserung des Gradientenabstiegsverfahrens durch Merkmalstandardisierung	64
2.3.4	Großmaßstäbliches Machine Learning und stochastisches Gradientenabstiegsverfahren	66
2.4	Zusammenfassung	71
3	Machine-Learning-Klassifizierer mit scikit-learn verwenden	73
3.1	Auswahl eines Klassifizierungsalgorithmus	73
3.2	Erste Schritte mit scikit-learn: Trainieren eines Perzeptrons	74
3.3	Klassenwahrscheinlichkeiten durch logistische Regression modellieren	80
3.3.1	Logistische Regression und bedingte Wahrscheinlichkeiten	81
3.3.2	Gewichtungen der logistischen Straffunktion ermitteln	84
3.3.3	Konvertieren einer Adaline-Implementierung in einen Algorithmus für eine logistische Regression	87
3.3.4	Trainieren eines logistischen Regressionsmodells mit scikit-learn	91
3.3.5	Überanpassung durch Regularisierung verhindern	93
3.4	Maximum-Margin-Klassifizierung mit Support Vector Machines. . .	96
3.4.1	Maximierung des Randbereichs	97
3.4.2	Handhabung des nicht linear trennbaren Falls mit Schlupfvariablen	98
3.4.3	Alternative Implementierungen in scikit-learn	100
3.5	Nichtlineare Aufgaben mit einer Kernel-SVM lösen	101
3.5.1	Kernel-Methoden für linear nicht trennbare Daten	101
3.5.2	Mit dem Kernel-Trick Hyperebenen in höherdimensionalen Räumen finden	103
3.6	Lernen mit Entscheidungsbäumen	107
3.6.1	Maximierung des Informationsgewinns: Daten ausreizen . .	108
3.6.2	Konstruktion eines Entscheidungsbaums	112
3.6.3	Mehrere Entscheidungsbäume zu einem Random Forest kombinieren	116
3.7	k-Nearest-Neighbor: Ein Lazy-Learning-Algorithmus	119
3.8	Zusammenfassung	123

4	Gut geeignete Trainingsdatenmengen: Datenvorverarbeitung	125
4.1	Umgang mit fehlenden Daten	125
4.1.1	Fehlende Werte in Tabellendaten	126
4.1.2	Exemplare oder Merkmale mit fehlenden Daten entfernen.	127
4.1.3	Fehlende Werte ergänzen	128
4.1.4	Die Schätzer-API von scikit-learn	129
4.2	Handhabung kategorialer Daten	130
4.2.1	Nominale und ordinale Merkmale	130
4.2.2	Erstellen einer Beispieldatenmenge	130
4.2.3	Zuweisung von ordinalen Merkmalen	131
4.2.4	Codierung der Klassenbezeichnungen.	132
4.2.5	One-hot-Codierung der nominalen Merkmale.	133
4.3	Aufteilung einer Datensammlung in Trainings- und Testdaten	136
4.4	Anpassung der Merkmale.	138
4.5	Auswahl aussagekräftiger Merkmale	140
4.5.1	L1- und L2-Regularisierung als Straffunktionen	141
4.5.2	Geometrische Interpretation der L2-Regularisierung	141
4.5.3	Dünnbesetzte Lösungen mit L1-Regularisierung	143
4.5.4	Algorithmen zur sequenziellen Auswahl von Merkmalen	147
4.6	Beurteilung der Bedeutung von Merkmalen mit Random Forests	154
4.7	Zusammenfassung	156
5	Datenkomprimierung durch Dimensionsreduktion	159
5.1	Unüberwachte Dimensionsreduktion durch Hauptkomponentenanalyse	159
5.1.1	Schritte bei der Hauptkomponentenanalyse	160
5.1.2	Schrittweise Extraktion der Hauptkomponenten.	161
5.1.3	Totale Varianz und Varianzaufklärung.	164
5.1.4	Merkmalstransformation.	165
5.1.5	Hauptkomponentenanalyse mit scikit-learn	168
5.2	Überwachte Datenkomprimierung durch lineare Diskriminanzanalyse.	171
5.2.1	Hauptkomponentenanalyse vs. lineare Diskriminanzanalyse	172
5.2.2	Die interne Funktionsweise der linearen Diskriminanzanalyse	173

5.2.3	Berechnung der Streumatrizen	174
5.2.4	Auswahl linearer Diskriminanten für den neuen Merkmalsunterraum	176
5.2.5	Projektion in den neuen Merkmalsraum	179
5.2.6	LDA mit scikit-learn	180
5.3	Kernel-Hauptkomponentenanalyse für nichtlineare Zuordnungen verwenden	181
5.3.1	Kernel-Funktionen und der Kernel-Trick	182
5.3.2	Implementierung einer Kernel-Hauptkomponentenanalyse in Python	186
5.3.3	Projizieren neuer Datenpunkte	193
5.3.4	Kernel-Hauptkomponentenanalyse mit scikit-learn	197
5.4	Zusammenfassung	198
6	Best Practices zur Modellbewertung und Hyperparameter-Abstimmung	201
6.1	Arbeitsabläufe mit Pipelines optimieren	201
6.1.1	Die Wisconsin-Brustkrebs-Datensammlung	201
6.1.2	Transformer und Schätzer in einer Pipeline kombinieren	203
6.2	Beurteilung des Modells durch k-fache Kreuzvalidierung	205
6.2.1	2-fache Kreuzvalidierung	205
6.2.2	k-fache Kreuzvalidierung	206
6.3	Algorithmen mit Lern- und Validierungskurven debuggen	211
6.3.1	Probleme mit Bias und Varianz anhand von Lernkurven erkennen	211
6.3.2	Überanpassung und Unteranpassung anhand von Validierungskurven erkennen	214
6.4	Feinabstimmung eines Lernmodells durch Rastersuche	216
6.4.1	Hyperparameter-Abstimmung durch Rastersuche	216
6.4.2	Algorithmenauswahl durch verschachtelte Kreuzvalidierung	218
6.5	Verschiedene Kriterien zur Leistungsbewertung	220
6.5.1	Interpretation einer Wahrheitsmatrix	220
6.5.2	Optimierung der Genauigkeit und der Trefferquote eines Klassifizierungsmodells	222
6.5.3	Receiver-Operating-Characteristic-Diagramme	224
6.5.4	Bewertungskriterien für Mehrfachklassifizierungen	227

6.6	Handhabung unausgewogener Klassenverteilung	228
6.7	Zusammenfassung	231
7	Kombination verschiedener Modelle für das Ensemble Learning . .	233
7.1	Ensemble Learning	233
7.2	Klassifizierer durch Mehrheitsentscheidung kombinieren	237
7.2.1	Implementierung eines einfachen Mehrheitsentscheidungs-Klassifizierers	237
7.2.2	Vorhersagen nach dem Prinzip der Mehrheitsentscheidung treffen	244
7.3	Bewertung und Abstimmung des Klassifizierer-Ensembles	247
7.4	Bagging: Klassifizierer-Ensembles anhand von Bootstrap-Stichproben entwickeln	253
7.4.1	Bagging kurz zusammengefasst	254
7.4.2	Klassifizierung der Wein-Datensammlung durch Bagging	255
7.5	Schwache Klassifizierer durch adaptives Boosting verbessern	258
7.5.1	Funktionsweise des Boostings	259
7.5.2	AdaBoost mit scikit-learn anwenden	263
7.6	Zusammenfassung	266
8	Machine Learning zur Analyse von Stimmungslagen nutzen	267
8.1	Die IMDb-Filmdatenbank	267
8.1.1	Herunterladen der Datensammlung	268
8.1.2	Vorverarbeiten der Filmbewertungsdaten	268
8.2	Das Bag-of-words-Modell	270
8.2.1	Wörter in Merkmalsvektoren umwandeln	270
8.2.2	Beurteilung der Wortrelevanz durch das Tf-idf-Maß	272
8.2.3	Textdaten bereinigen	275
8.2.4	Dokumente in Token zerlegen	277
8.3	Ein logistisches Regressionsmodell für die Dokument- klassifizierung trainieren	279
8.4	Verarbeitung großer Datenmengen: Online-Algorithmen und Out-of-Core Learning	282
8.5	Topic Modeling mit latenter Dirichlet-Allokation	285
8.5.1	Aufteilung von Texten mit der LDA	286
8.5.2	LDA mit scikit-learn	286
8.6	Zusammenfassung	290

9	Einbettung eines Machine-Learning-Modells in eine Webanwendung	291
9.1	Serialisierung angepasster Schätzer mit scikit-learn	291
9.2	Einrichtung einer SQLite-Datenbank zum Speichern von Daten	295
9.3	Entwicklung einer Webanwendung mit Flask	297
	9.3.1 Die erste Webanwendung mit Flask	298
	9.3.2 Formularvalidierung und -ausgabe	300
9.4	Der Filmbewertungsklassifizierer als Webanwendung	304
	9.4.1 Dateien und Ordner – die Verzeichnisstruktur	306
	9.4.2 Implementierung der Hauptanwendung app.py	306
	9.4.3 Einrichtung des Bewertungsformulars	309
	9.4.4 Eine Vorlage für die Ergebnisseite erstellen	310
9.5	Einrichtung der Webanwendung auf einem öffentlich zugänglichen Webserver	312
	9.5.1 Erstellen eines Benutzerkontos bei PythonAnywhere	312
	9.5.2 Hochladen der Filmbewertungsanwendung	313
	9.5.3 Updates des Filmbewertungsklassifizierers	314
9.6	Zusammenfassung	316
10	Vorhersage stetiger Zielvariablen durch Regressionsanalyse	317
10.1	Lineare Regression	317
	10.1.1 Ein einfaches lineares Regressionsmodell	318
	10.1.2 Multiple lineare Regression	319
10.2	Die Lebensbedingungen-Datensammlung	320
	10.2.1 Einlesen der Datenmenge in einen DataFrame	320
	10.2.2 Visualisierung der wichtigen Eigenschaften einer Datenmenge	321
	10.2.3 Zusammenhänge anhand der Korrelationsmatrix erkennen	323
10.3	Implementierung eines linearen Regressionsmodells mit der Methode der kleinsten Quadrate	326
	10.3.1 Berechnung der Regressionsparameter mit dem Gradientenabstiegsverfahren	326
	10.3.2 Abschätzung der Koeffizienten eines Regressionsmodells mit scikit-learn	330
10.4	Anpassung eines robusten Regressionsmodells mit dem RANSAC-Algorithmus	332
10.5	Bewertung der Leistung linearer Regressionsmodelle	335
10.6	Regularisierungsverfahren für die Regression einsetzen	338

10.7	Polynomiale Regression: Umwandeln einer linearen Regression in eine Kurve	340
10.7.1	Hinzufügen polynomialer Terme mit scikit-learn	341
10.7.2	Modellierung nichtlinearer Zusammenhänge in der Lebensbedingungen-Datensammlung	342
10.8	Handhabung nichtlinearer Beziehungen mit Random Forests	346
10.8.1	Entscheidungsbaum-Regression	346
10.8.2	Random-Forest-Regression	348
10.9	Zusammenfassung	351
11	Verwendung nicht gekennzeichnete Daten: Clusteranalyse	353
11.1	Gruppierung von Objekten nach Ähnlichkeit mit dem k-Means-Algorithmus	353
11.1.1	K-Means-Clustering mit scikit-learn	354
11.1.2	Der k-Means++-Algorithmus	358
11.1.3	»Harte« und »weiche« Clustering-Algorithmen	359
11.1.4	Die optimale Anzahl der Cluster mit dem Ellenbogenkriterium ermitteln	362
11.1.5	Quantifizierung der Clustering-Güte mit Silhouettendiagrammen	363
11.2	Cluster als hierarchischen Baum organisieren	368
11.2.1	Gruppierung von Clustern	368
11.2.2	Hierarchisches Clustering einer Distanzmatrix	370
11.2.3	Dendrogramme und Heatmaps verknüpfen	373
11.2.4	Agglomeratives Clustering mit scikit-learn	375
11.3	Bereiche hoher Dichte mit DBSCAN ermitteln	376
11.4	Zusammenfassung	382
12	Implementierung eines künstlichen neuronalen Netzes	383
12.1	Modellierung komplexer Funktionen mit künstlichen neuronalen Netzen	383
12.1.1	Einschichtige neuronale Netze	385
12.1.2	Mehrschichtige neuronale Netzarchitektur	387
12.1.3	Aktivierung eines neuronalen Netzes durch Vorwärtspropagation	390
12.2	Klassifizierung handgeschriebener Ziffern	392
12.2.1	Die MNIST-Datensammlung	393
12.2.2	Implementierung eines mehrschichtigen Perzeptrons	399
12.3	Trainieren eines künstlichen neuronalen Netzes	410
12.3.1	Berechnung der logistischen Straffunktion	410

12.3.2	Ein Gespür für die Backpropagation entwickeln	413
12.3.3	Trainieren neuronaler Netze durch Backpropagation	415
12.4	Konvergenz in neuronalen Netzen.	418
12.5	Abschließende Bemerkungen zur Implementierung neuronaler Netze.	420
12.6	Zusammenfassung	420
13	Parallelisierung des Trainings neuronaler Netze mit TensorFlow.	423
13.1	TensorFlow und Trainingsleistung	423
13.1.1	Was genau ist TensorFlow?	425
13.1.2	TensorFlow erlernen	425
13.1.3	Erste Schritte mit TensorFlow.	426
13.1.4	Mit Array-Strukturen arbeiten.	428
13.1.5	Entwicklung eines einfachen Modells mit TensorFlows Low-level-API	430
13.2	Training neuronaler Netze mit TensorFlows High-level-APIs.	434
13.2.1	Entwicklung mehrschichtiger neuronaler Netze mit TensorFlows Layers-API.	435
13.2.2	Entwicklung eines mehrschichtigen neuronalen Netzes mit Keras	439
13.3	Auswahl der Aktivierungsfunktionen mehrschichtiger neuronaler Netze.	444
13.3.1	Die logistische Funktion kurz zusammengefasst	445
13.3.2	Wahrscheinlichkeiten bei der Mehrfachklassifizierung mit der softmax-Funktion abschätzen.	447
13.3.3	Verbreiterung des Ausgabespektrums mittels Tangens hyperbolicus	448
13.3.4	Aktivierung durch rektifizierte Lineareinheiten.	449
13.4	Zusammenfassung	451
14	Die Funktionsweise von TensorFlow im Detail.	453
14.1	Grundlegende Merkmale von TensorFlow	453
14.2	TensorFlow-Tensoren und deren Rang	454
14.2.1	Rang und Form eines Tensors ermitteln	455
14.3	TensorFlow-Berechnungsgraphen	456
14.4	Platzhalter in TensorFlow	458
14.4.1	Platzhalter definieren.	459
14.4.2	Platzhaltern Daten zuführen	459

14.4.3	Platzhalter für Datenarrays mit variierenden Stapelgrößen definieren	460
14.5	Variablen in TensorFlow	461
14.5.1	Variablen definieren	461
14.5.2	Variablen initialisieren	464
14.5.3	Geltungsbereich von Variablen	465
14.5.4	Wiederverwendung von Variablen	466
14.6	Erstellen eines Regressionsmodells	469
14.7	Ausführung von Objekten in einem TensorFlow-Graphen unter Verwendung ihres Namens	473
14.8	Speichern und wiederherstellen eines Modells in TensorFlow	474
14.9	Tensoren als mehrdimensionale Datenarrays transformieren	477
14.10	Mechanismen der Flusskontrolle beim Erstellen von Graphen verwenden	481
14.11	Graphen mit TensorBoard visualisieren	484
14.11.1	Erweitern Sie Ihre TensorBoard-Kenntnisse	487
14.12	Zusammenfassung	488
15	Bildklassifizierung mit tiefen konvolutionalen neuronalen Netzen	489
15.1	Bausteine konvolutionaler neuronaler Netze	489
15.1.1	CNNs und Merkmals-hierarchie	490
15.1.2	Diskrete Faltungen	491
15.1.3	Subsampling	500
15.2	Implementierung eines CNNs	501
15.2.1	Verwendung mehrerer Eingabe- oder Farbkanäle	501
15.2.2	Regularisierung eines neuronalen Netzes mit Dropout	504
15.3	Implementierung eines tiefen konvolutionalen neuronalen Netzes mit TensorFlow	507
15.3.1	Die mehrschichtige CNN-Architektur	507
15.3.2	Einlesen und Vorverarbeiten der Daten	508
15.3.3	Implementierung eines CNNs mit TensorFlows Low-level-API	509
15.3.4	Implementierung eines CNNs mit TensorFlows Layers-API	521
15.4	Zusammenfassung	527

16	Modellierung sequenzieller Daten durch rekurrente neuronale Netze	529
16.1	Sequenzielle Daten	529
16.1.1	Modellierung sequenzieller Daten: Die Reihenfolge ist von Bedeutung	530
16.1.2	Repräsentierung von Sequenzen	530
16.1.3	Verschiedene Kategorien der Sequenzmodellierung	531
16.2	Sequenzmodellierung mit RNNs	532
16.2.1	Struktur und Ablauf eines RNNs	532
16.2.2	Aktivierungen eines RNNs berechnen	534
16.2.3	Probleme bei der Erkennung weitreichender Interaktionen	537
16.2.4	LSTM-Einheiten	538
16.3	Implementierung eines mehrschichtigen RNNs zur Sequenzmodellierung mit TensorFlow	540
16.4	Projekt 1: Analyse der Stimmungslage in der IMDB-Filmbewertungsdatenbank mit mehrschichtigen RNNs	541
16.4.1	Datenaufbereitung	541
16.4.2	Einbettung	545
16.4.3	Erstellen eines RNN-Modells	547
16.4.4	Der Konstruktor der SentimentRNN-Klasse	548
16.4.5	Die build-Methode	548
16.4.6	Die train-Methode	552
16.4.7	Die predict-Methode	553
16.4.8	Instanziierung der SentimentRNN-Klasse	554
16.4.9	Training und Optimierung des RNN-Modells zur Stimmungsanalyse	554
16.5	Projekt 2: Implementierung eines RNNs zur Sprachmodellierung durch Zeichen mit TensorFlow	555
16.5.1	Datenaufbereitung	556
16.5.2	Erstellen eines RNNs zur Sprachmodellierung durch Zeichen	560
16.5.3	Der Konstruktor	560
16.5.4	Die build-Methode	561
16.5.5	Die train-Methode	564
16.5.6	Die sample-Methode	565
16.5.7	Erstellen und Trainieren des CharRNN-Modells	566
16.5.8	Das CharRNN-Modell im Sampling-Modus	567
16.6	Zusammenfassung und Schlusswort	568
	Stichwortverzeichnis	571