Themen

k-nächste-Nachbarn-Algorithmus

Ein wichtiger Einsatzbereich des Maschinellen Lernens ist die Klassifikation von Daten. Dabei lernt ein Algorithmus anhand von Trainingsdaten, neue Daten in Kategorien oder Klassen einzuordnen, beispielsweise beim Sortieren von E-Mails in Spam und Nicht-Spam.

Der k-nächste-Nachbarn-Algorithmus ist einer der gebräuchlichsten Algorithmen für die Klassifizierung von Daten und zugleich einer der einfachsten Algorithmen des überwachten maschinellen Lernens. Er geht davon aus, dass die Ähnlichkeit zweier Datenpunkte umso größer ist, je näher sie beieinander liegen. Das Lernen erfolgt, indem beispielhafte Datenpunkte gespeichert werden, die bereits den jeweiligen Klassen zugeordnet sind.

Man könnte den k-nächste-Nachbarn-Algorithmus beispielsweise anwenden, um für ein Heizgebläse vorherzusagen, ob es bei einer bestimmten Kombination von Heizleistung und Einschaltdauer ausfällt oder nicht. Als Ausgangsbasis dienen Daten von anderen Heizgebläsen, von denen bereits bekannt ist, ob sie ausgefallen sind ○ oder nicht ●.

Diese Trainingsdaten kann man zusammen mit den Betriebsdaten des neuen Heizgebläses ● (Testdaten)in ein Diagramm eintragen. Nun werden die den Testdaten ● am nächsten platzierten Datenpunkte betrachtet und gezählt, wie viele dieser nahen Datenpunkte zur Klasse „ausgefallen“ ○ gehören und wie viele zur Klasse „nicht ausgefallen“ ●. Die Klasse des neuen Datenpunkts wird von der Mehrheit der ihn umgebenden nächsten Nachbarn bestimmt.

Das k im Namen des Algorithmus bestimmt dabei die Anzahl der einbezogenen Nachbarpunkte.

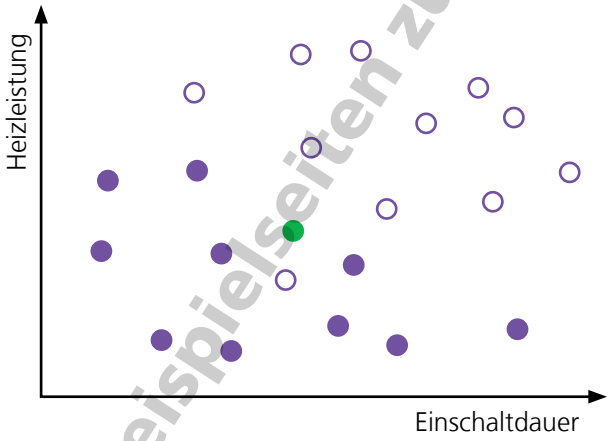
Für k = 1 betrachtet man nur den nächstgelegenen Datenpunkt. Da dieser zur Klasse „kaputtgegangen“ gehört, liegt die Vermutung nahe, dass auch das neue Heizgebläse aufgrund seiner Kombination aus Heizleistung und Betriebsdauer kaputtgehen wird.

Wählt man hingegen k = 3, werden zwei weitere Punkte in die Betrachtung einbezogen, die beide zur Klasse „nicht kaputtgegangen“ gehören. Demnach bestünde beim neuen Heizgebläse eine gute Chance, dass er nicht allzu bald kaputtgeht.

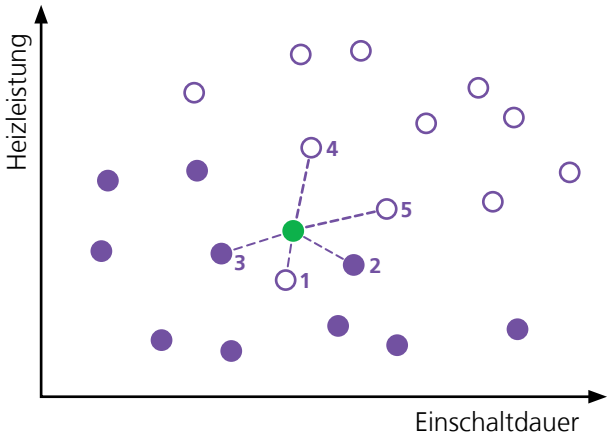
Bei k = 5 sind hingegen wieder die Datenpunkte der Klasse „kaputtgegangen“ in der Mehrzahl, das neue Heizgebläse müsste folglich in die Klasse „kaputtgegangen“ eingruppiert werden.

Die Wahl eines geeigneten Wertes für k ist folglich entscheidend für das Ergebnis, das der k-nächste-Nachbarn-Algorithmus liefert. Bei einem kleinen k können „Ausreißer“ in den Trainingsdaten die Klassenzuordnung verfälschen.

Wird k zu groß gewählt, besteht die Gefahr, Datenpunkte mit einem großen Abstand (und folglich geringer Ähnlichkeit) zum Testpunkt in die Klassifikationsentscheidung einzubeziehen. Dieses Gefahr besteht insbesondere dann, wenn nur wenige Trainingsdaten verfügbar sind oder diese nicht gleichverteilt vorliegen.



Trainingsdaten:
○ Heizgebläse, die ausgefallen sind
● Heizgebläse, die nicht ausgefallen sind



Testdaten:
● neues Heizgebläse
Die 5 nächsten Nachbarn sind gekennzeichnet.

k-nächste-Nachbarn-Algorithmus

Aufgabe 1
Erläutere das Prinzip des k-nächste-Nachbarn-Algorithmus.

Welche Rolle spielt dabei der Parameter k?

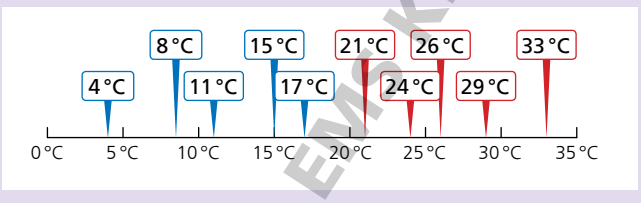
Der k-nächste-Nachbarn-Algorithmus geht davon aus, dass die Ähnlichkeit zweier Datenpunkte umso größer ist, je näher sie beieinander liegen. Das Lernen erfolgt, indem beispielhafte Datenpunkte gespeichert werden, die bereits den jeweiligen Klassen zugeordnet sind. Die Klasse des neuen Datenpunkts wird von der Mehrheit der ihn umgebenden nächsten Nachbarn bestimmt.

Das k im Namen des Algorithmus bestimmt die Anzahl der einbezogenen Nachbarpunkte.

Aufgabe 2
Welche Auswirkungen kann es haben, wenn für k ein gerader Wert (2, 4, 6, ...) gewählt wird?

Wird für k ein gerader Wert gewählt, wird eine gerade Anzahl an Nachbarpunkten einbezogen. Dadurch kann es sein, dass sich die Nachbarpunkte zu gleichen Teilen auf zwei mögliche Klassen verteilen und keine Mehrheit vorliegt.

Aufgabe 3
In der Zeichnung sind als Trainingsdaten zehn Temperaturen eingezeichnet, die den beiden Klassen „kalt“ ☐ und „warm“ ☐ zugeordnet sind.



Ordne diese Testdaten mit Hilfe des k-nächste-Nachbarn-Algorithmus den Klassen „kalt“ ☐ und „warm“ ☐ zu.

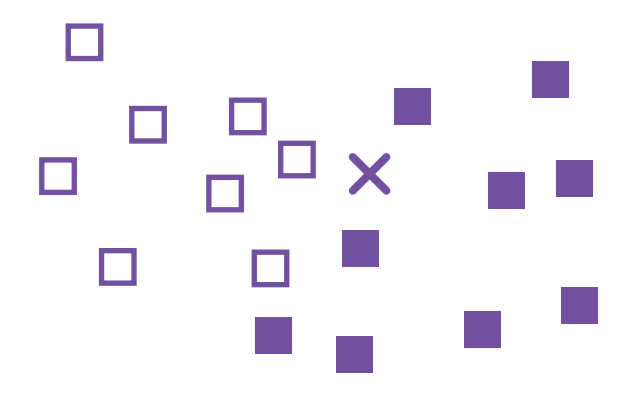
	k	Testwert		k	Testwert
a)	3	19	d)	3	20
b)	5	19	e)	7	20
c)	5	21	f)	4	16

	k	Testwert	Klasse
a)	3	19	kalt
b)	5	19	warm
c)	5	18	kalt
d)	3	20	warm
e)	7	20	warm
f)	4	16	kalt

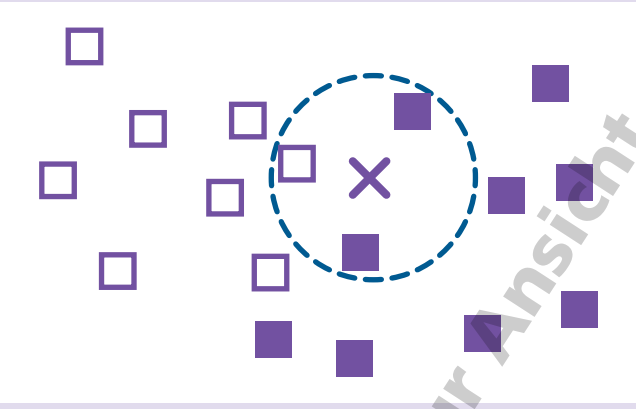
k-nächste-Nachbarn-Algorithmus

Aufgabe 4
Markiere in der Grafik, welche Punkte bei einem k-Wert von 3 durch den k-nächste-Nachbarn-Algorithmus berücksichtigt werden.

Zu welcher Klasse gehört der Testwert ✕?



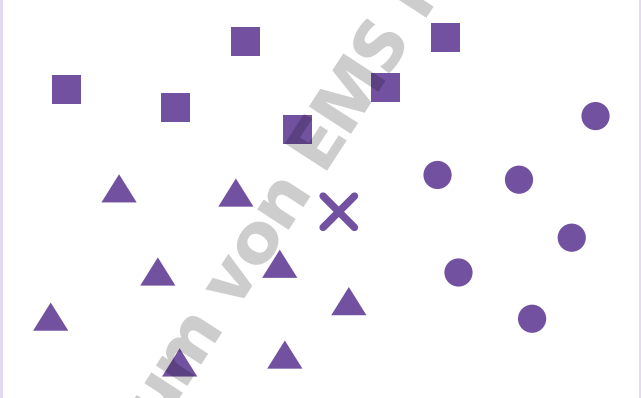
Beispiellösung



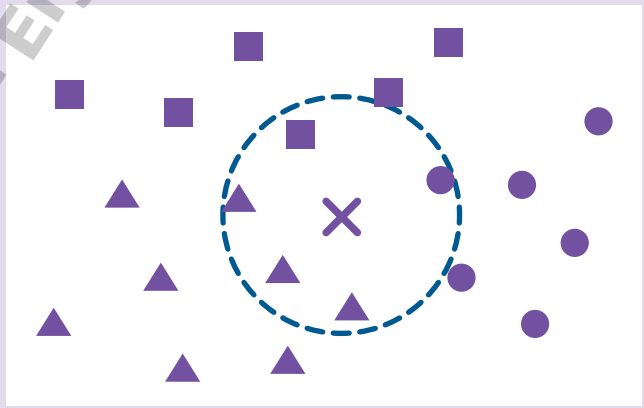
Der Testwert X gehört zur Klasse ■.

Aufgabe 5
Markiere in der Grafik, welche Punkte bei einem k-Wert von 5 durch den k-nächste-Nachbarn-Algorithmus berücksichtigt werden.

Zu welcher Klasse gehört der Testwert ✕?

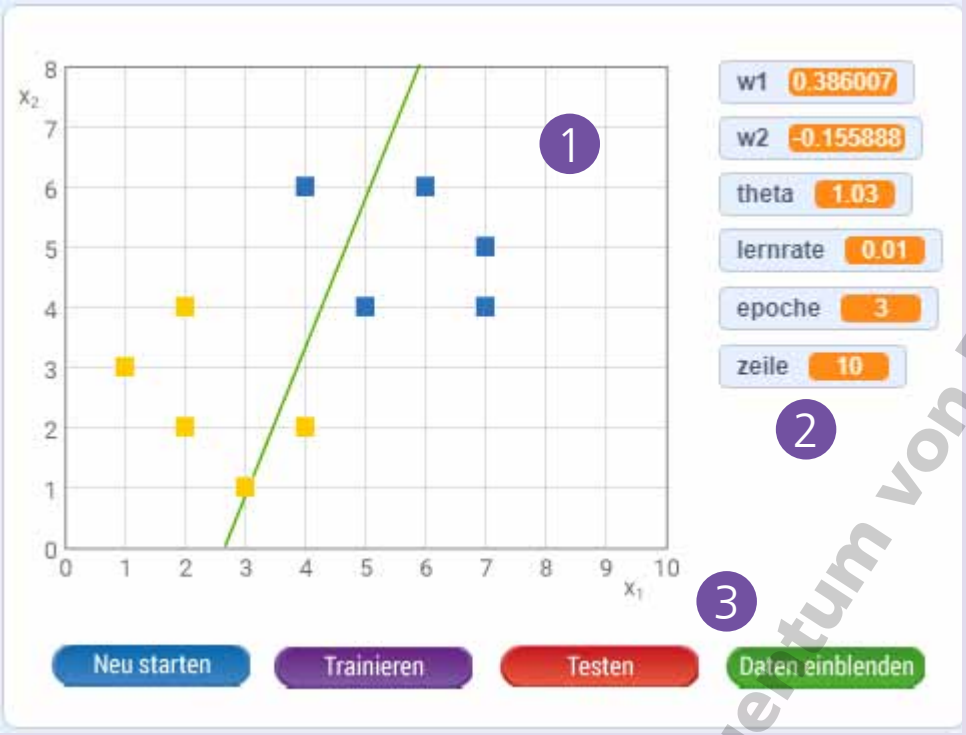


Beispiellösung



Der Testwert X gehört zur Klasse ▲.

Perzeptron simulieren



EMS-Kraus_Perzeptron.sb3

1 Im Koordinatensystem werden die beiden Gruppen der Trainingsdaten als gelbe und blaue Kästchen dargestellt.

Die grüne Gerade wird zu Beginn jedes Durchlaufs und nach jeder Epoche aus den jeweiligen Werten für die Gewichte w1 und w2 und den Schwellenwert theta berechnet.

2 Rechts sind die wichtigsten Variablen dargestellt. Hier kann man beobachten, wie die Epochen und bearbeiteten Trainingsdatensätze hochgezählt werden und wie sich die Variablen w1, w2 und theta dabei verändern.

3 Mit den vier Buttons unten wird das Simulationstool bedient.

Neu starten Mit diesem Button werden eventuell noch laufende Prozesse gestoppt und die Variablen auf 0 gesetzt. Damit befindet sich das Tool in einem definierten Zustand, bevor der nächste Trainingslauf beginnt.

Trainieren Jeder Trainingslauf beginnt mit dem Setzen der Gewichte w1 und w2 als Zufallszahlen im Bereich von –0,5 und 0,5. Der Schwellenwert theta wird auf 1, die Lernrate auf 0,01 gesetzt. Anschließend werden die Variablen w1, w2 und theta gemäß der bekannten Formeln optimiert, um die beiden Punktwolken voneinander zu separieren. Jeder Trainingslauf besteht aus maximal 30 Epochen zu je 10 Datensätzen. Der Trainingslauf bricht ab, sobald drei Epochen ohne Fehler abgeschlossen wurden.

Testen In diesem Modus werden Testwerte eingegeben, die als rotes Kästchen in das Koordinatensystem eingezeichnet werden. Als y_test wird die Klasse 0 (gelb) oder Klasse 1 (blau) angezeigt, in die das Perzeptron den Testwert klassifiziert hat.

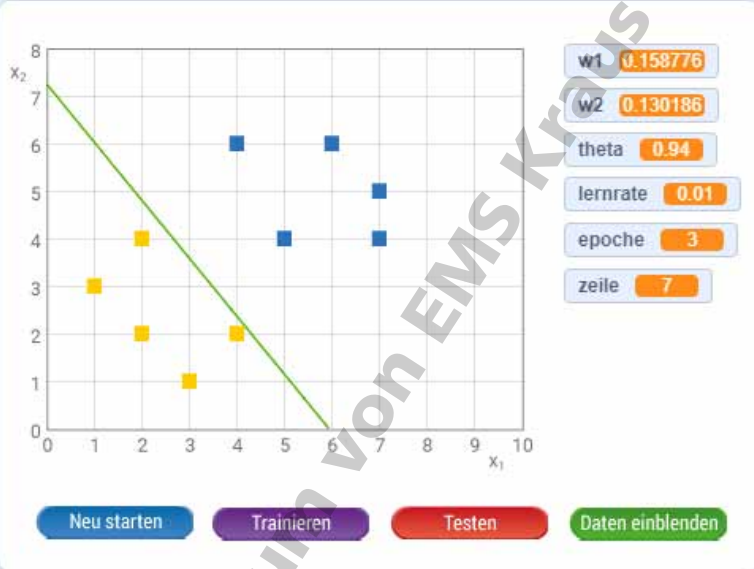
Daten einblenden / **Daten ausblenden** Mit diesen Buttons lassen sich die Trainingsdaten ein- und ausblenden. Die Daten lassen sich in diesen Tabellen auch ändern. Dabei muss man jedoch beachten, dass die beiden Klassen weiterhin linear separierbar bleiben.

Perzeptron simulieren

Aufgabe 1

a) Lade das Programm EMS-Kraus_Perzeptron.sb3 in deine Scratch-Umgebung. Starte einen neuen Durchlauf und trainiere das Perzeptron durch Klick auf „Trainieren“.

b) Beobachte die Veränderung der Variablen w1, w2 und theta während des Trainierens.



c) Bei welchen Startwerten von w1 und w2 benötigt der Algorithmus eher viele, bei welchen wenige Epochen?

Tendenziell viele Trainingsepochen sind notwendig, wenn w1 und w2 beide negativ sind oder wenn w1 oder w2 sehr groß oder sehr klein sind, also beispielsweise 0,4 oder –0,35.

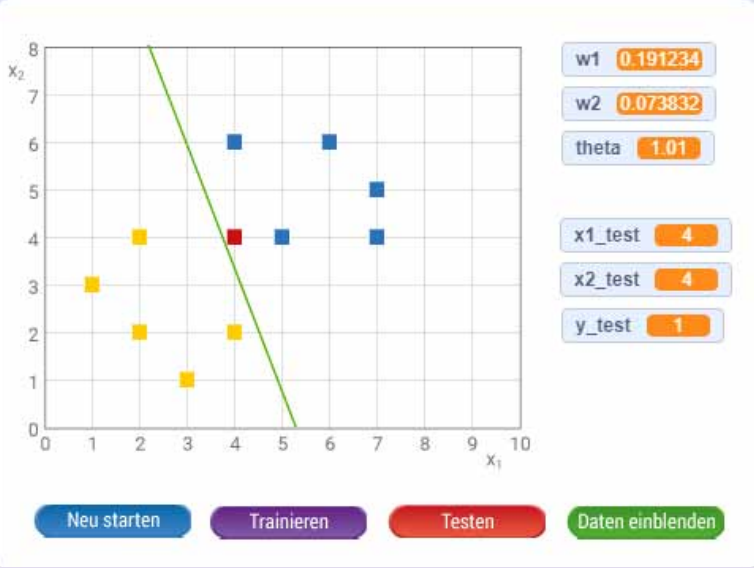
Die wenigsten Trainingsepochen werden benötigt, wenn w1 und w2 beide positiv sind und ihr Wert nahe 0 liegt.

Aufgabe 2

Starte den Testmodus, nachdem das Trainieren abgeschlossen ist.

Gib Testwerte ein und beobachte anhand der Variablen y_test, wie der Algorithmus sie klassifiziert: Klasse 0 (gelb) oder Klasse 1 (blau). Besonders interessant ist dabei der Bereich ganz nah an der Gerade.

Beachte, dass die Koordinaten der Kästchen ihrem Mittelpunkt entsprechen.



Perzeptron simulieren

Aufgabe 3

Definiere zwei neue Punktwolken für die Klasse 0 (gelb) und die Klasse 1 (blau) im Wertebereich für x1 zwischen 0 und 10 und für x2 zwischen 0 und 8.

Beachte, dass die beiden neuen Punktwolken linear separierbar sind.

Blende die Trainingsdaten ein und ändere die Werte für x1, x2 und y (0 oder 1).

Starte einen neuen Durchlauf und trainiere den Algorithmus mit den neuen Trainingsdaten.