Wie falsch ist falsch? (Loss)

Kapitel online

Bevor ein Modell besser werden kann, muss „schlecht“ eine Zahl sein: Ziehe eine Gerade durch die Punktwolke, sieh jeden Fehler als Balken — und drücke den Loss unter 5.

Praxis · Kapitel 12 von 50 · ca. 10 Min. · 150 XP

Einfach gesagt: Jede Abweichung zwischen Vorhersage und echtem Wert wird quadriert und gemittelt — so wird „daneben“ zu einer einzigen Zahl.

In „Was ist ein Neuronales Netz?“ hast du gesehen, WORAUS ein Netz besteht. Aber woher weiß das Netz, ob seine Antworten gut sind? Ganz einfach: **gar nicht** — solange niemand den Fehler misst. Genau das lernst du jetzt. Und du wirst danach jede „Loss-Kurve“ in diesem Kurs lesen können wie ein Profi.

Was du hier lernst

- Warum ein Modell **ohne Loss** gar nicht wissen kann, ob es gut ist
- Wie aus vielen Einzelfehlern **eine einzige Zahl** wird
- Warum Fehler **quadriert** oder als Betrag genommen werden — sonst heben sie sich auf
- Den Unterschied zwischen **MSE und MAE** — und was ein Ausreißer damit anstellt
- Wie derselbe Weg als **Tensor-Operation** aussieht: subtrahieren, quadrieren, mitteln

Für den Unterricht

Einstieg (2 Min): Kapitel gemeinsam lesen, Alltagsbeispiel aufgreifen.

Erkunden am Gerät (6 Min): die interaktiven Module auf herrki.de/115_fehler_messen ausprobieren.

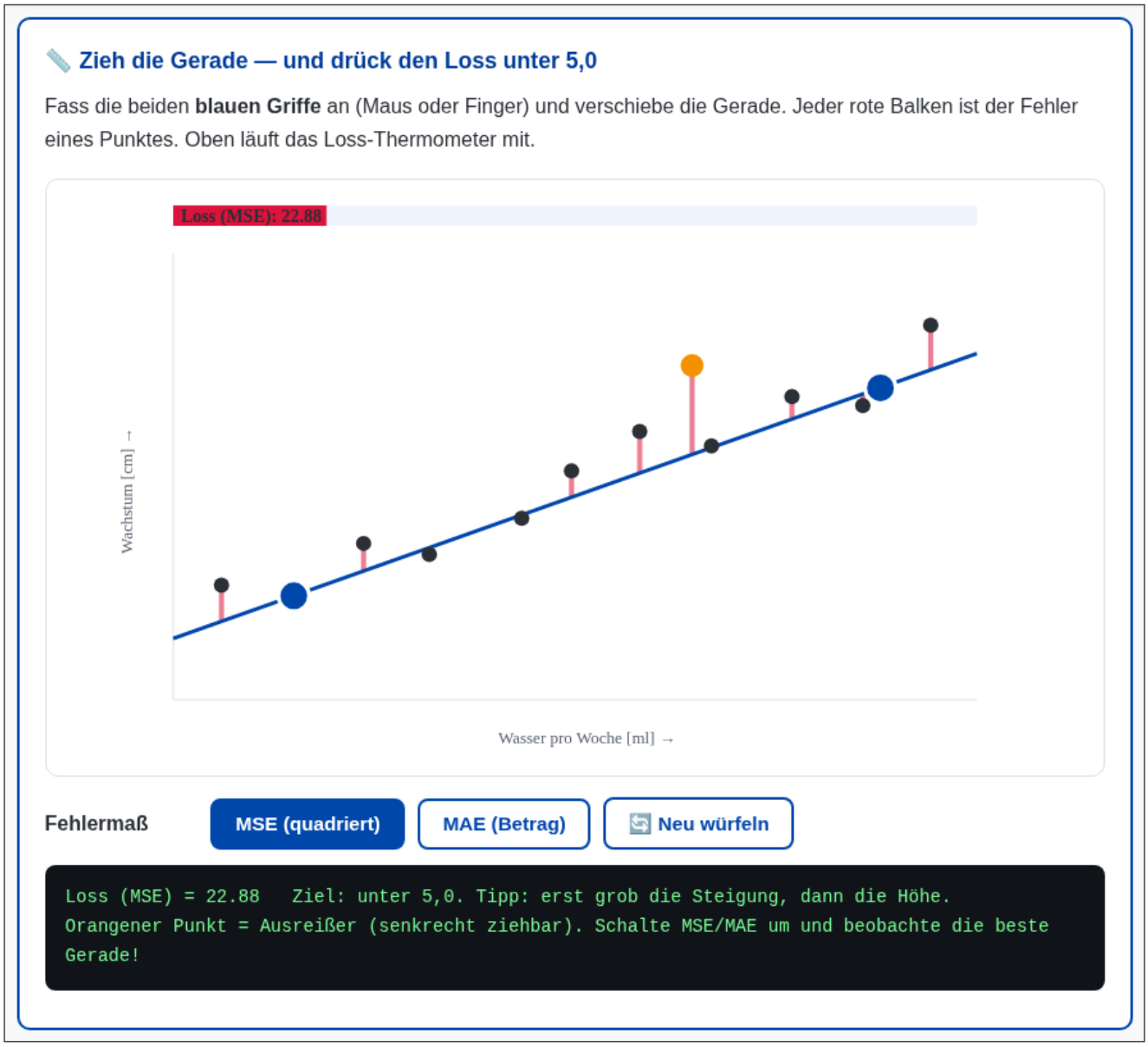
Sichern mit Quiz (2 Min): „Kurzer Check" besprechen, „Das nimmst du mit" festhalten.

Ein Beispiel aus dem Alltag

Eine Pflanze wächst umso mehr, je mehr Wasser sie bekommt (kennst du aus Kapitel „Lineare Regression“ — falls nicht: kommt gleich danach!). Du hast 12 Messungen und legst eine Gerade als „Modell“ hindurch: *Wachstum* = *Steigung* × *Wasser* + *Start*. Aber welche Gerade ist die **beste**? Dafür brauchen wir eine Fehler-Zahl: den **Loss**.

Am Gerät ausprobieren: herrki.de/115_fehler_messen#abschnitt-2

Direkt zum Modul



Interaktives Modul – am Gerät ausprobieren

Was du gerade berechnet hast

1Für jeden Punkt: **Fehler = Vorhersage − echter Wert** (der rote Balken)

2Fehler dürfen sich nicht gegenseitig aufheben (+3 und −3 sind BEIDE schlecht) →

quadrieren (MSE) oder **Betrag** nehmen (MAE)

3**Mittelwert** über alle Punkte → EINE Zahl: der Loss

MSE oder MAE — und der Ausreißer

Im Diagramm gibt es einen **orangenen Sonderpunkt**: den kannst du senkrecht verschieben. Zieh ihn mal weit nach oben und schalte zwischen MSE und MAE um:

⚡ **Der Aha-Moment:** Beim **MSE** wird der Fehler quadriert — aus „10 daneben“ werden 100 Strafpunkte. Ein einziger Ausreißer reißt die beste Gerade zu sich herüber. Beim **MAE** zählt er nur einfach. **Die Wahl der Fehlerformel verändert das Ergebnis** — sie ist eine Design-Entscheidung, keine Naturkonstante.

Runter zum Tensor

Für den Computer sind deine 12 Messungen natürlich Tensoren (Kapitel „Excel/CSV → Tensoren“). Rechnen wir den Loss einmal von Hand — zum Mitrechnen auf **4 der 12 Messungen** verkleinert, mit der Geraden *Wachstum = 0,45 × Wasser + 9*:

```
Wasser [ml] [4]: 20 40 60 80 // vier der zwölf Messungen
Vorhersage [4]: 18 27 36 45 // 0,45·Wasser + 9 = die blaue Gerade
Gemessen [4]: 16 30 33 47 // die schwarzen Punkte, Wachstum in cm
Differenz [4]: +2 −3 +3 −2 // .sub() = Länge der roten Balken
Quadriert [4]: 4 9 9 4 // .square()

Messung 1: 18 − 16 = +2 -> (+2)² = 4 // Gerade lag zu hoch
Messung 2: 27 − 30 = −3 -> (−3)² = 9 // Gerade lag zu tief
Loss : (4 + 9 + 9 + 4) / 4 = 6,5 // .mean() = EINE Zahl
```

Und jetzt der orange Ausreißer aus dem Diagramm dazu — gleiche Gerade, eine Messung mehr:

```
Bei 60 ml sind es nicht 33, sondern 45 cm -> 36 − 45 = −9

Differenz [5]: +2 −3 +3 −2 −9
Quadriert [5]: 4 9 9 4 81 // EIN Punkt, 81 Strafpunkte
MSE : (4+9+9+4+81) / 5 = 21,4 // aus 6,5 wird 21,4
MAE : (2+3+3+2+ 9) / 5 = 3,8 // .abs() statt .square()
```

So heißt derselbe Weg in TensorFlow.js („Tensor-Operationen“) — nur eben mit allen 12 Werten: aus [12] Vorhersagen werden [12] Differenzen, [12] Quadrate, und ganz am Ende eine einzige Zahl ohne jede Achse:

```
// Vorhersagen und echte Werte sind Tensoren:
const fehler = vorhersagen.sub(echteWerte); // Schritt 1: Differenz [12]
const bestraft = fehler.square(); // Schritt 2: quadrieren [12]
const loss = bestraft.mean(); // Schritt 3: Mittelwert -> EINE Zahl
```

Mehr steckt hinter dem geheimnisvollen „Loss“ nicht: **subtrahieren, quadrieren, mitteln.**

Zurück zur Anwendung

Wenn du im nächsten Kapitel `model.fit()` aufrufst, tut TensorFlow genau das, was du eben von Hand gemacht hast: Es berechnet diese eine Zahl — und verschiebt die Gerade dann Schritt für Schritt so, dass die Zahl kleiner wird. WIE es die richtige Richtung findet, ohne zu raten? Das ist der geniale Trick im nächsten Kapitel.

Zum Weiterlesen

- **Wenn MSE nicht passt:** Michael Nielsen, *Neural Networks and Deep Learning*, Kapitel 3 — neuralnetworksanddeeplearning.com/chap3.html — zeigt, warum es neben dem quadrierten Fehler noch die Cross-Entropy gibt und wann sie besser lernt.
- **Kurz und praktisch:** Google Machine Learning Crash Course, *Loss* — developers.google.com/machine-learning/crash-course (developers.google.com/machine-learning/crash-course/linear-regression/loss) — dieselbe Rechnung noch einmal in zehn Minuten, mit MSE und MAE nebeneinander.
- **Das Lehrbuch dazu:** Goodfellow, Bengio, Courville, *Deep Learning*, Kapitel 5 „Machine Learning Basics“ — deeplearningbook.org/contents/ml.html — ordnet den Loss in den großen Rahmen ein: Kostenfunktion, Schätzer, Verallgemeinerung.
- **Auf HerrKI:** 118 — Gradient Descent zum Anfassen (was mit dieser Zahl passiert) und 060 — Tensor-Operationen (`sub`, `square`, `mean` im Detail)

Kurzer Check – mit Lösung

Lösungen für Lehrkräfte – für Lernende die Online-Version nutzen.

Frage 1: Wozu braucht ein Modell überhaupt einen Loss?

- Um schneller zu rechnen.
- ✔ **Damit „schlecht“ eine messbare Zahl ist — nur so lässt sich gezielt verbessern.**
- Um die Daten zu speichern.

Ohne Fehler-Zahl gibt es keine Richtung für „besser“. Der komplette Lern-Kreislauf aus Kapitel 005 hängt an diesem einen Wert.

Frage 2: Warum werden die Fehler quadriert oder als Betrag genommen?

- ✔ **Damit sich positive und negative Fehler nicht gegenseitig auslöschen.**
- Weil Computer keine negativen Zahlen kennen.
- Damit der Loss größer aussieht.

+3 und −3 daneben ergäben sonst im Schnitt 0 — „perfekt“, obwohl beides falsch war. Quadrat oder Betrag machen jeden Fehler zu Strafpunkten.

Frage 3: Ein Datensatz enthält einen extremen Ausreißer. Was passiert beim MSE?

- Nichts — Ausreißer werden ignoriert.
- Der MSE lässt sich dann nicht mehr berechnen.
- ✔ **Der quadrierte Riesenfehler dominiert den Loss — die Gerade wird zum Ausreißer hingezogen.**

$10^2 = 100$: Quadrieren bestraft große Fehler überproportional. Deshalb nimmt man bei verrauschten Daten mit Ausreißern oft lieber MAE.

Das nimmst du mit

- **Ohne Loss kann kein Modell besser werden.** Erst wenn „schlecht“ eine Zahl ist, gibt es überhaupt eine Richtung für „besser“.
- **Fehler dürfen sich nicht gegenseitig aufheben.** +3 und −3 ergäben im Schnitt 0 — deshalb quadrieren (MSE) oder den Betrag nehmen (MAE).
- **Aus vielen Einzelfehlern wird EINE Zahl.** Subtrahieren, quadrieren, mitteln — als Tensor-Kette `sub()` → `square()` → `mean()`.
- **MSE bestraft Ausreißer überproportional.** Aus „9 daneben“ werden 81 Strafpunkte: ein einziger Punkt zieht die ganze Gerade zu sich herüber.
- **Die Fehlerformel ist eine Design-Entscheidung.** Verrauschte Daten mit Ausreißern → eher MAE. Keine Naturkonstante, sondern deine Wahl.

So zitierst du dieses Kapitel

HerrKI: Wie falsch ist falsch? (Loss). herrki.de/115_fehler_messen, ein privates Hobbyprojekt von Terragon.de, abgerufen am 09.09.2026.