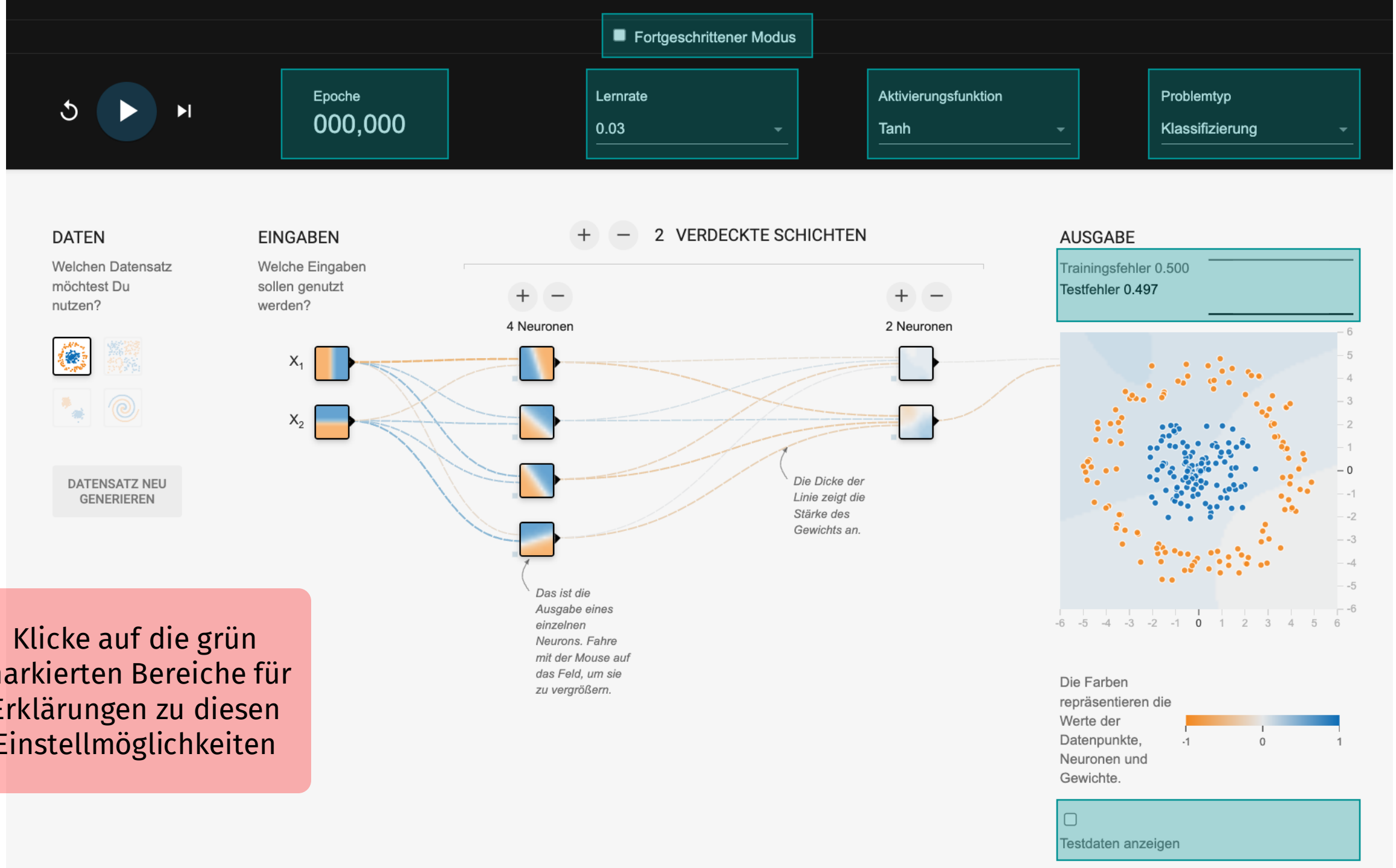


Klicke auf die grün markierten Bereiche für Erklärungen zu diesen Einstellmöglichkeiten

Epoche

- Epoche: Berechnungen von Vorhersagen für die Trainingsdaten mit dem aktuellen Modell
- Anpassung der Parameter (Gewichte & Bias-Werte) mithilfe der Fehlerfunktion

➥ Ziel: Fehler verkleinern

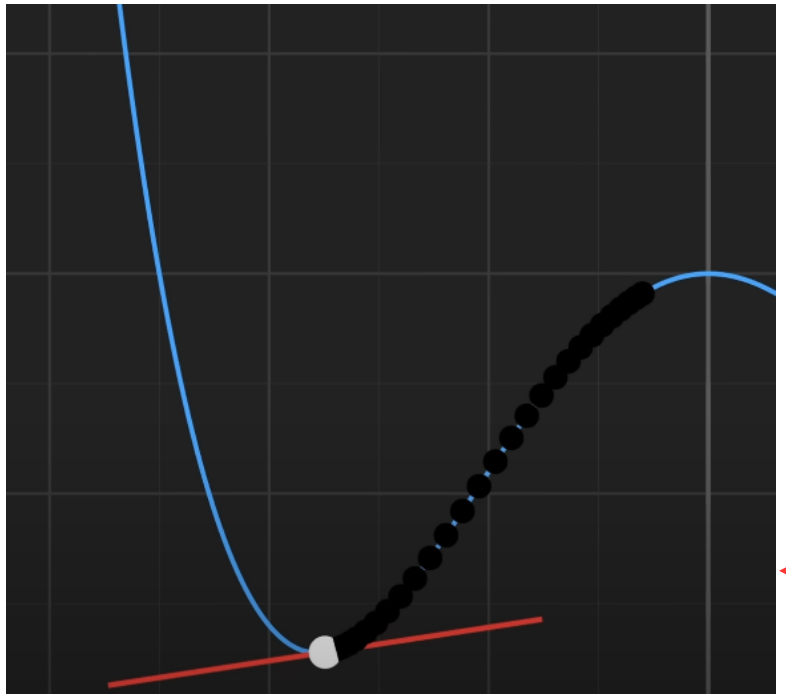
- Nächste Epoche: Modell mit den angepassten Parametern für die Vorhersage nutzen

Zu wenige Epochen ➥ Modell wird nicht ausreichend angepasst (Underfitting)

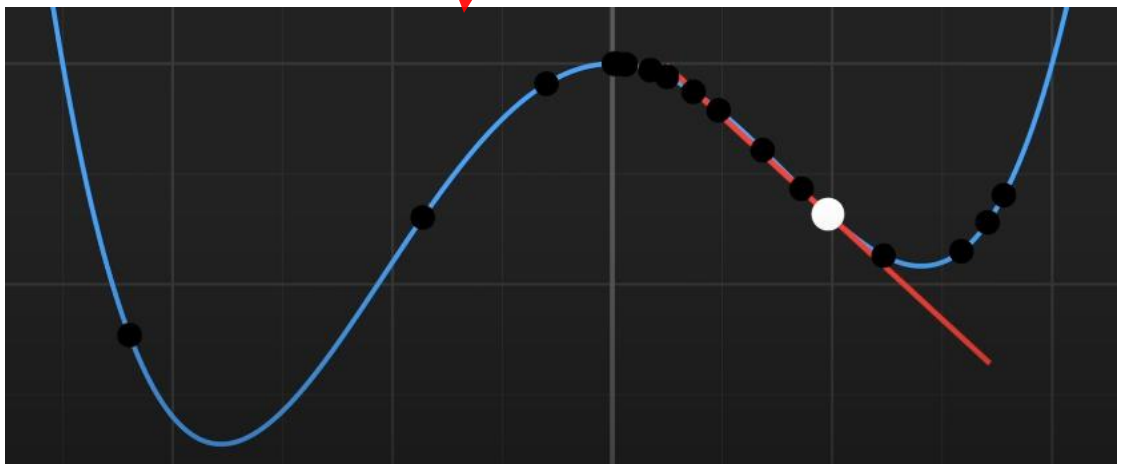
Zu viele Epochen ➥ Modell wird zu stark an Trainingsdaten angepasst (Overfitting)

Zur Übersicht

Lernrate



Suche eines lokalen Minimums für die Fehlerfunktion



- Wie stark sollen die Parameter des Modells in jeder Epoche verändert werden?
- Zu kleine Schritte
  ➥ Training dauert zu lange
- Zu große Schritte
  ➥ Minimum wird verfehlt

Zur Übersicht

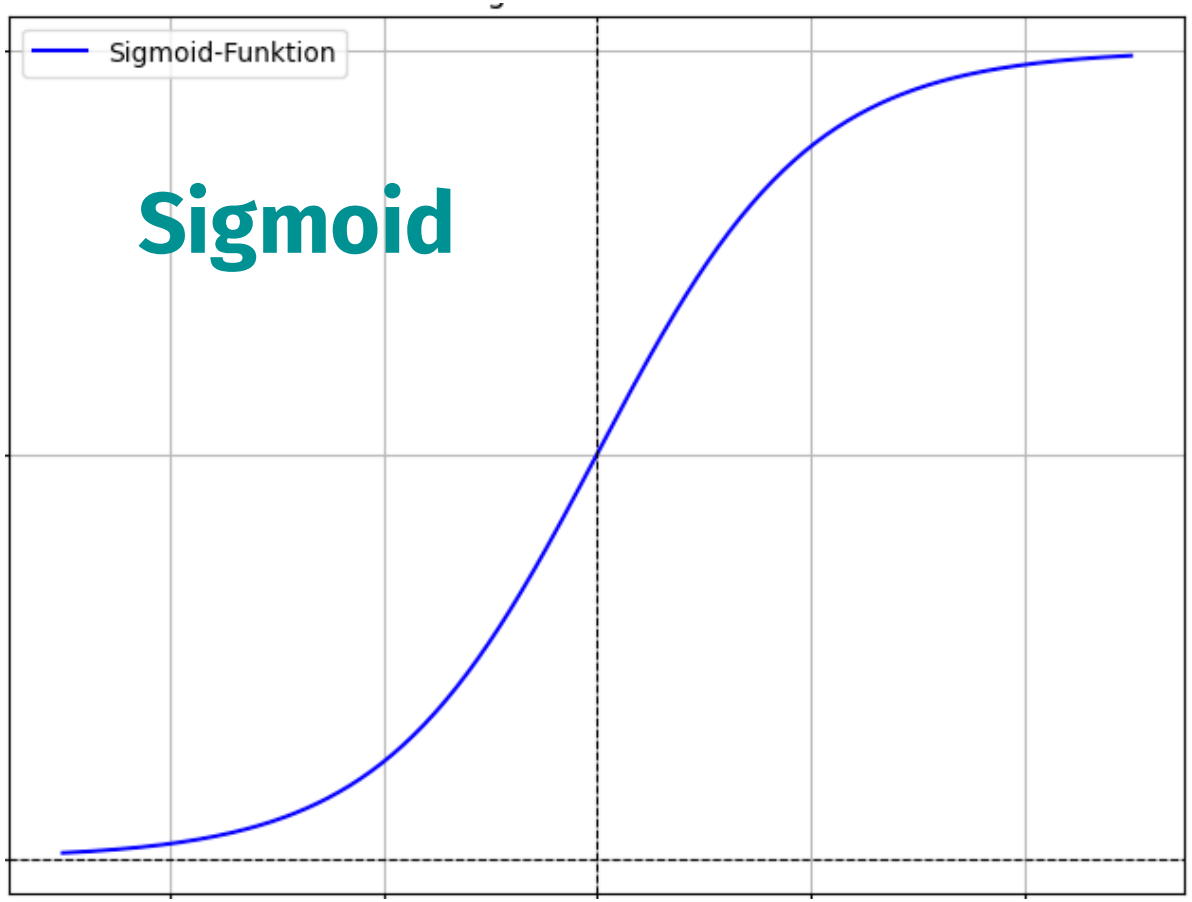
Aktivierungsfunktion

- Linear: Viele Klassifizierungsprobleme sind nicht möglich, da dabei nicht-linear getrennt werden muss
  - Alle weiteren Aktivierungsfunktionen sind nicht linear!
- Tanh, Sigmoid: Klassische Aktivierungsfunktionen mit relativ hohem Rechenaufwand
- ReLU: Meistgenutzt, geringerer Rechenaufwand

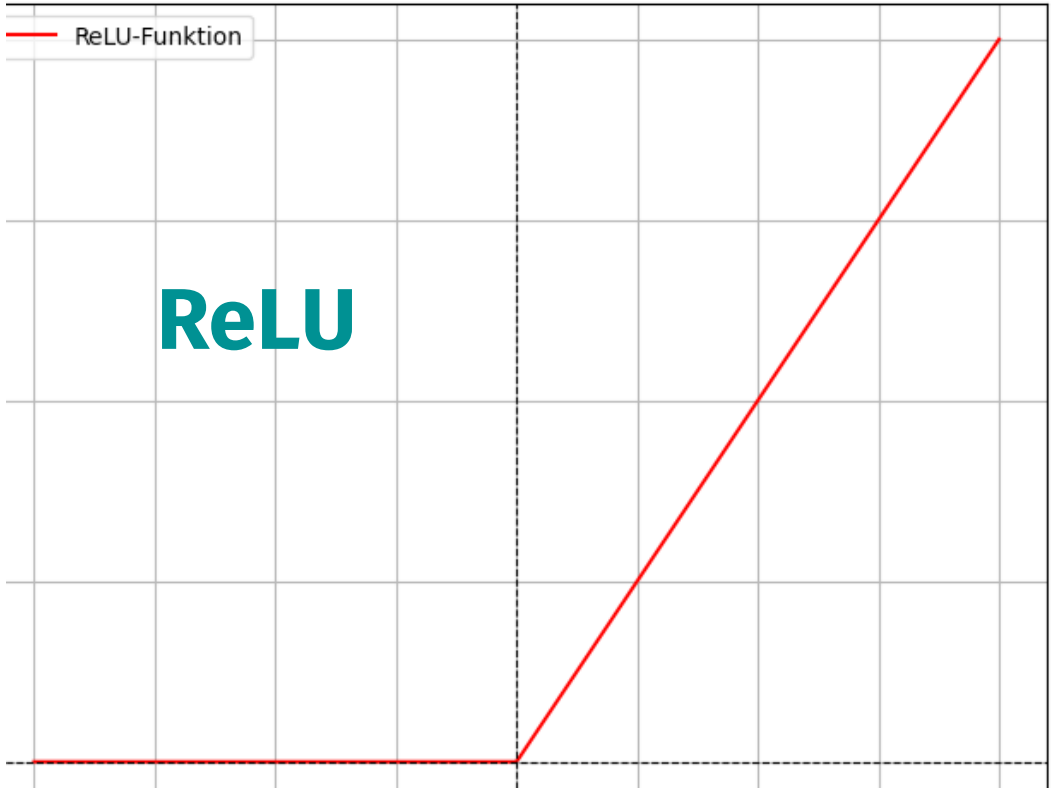
```
def relu(x: int):
    if x < 0:
        return 0
    else:
        return x
```

Zur Übersicht

Beispielhafte Aktivierungsfunktionen

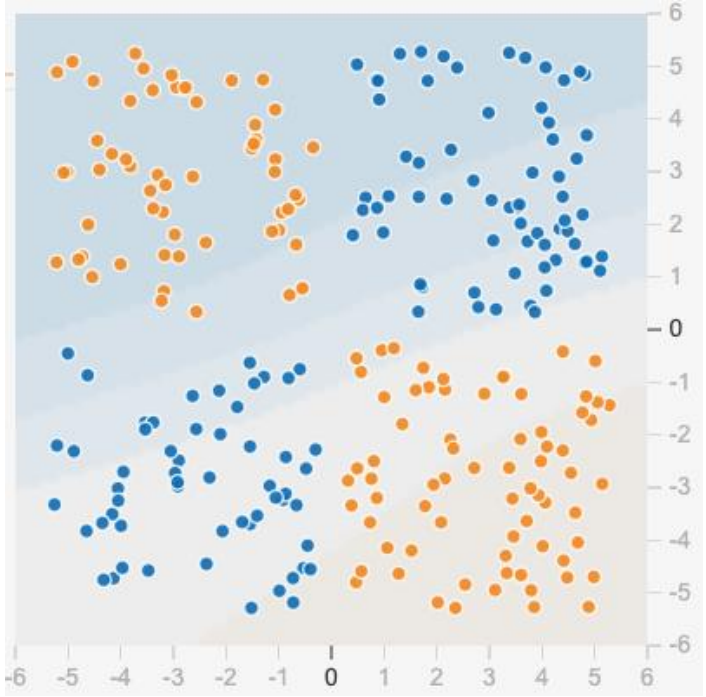



```python
def relu(x: int):
    if x < 0:
        return 0
    else:
        return x
```

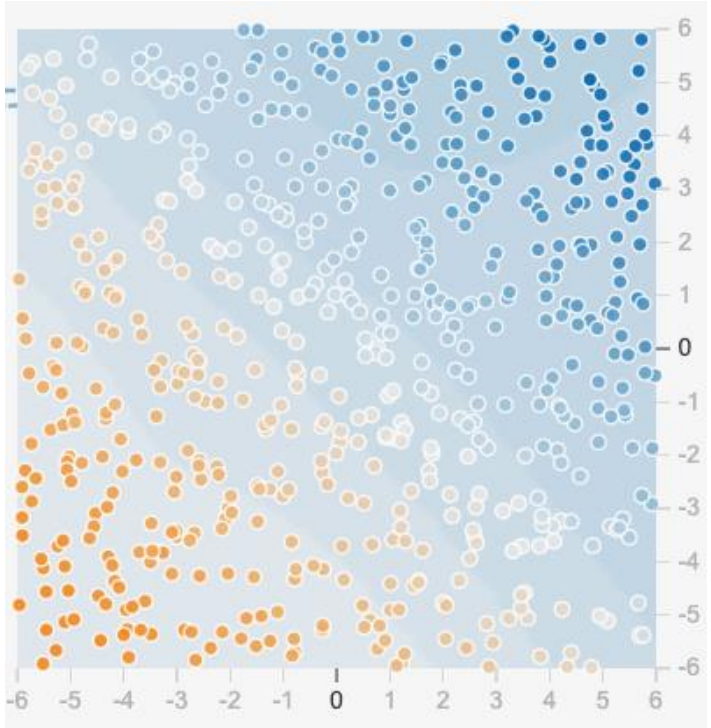


Problemtyp

- Klassifizierung: Datenpunkte in Klassen eingeteilt
  - Hier: Datenpunkte haben entweder den Wert -1 oder 1
  - Darstellung: Farbe orange oder blau



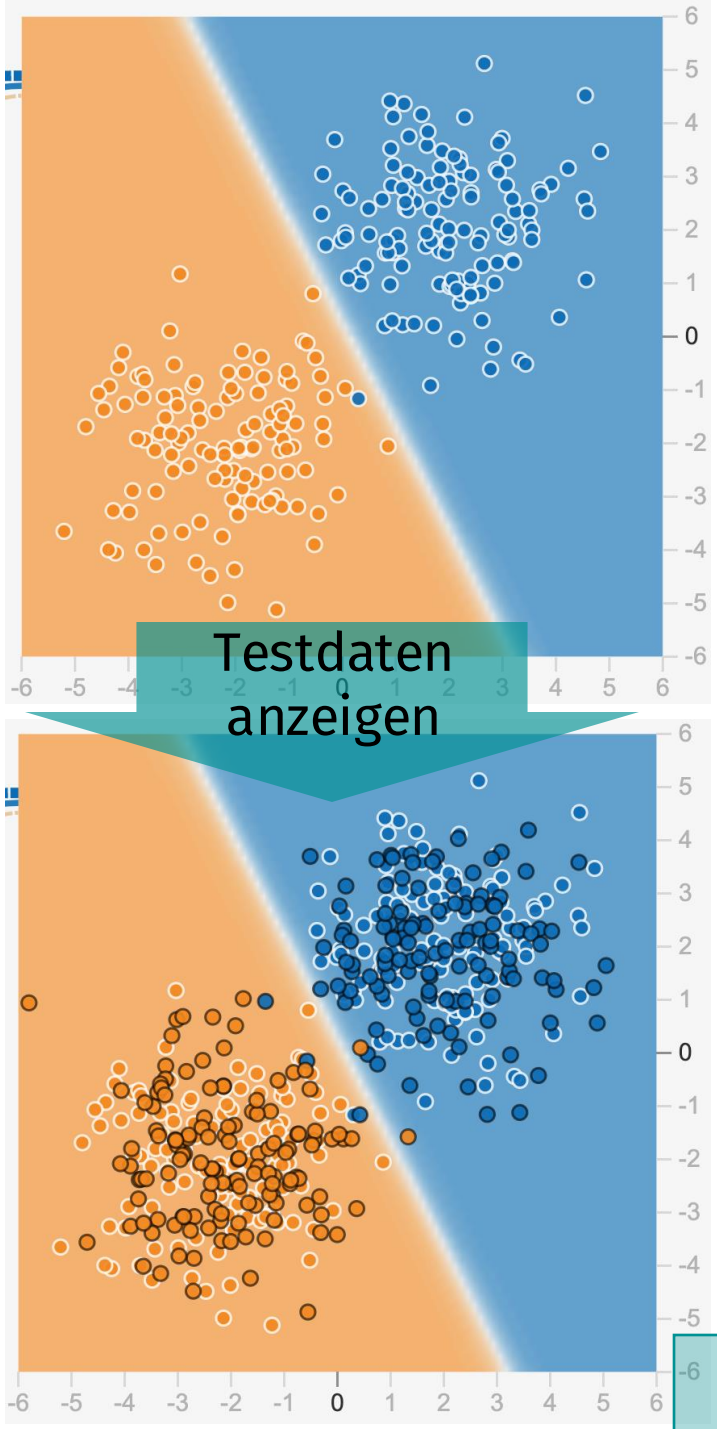
- Regression: Kontinuierliche Werte
  - Hier: Datenpunkte haben kontinuierliche Werte zwischen -1 und 1
  - Darstellung: Farbabstufungen zwischen dunkelblau (-1), weiß (0) und orange (1)



Zur Übersicht

Testdaten anzeigen

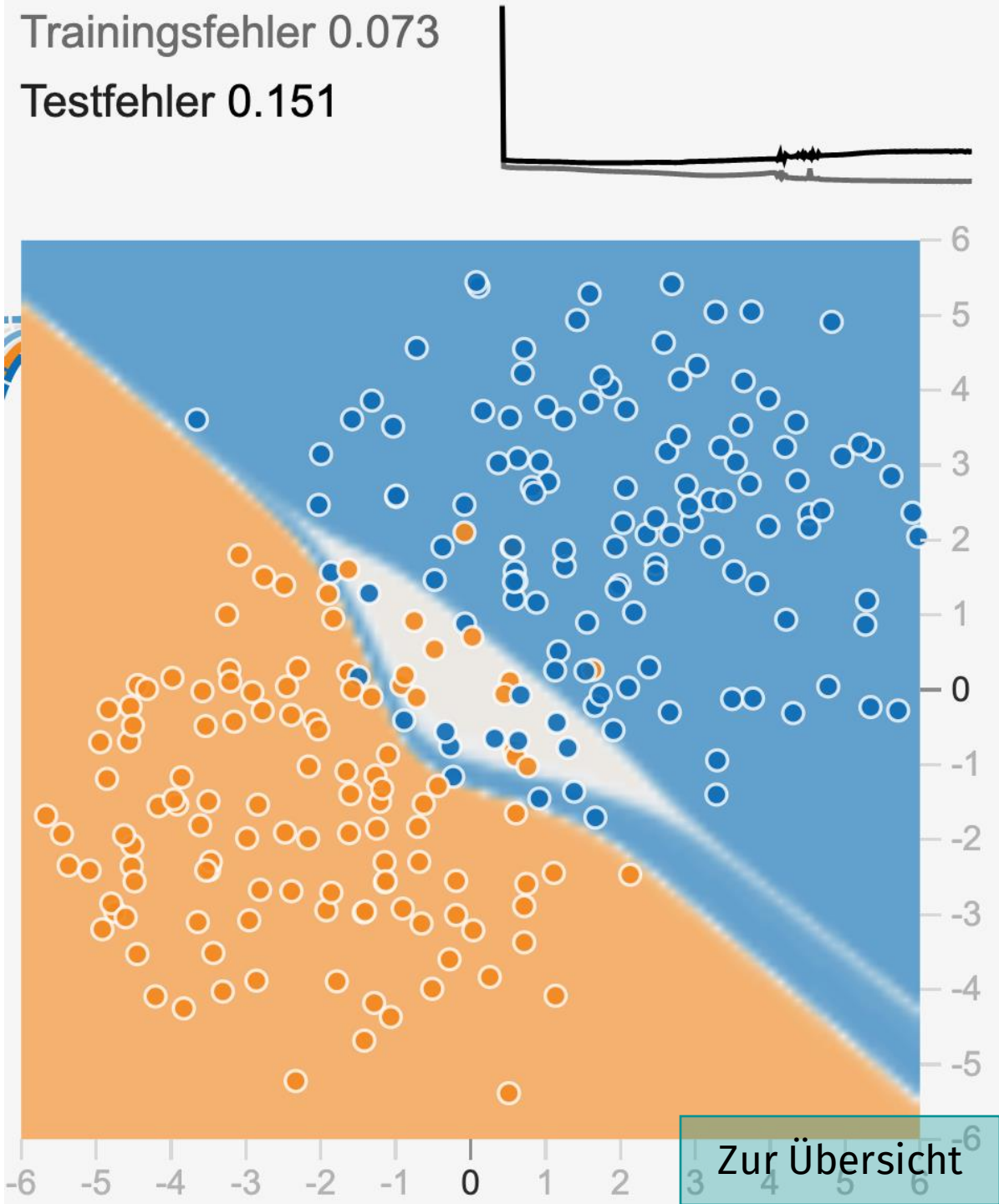
- Ziel:
  - Keine exakte Anpassung an Trainingsdaten
  - Modell soll für Daten funktionieren, mit denen nicht trainiert wurde
    ➥ Generalisierung
- Überprüfen mit Testdaten, ob das gewählte Modell (Anzahl Schichten/Neuronen, Aktivierungsfunktion) geeignet ist, um zu generalisieren



Zur Übersicht

Trainings- und Testfehler

- Trainingsfehler: Wert der Fehlerfunktion für die Trainingsdaten
- Testfehler: Wert der Fehlerfunktion für die Trainingsdaten
- Trainingsfehler gering, Testfehler hoch
  ➥ Generalisierung funktioniert nicht (Overfitting)



Zur Übersicht

Fortgeschrittener Modus

Zurück

Epoche
000,000

Lernrate
0.03

Aktivierungsfunktion
Tanh

Regularisierung
Keine

Regularisierungsrate
0

Problemtyp
Klassifizierung

DATEN

Welchen Datensatz möchtest Du nutzen?

Anteil an Trainingsdaten: 50%

Rauschen: 0

Stapelgröße: 10

DATENSATZ NEU GENERIEREN

EINGABEN

Welche Eingaben sollen genutzt werden?

X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$

2 VERDECKTE SCHICHTEN

4 Neuronen

2 Neuronen

Die Dicke der Linie zeigt die Stärke des Gewichts an.

Das ist die Ausgabe eines einzelnen Neurons. Fahre mit der Maus auf das Feld, um sie zu vergrößern.

AUSGABE

Trainingsfehler 0.500

Testfehler 0.497

Die Farben repräsentieren die Werte der Datenpunkte, Neuronen und Gewichte.

-1 0 1

Testdaten anzeigen

Diskrete Ausgabe

Regularisierung

- Strategie zur Vermeidung von Overfitting
- Komplexe Modelle werden „bestraft“ (Höherer Wert der Fehlerfunktion)

➥ Ziel: Möglichst geringe Komplexität des Modells

- L1-Regularisierung: Summe der Beträge der Gewichte wird auf Fehlerfunktion addiert
  - kann einzelne Gewichte exakt auf 0 setzen (feature selection)
- L2-Regularisierung: Summe der Quadrate der Gewichte wird auf Fehlerfunktion aufaddiert
  - typischerweise bevorzugt; Gleichmäßigere Verkleinerung der Gewichte

Zur Übersicht

Regularisierungsrate (am Bsp. L1)

- Die Summe der Beträge der Gewichte wird mit der Regularisierungsrate multipliziert, bevor sie zur Fehlerfunktion addiert wird:

$$\begin{aligned} &Fehler \\ &= Abweichung\ zwischen\ Prognosen\ und\ Trainingsdaten \\ &+ Regularisierungsrate \cdot Summe\ der\ Beträge\ der\ Gewichte \end{aligned}$$

- Große Regularisierungsrate (Vorsicht: Zu groß ➥ Underfitting)

➥ Großer Regularisierungsterm

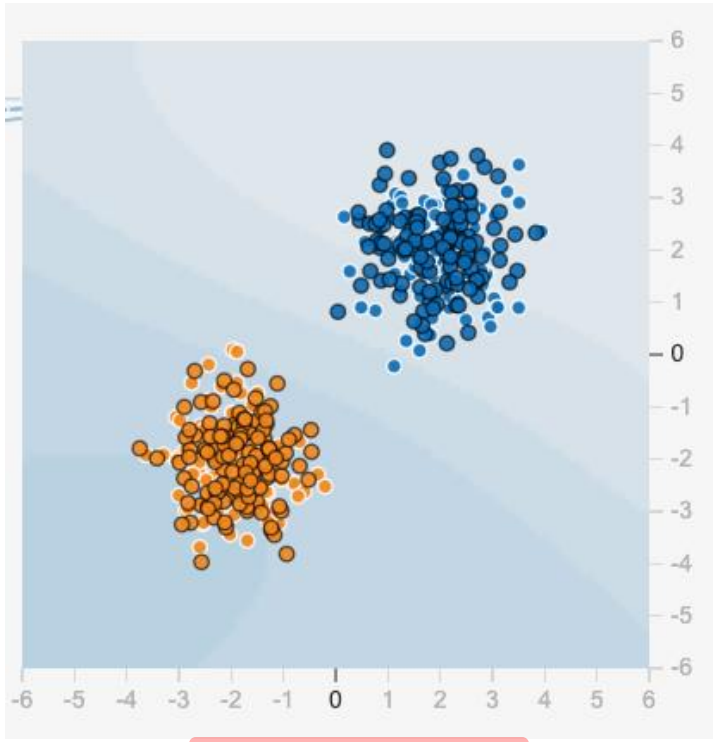
➥ Kleine Gewichte gehen gegen 0

➥ Modell wird einfacher

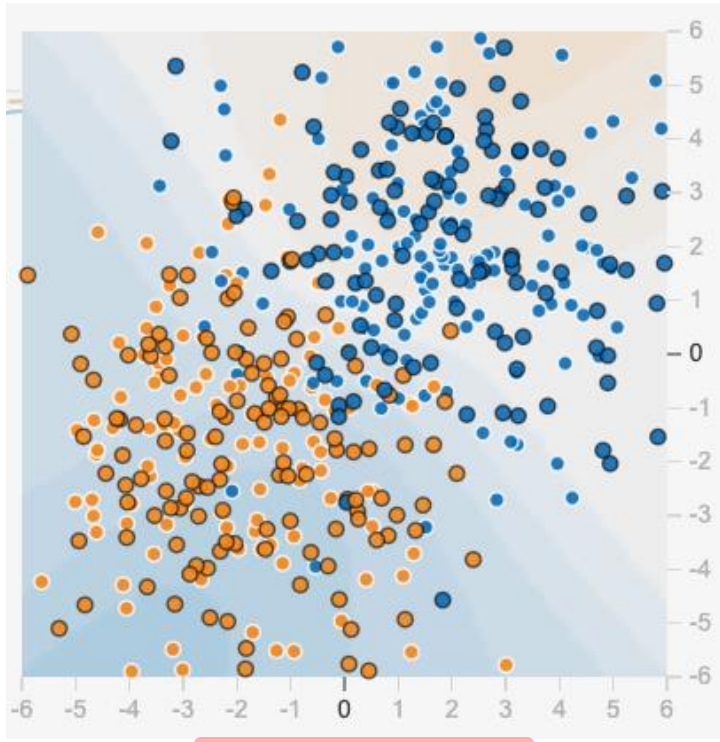
Zur Übersicht

Rauschen

- Großer Rausch-Faktor ➥ Viele Ausreißer ➥ Trennung schwerer
- Insbesondere größere Gefahr des Overfittings ➥ Zu starke Anpassung an Ausreißer



Rauschen: 0



Rauschen: 50

Zur Übersicht

Stapelgröße (engl. batch size)

- Aufteilung des Trainingsdatensatzes in zufällige Stapel
- Berechnung von Fehler + Anpassung nur für einzelne Stapel

➥ Geringere Rechenzeit

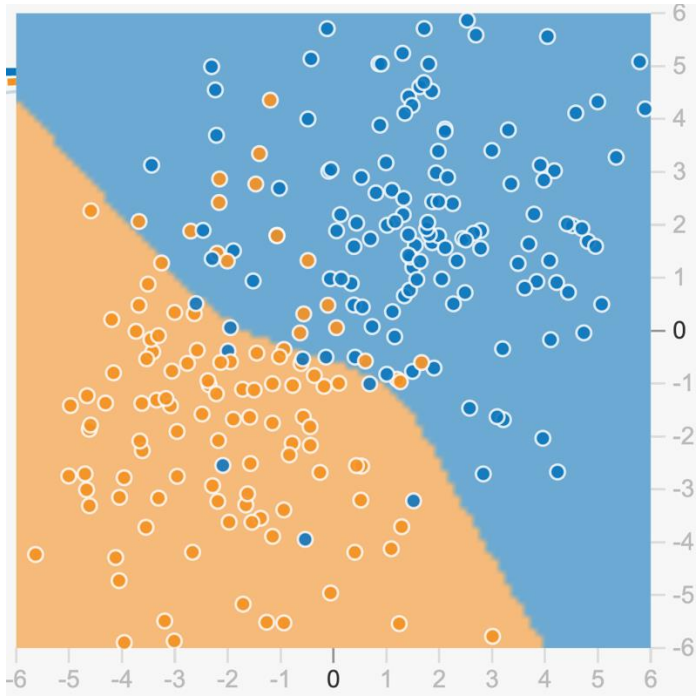
- Zusätzlicher Vorteil: Kein Steckenbleiben an Sattelpunkten der Fehlerfunktion, aufgrund der ungenauen Berechnung
- Epoche besteht damit aus mehreren Iterationen, bis alle Stapel durchgegangen wurden

Zur Übersicht

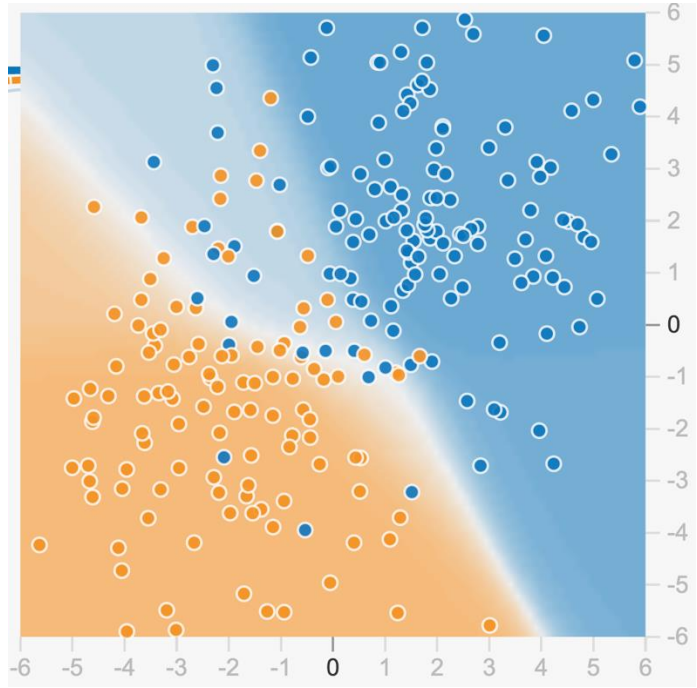
Diskrete Ausgabe

- Diskrete Ausgabe:

Nur Ausgaben -1 (orange) oder 1 (blau) sind möglich



- Gegensatz: Kontinuierliche Ausgaben zwischen -1 und 1
  - Darstellung durch abgeschwächte Farbwerte



Zur Übersicht