MLP-Rechner

Empfehlungen für Trainingsparameter

Multilayer Perceptron – Hyperparameter-Leitfaden

Deutsch

Wolfgang Böhme

wolfgang.boehme@economic-modeling.de

www.economic-modeling.de

2. November 2026

Dokumentationen dieser Webseite wurden mit Unterstützung der KI Claude (Anthropic) entwickelt.

Inhaltsverzeichnis

1 Grundlegende Konzepte **2**

2 Beispieldatensätze im Rechner v20 **2**

3 Kreuzkurse (UZS): 4 Inputs → EUR/UZS **2**

4 Währungsdaten (UZS): 4 Inputs → EUR **3**

5 Logische Gatter: XOR, AND, OR **4**

6 Kreis-Klassifikation (200 Samples) **5**

7 Einfache Regression – Sinuskurve (50 Samples) **5**

8 Weitere typische Anwendungsfälle **6**
8.1 Iris-Blumenklassifikation (3 Klassen) 6
8.2 Zeitreihen-Regression (z. B. Temperatur, Aktienkurse) 6
8.3 Hauspreisvorhersage (mehrere Merkmale) 7
8.4 Spam-Erkennung / Binäre Textklassifikation 7

9 Allgemeine Faustregeln **8**

Grundlegende Konzepte

Aktivierungsfunktionen im Überblick:

- **ReLU** (*Rectified Linear Unit*): $f(x) = \max(0, x)$ – schnell, stabil, Standard für Hidden Layers bei Regression
- **Sigmoid**: $f(x) = \frac{1}{1+e^{-x}}$ – Ausgabe im Bereich $[0, 1]$, geeignet für binäre Klassifikation
- **Tanh**: $f(x) = \tanh(x)$ – Ausgabe in $[-1, 1]$, gut für symmetrische Daten und Zeitreihen
- **Linear**: $f(x) = x$ – zwingend für kontinuierliche Regression im Output-Layer

Loss-Funktionen:

- **MSE** (*Mean Squared Error*): Standard für Regression, bestraft große Abweichungen stark
- **BCE** (*Binary Cross Entropy*): Standard für binäre Klassifikation (Output mit Sigmoid)

Beispieldatensätze im Rechner v20

Automatische Übernahme: Beim Laden eines Beispiels (Tab „Daten“) übernimmt der Rechner die in diesem Dokument empfohlenen Einstellungen und erstellt das Netzwerk. Weicht eine Einstellung später von der Empfehlung ab, zeigt der Tab „Konfiguration“ eine Soll/Ist-Tabelle mit Warnzeichen; die Schaltfläche „Empfohlene Einstellungen übernehmen“ stellt die Werte wieder her. Für eigene CSV-Dateien leitet der Rechner die Empfehlung aus den Daten ab (Faustregeln in Abschnitt 9).

Reproduzierbarkeit: Startwert (Seed) 42 und Mischen der Trainingsdaten je Epoche sind voreingestellt. Gleicher Startwert und gleiche Einstellungen liefern exakt dasselbe Ergebnis; die Zahlen in diesem Dokument wurden so ermittelt.

Kreuzkurse (UZS): 4 Inputs → EUR/UZS

Anwendungsfall: USD/UZS, EUR/USD, GBP/UZS, CNY/UZS als Inputs – EUR/UZS als Output. 251 Handelstage (03.11.2025–30.10.2026), synthetisch erzeugt, mit Datumsspalte.

Datenbeschreibung: Die Kurse schwanken um ein stabiles Niveau ohne starken Trend. Das Ziel folgt dem Kreuzkurs-Zusammenhang

$$\text{EUR/UZS} \approx \text{EUR/USD} \times \text{USD/UZS}$$

mit einem kleinen Rauschen (Standardabweichung 0,04 %). GBP/UZS und CNY/UZS haben keinen Einfluss auf das Ziel – das Netz muss die relevanten Inputs selbst finden. Die Datumsspalte wird beim Laden automatisch ignoriert.

Ziel des MLP: Das Netz soll den (fast) multiplikativen Zusammenhang aus den Daten lernen. Da das Ziel durch zwei Inputs bestimmt ist, ist eine deutlich bessere Prognose als mit einfachen Regeln möglich. Das Beispiel zeigt, wie ein *gut prognostizierbarer* Datensatz im Backtesting aussieht.

Parameter	Empfehlung (Voreinstellung)
Architektur	4 → 8–16 → 4–8 → 1 (voreingestellt: 4 → 12 → 6 → 1)
Aktivierung Hidden	ReLU
Aktivierung Output	**Linear**
Loss-Funktion	MSE
Lernrate	0,001 – 0,005 (0,003)
Epochen	2000 – 5000 (2000)
Batch-Größe	16 – 32 (16)
Normalisierung	**aktiv**
Train/Test-Split	80 / 20, chronologisch (letzte 51 Zeilen = Test)

Ergebnis im Backtesting (Voreinstellung, Startwert 42):

Verfahren (Testzeitraum, 51 Tage)	MAE	MAPE	R^2
MLP	5,34	0,037 %	0,996
Wert der Vorzeile („wie am Vortag“)	50,92	0,354 %	0,621
Letzter Wert vor dem Testzeitraum	80,15	0,558 %	−0,0005
Mittelwert der Trainingsdaten	85,88	0,597 %	−0,069

Das Netz ist deutlich besser als alle einfachen Regeln (RMSE rund 90 % unter der besten Regel). Training und Test liegen nahe beieinander (MAPE 0,033 % bzw. 0,037 %) – keine Überanpassung.

> **Extrapolation:** 27 der 51 Testzeilen haben mindestens einen Input außerhalb des Trainingsbereichs (USD/UZS bis 14,8 % der Bandbreite darunter, GBP/UZS bis 24,7 % darüber). Der Fehler dieser Zeilen ist trotzdem nicht größer (MAE 5,03 gegenüber 5,68), weil GBP/UZS für das Ziel keine Rolle spielt und der Zusammenhang mit USD/UZS knapp außerhalb noch annähernd linear ist.

Währungsdaten (UZS): 4 Inputs → EUR

Anwendungsfall: USD/UZS, CNY/UZS, GBP/UZS, CHF/UZS als Inputs – EUR/UZS als Output. 251 Zeilen (Datei `Währungsdaten_NN_4-1.csv`), kontinuierliche Werte, enge Wertebereiche.

Datenbeschreibung: Tageskurse von vier Währungen gegenüber dem usbekischen Som über rund ein Jahr. Die Werte liegen in engen Bandbreiten (z. B. USD: 12 394–12 624, CNY: 1 716–1 749). Die *Niveaus* der Kurse bewegen sich teilweise gemeinsam; die *Tagesveränderungen* von EUR und den vier Input-Kursen hängen jedoch praktisch nicht zusammen (Korrelation ≈ 0). Außerdem sind Wochenenden als dreifach wiederholte Zeilen enthalten.

Ziel des MLP: Das Netz soll aus vier Kursen den EUR/UZS-Kurs schätzen – ein *Regressionsproblem*. Wegen des fehlenden Zusammenhangs in den Tagesveränderungen ist der Datensatz vor allem ein *Lehrbeispiel*: Das Backtesting zeigt, dass eine niedrige Fehlerquote im Training noch keine brauchbare Prognose bedeutet.

Parameter	Empfehlung (Voreinstellung)
Architektur	4 → 8–16 → 8–16 → 1 (voreingestellt: 4 → 16 → 8 → 1)
Aktivierung Hidden	ReLU
Aktivierung Output	**Linear** (zwingend bei kontinuierlichem Output)
Loss-Funktion	MSE
Lernrate	0,001 – 0,005 (0,003)
Epochen	2000 – 5000 (3000)
Batch-Größe	16 – 32 (16)
Normalisierung	**immer aktiv** (Werte im Bereich 12 000–16 000)
Train/Test-Split	80 / 20, chronologisch

Ergebnis im Backtesting (Voreinstellung, Startwert 42):

Verfahren (Testzeitraum)	MAE	MAPE	R^2
MLP (4 Inputs)	75,43	0,557 %	−4,24
MLP mit Vortageswert als 5. Input*	16,23	0,120 %	0,638
Wert der Vorzeile („wie am Vortag“)	15,85	0,117 %	0,618
Letzter Wert vor dem Testzeitraum	31,04	0,230 %	−0,005
Mittelwert der Trainingsdaten	69,40	0,512 %	−3,52

* Testzeitraum 50 statt 51 Zeilen, da mit der Lag-Option die erste Zeile entfällt; die Regel „wie am Vortag“ erreicht dort MAE 15,32, MAPE 0,113 %, R^2 0,633.

Bewertung: Mit den vier Inputs ist das Netz schlechter als ein konstanter Wert ($R^2 < 0$); der Rechner meldet zusätzlich mögliche Überanpassung (Trainings-MAPE 0,255 %, Test-MAPE 0,557 %). Mit der Option „Vortageswert der Zielgröße als zusätzlichen Input“ (Tab „Daten“) erreicht das Netz etwa die Güte der Regel „wie am Vortag“ – mehr ist aus diesen Daten nicht herauszuholen. Ein MAPE von unter 1 % ist bei Wechselkursen also *kein* Gütezeichen; entscheidend ist der Vergleich mit den einfachen Regeln.

> **Hinweis:** Der Tab „Vorhersage“ zeigt zu jedem Input den Trainingsbereich und warnt bei Extrapolation mit Angabe der Abweichung (absolut und in Prozent der Bandbreite). Im Testzeitraum liegt GBP/UZS in 18 Zeilen bis zu 14,5 % über dem Trainingsmaximum.

Logische Gatter: XOR, AND, OR

Klassische Lehrbeispiele für binäre Klassifikation mit 2 Inputs und 4 Samples.

Datenbeschreibung: Wahrheitstabellen der booleschen Logik. Zwei binäre Inputs (0 oder 1) und ein binärer Output. AND liefert 1 nur wenn beide Inputs 1 sind; OR liefert 1 wenn mindestens ein Input 1 ist; XOR liefert 1 wenn genau ein Input 1 ist. Der Datensatz umfasst alle vier möglichen Eingabekombinationen.

Ziel des MLP: Das Netz soll die logische Funktion erlernen. XOR ist dabei besonders bedeutsam: Es ist *nicht linear separierbar* und kann deshalb nicht durch ein Netz ohne Hidden Layer gelöst werden. Das Beispiel demonstriert anschaulich, warum mehrschichtige Netze notwendig sind – ein historisch wichtiges Ergebnis der KI-Forschung.

Parameter	Empfehlung
Architektur	$2 \to 4$–$8 \to 4$–$8 \to 1$ (der Rechner verlangt mindestens 2 Hidden Layers)
Aktivierung Hidden	Sigmoid
Aktivierung Output	Sigmoid
Loss-Funktion	BCE
Lernrate	0,1
Epochen	5000 – 10000 (XOR benötigt mehr)
Batch-Größe	1 – 4 (alle Samples)
Normalisierung	aus (Daten bereits 0/1)

Hinweis zu XOR: XOR ist nicht linear separierbar – mindestens 1 Hidden Layer mit ≥ 4 Neuronen ist zwingend erforderlich. AND und OR sind linear separierbar und könnten theoretisch ohne Hidden Layer gelöst werden.

Kreis-Klassifikation (200 Samples)

200 zufällige Punkte in $[-1, 1]^2$*. Output 1 wenn innerhalb Kreis mit Radius* $\sqrt{0{,}5}$*, sonst 0.*

Datenbeschreibung: 200 zufällig erzeugte Punkte im Quadrat $[-1, 1]^2$. Ein Punkt gehört zur Klasse 1 (innerhalb), wenn $x^2 + y^2 < 0{,}5$, andernfalls zur Klasse 0 (außerhalb). Die Entscheidungsgrenze ist eine Kreislinie – eine typisch nichtlineare Struktur, die lineare Klassifikatoren nicht erlernen können.

Ziel des MLP: Das Netz soll eine nichtlineare Entscheidungsgrenze im zweidimensionalen Raum erlernen. Dies ist ein *binäres Klassifikationsproblem* und zeigt, wie MLPs beliebig geformte Trennflächen approximieren können. Gut geeignet zum Experimentieren mit Netztiefe und Neuronenanzahl.

Parameter	Empfehlung
Architektur	$2 \to 8$–$16 \to 8 \to 1$
Aktivierung Hidden	ReLU
Aktivierung Output	Sigmoid
Loss-Funktion	BCE
Lernrate	0,01
Epochen	2000 – 3000
Batch-Größe	16 – 32
Normalisierung	aus (Daten in $[-1, 1]$)
Train/Test-Split	80 / 20

Einfache Regression – Sinuskurve (50 Samples)

$y = 0{,}5 + 0{,}4 \cdot \sin(6x) + \varepsilon$*, mit* $x \in [0, 1]$ *und Rauschen* ε*.*

Datenbeschreibung: 50 gleichmäßig verteilte Punkte entlang einer gedämpften Sinuskurve mit überlagerten Zufallsstörungen. Der Input x läuft von 0 bis 1, der Output y schwankt zwischen ca. 0,1 und 0,9. Die Kurve enthält mehrere Perioden und testet die Fähigkeit des Netzes, periodische Strukturen zu erfassen.

Ziel des MLP: Das Netz soll eine kontinuierliche nichtlineare Funktion aus verrauschten Stichproben approximieren – ein klassisches *Regressionsproblem*. Zeigt den Unterschied zwischen Unter- und Überanpassung: Zu wenige Neuronen können die Kurve nicht abbilden; zu viele lernen das Rauschen auswendig.

Parameter	Empfehlung
Architektur	$1 \rightarrow 8\text{–}16 \rightarrow 8 \rightarrow 1$
Aktivierung Hidden	Tanh (symmetrisch, gut für Sinusfunktion)
Aktivierung Output	Linear
Loss-Funktion	MSE
Lernrate	0,01
Epochen	3000 – 5000
Batch-Größe	8 – 16
Normalisierung	aus (Werte in $[0, 1]$)

Weitere typische Anwendungsfälle

Iris-Blumenklassifikation (3 Klassen)

4 botanische Merkmale → 3 Klassen (Setosa, Versicolor, Virginica). 150 Samples.

Datenbeschreibung: 150 Schwertlilienblüten mit je 4 Merkmalen: Kelchblattlänge/-breite und Blütenblattlänge/-breite (in cm). Drei Klassen zu je 50 Samples. Setosa ist von den anderen zwei gut trennbar; Versicolor und Virginica überlappen teilweise.

Ziel des MLP: Mehrklassen-Klassifikation anhand botanischer Merkmale. Da der Rechner nur ein Ausgabeneuron und keinen Softmax-Output hat, wird je Klasse ein eigenes Netz trainiert („eine Klasse gegen den Rest“, Zielspalte 0/1); vorhergesagt wird die Klasse mit dem höchsten Output. Klassisches Benchmark-Dataset zum Vergleich von Klassifikationsalgorithmen.

Parameter	Empfehlung
Architektur	$4 \rightarrow 8\text{–}16 \rightarrow 4\text{–}8 \rightarrow 1$ (je Klasse)
Aktivierung Hidden	ReLU
Aktivierung Output	Sigmoid (Softmax nicht verfügbar)
Loss-Funktion	BCE
Lernrate	0,01
Epochen	2000 – 4000
Batch-Größe	16 – 32
Normalisierung	aktiv

Zeitreihen-Regression (z. B. Temperatur, Aktienkurse)

Mehrere zeitlich versetzte Werte als Input → nächster Wert als Output.

Datenbeschreibung: 366 Tagestemperaturen mit jahreszeitsabhängigem Verlauf (Sinus-Jahresgang ± Rauschen). Als Inputs dienen die Temperaturen der letzten 3 Tage, als Output die Temperatur des Folgetages. Die Werte liegen im Bereich ca. -3 bis $+23\,°\mathrm{C}$.

Ziel des MLP: Vorhersage des nächsten Wertes einer Zeitreihe aus vergangenen Beobachtungen – sogenanntes *Autoregression*. Zeigt, wie ein MLP ohne explizites Gedächtnis kurzfristige

Abhängigkeiten in Zeitreihen nutzen kann. Übertragbar auf Aktienkurse, Energieverbrauch oder Sensordaten.

Parameter	Empfehlung
Architektur	n → 16–32 → 16 → 1
Aktivierung Hidden	Tanh oder ReLU
Aktivierung Output	Linear
Loss-Funktion	MSE
Lernrate	0,001 – 0,005
Epochen	3000 – 8000
Batch-Größe	16 – 64
Normalisierung	**immer aktiv**

Hauspreisvorhersage (mehrere Merkmale)

Fläche, Zimmer, Lage, Baujahr usw. → Preis. 500 Samples, 4 Inputs.

Datenbeschreibung: 500 synthetische Immobiliendaten mit den Merkmalen Wohnfläche (40–200 m²), Zimmeranzahl (1–6), Baujahr (1950–2023) und Entfernung zum Stadtzentrum (1–30 km). Der Preis wurde als gewichtete Linearkombination dieser Faktoren mit Rauschen erzeugt (realistisch: 50 000–650 000 EUR).

Ziel des MLP: Schätzung eines kontinuierlichen Immobilienwertes aus mehreren heterogenen Merkmalen – ein typisches *Regressionsproblem* mit praktischer Relevanz. Demonstriert den Umgang mit unterschiedlich skalierten Inputs (daher Normalisierung zwingend) und die Wichtigkeit von Train/Test-Trennung zur Bewertung der Generalisierung.

Parameter	Empfehlung
Architektur	n → 32–64 → 16–32 → 1
Aktivierung Hidden	ReLU
Aktivierung Output	Linear
Loss-Funktion	MSE
Lernrate	0,001
Epochen	3000 – 10000
Batch-Größe	32 – 64
Normalisierung	**immer aktiv**
Train/Test-Split	80 / 20

Spam-Erkennung / Binäre Textklassifikation

Merkmalsvektoren aus Text (z. B. Worthäufigkeiten) → Spam (1) / Kein Spam (0). 400 Samples, 5 Inputs.

Datenbeschreibung: 400 synthetische E-Mail-Repräsentationen mit 5 normierten Merkmalen: Häufigkeit der Wörter *gratis*, *gewinn*, *klicken*, Anzahl der Links und Häufigkeit von Ausrufezeichen. Spam-E-Mails (200 Samples) zeigen deutlich höhere Werte in allen Merkmalen als normale E-Mails (200 Samples). Ausgeglichene Klassenverteilung (50/50).

Ziel des MLP: Binäre Klassifikation zur automatischen Spam-Erkennung – ein *Klassifikationsproblem* mit direktem praktischem Nutzen. Zeigt, wie ein MLP aus einfachen Textmerkmalen

eine zuverlässige Entscheidungsregel erlernen kann. Übertragbar auf andere binäre Klassifikationsaufgaben wie Betrugserkennung oder medizinische Diagnose.

Parameter	Empfehlung
Architektur	n → 32–64 → 16 → 1
Aktivierung Hidden	ReLU
Aktivierung Output	Sigmoid
Loss-Funktion	BCE
Lernrate	0,001 – 0,01
Epochen	2000 – 5000
Batch-Größe	32 – 64
Normalisierung	aktiv (TF-IDF-Werte normieren)
Train/Test-Split	80 / 20

Allgemeine Faustregeln

Situation	Empfehlung
Loss divergiert (NaN)	Lernrate halbieren, Normalisierung prüfen
Loss stagniert früh	Lernrate leicht erhöhen, mehr Neuronen
Loss fällt, aber Test-Loss steigt	Überanpassung – weniger Neuronen oder mehr Daten
Output immer nahe 0 oder 1	Falscher Output-Aktivierungsfunktion (Linear statt Sigmoid?)
Sehr langsame Konvergenz	Xavier-Initialisierung prüfen (im Rechner bereits aktiv)
Kleine Datenmenge (<100)	Kleine Batch-Größe (4–8), mehr Epochen
Netz nicht besser als „wie am Vortag“	Zeitreihe: Vortageswert als Input aktivieren; Inputs mit echtem Zusammenhang zum Ziel suchen
Viele Testzeilen mit „Extrapolation“	Längeren Trainingszeitraum wählen oder Fehler der markierten Zeilen prüfen
Eigene Daten: Startwerte	Hidden 1 = 2–4 × Inputs, Hidden 2 ≈ halb so groß; Output Linear (Regression) bzw. Sigmoid (Ziel nur 0/1)

Xavier-Initialisierung und Startwert (Rechner v20): Die Startgewichte werden gleichverteilt in $\pm\sqrt{1/n_{\text{Inputs}}}$ gezogen. Das verhindert das Explodieren oder Verschwinden von Gradienten beim Trainingsstart. Die Zufallszahlen stammen aus einem Generator mit festem Startwert (Seed, Tab „Konfiguration“); damit sind Startgewichte und Mischreihenfolge – und so das gesamte Ergebnis – reproduzierbar.