Ja oder Nein? (Binäre Klassifikation)

Ja/Nein-Entscheidungen lernen — und sehen, wie sicher sich das Netz ist: Sigmoid, Entscheidungsgrenzen, Genauigkeit.

Fortgeschritten · Kapitel 21 von 50 · ca. 25 Min. · 250 XP

Kapitel online

Einfach gesagt: Dein Mailprogramm sortiert Spam aus, ohne feste Regeln — es presst jede Mail in eine Zahl zwischen 0 und 1 und entscheidet ab 0,5.

Heute Morgen, ohne dass du es gemerkt hast: Dein Mail-Programm hat 14 Mails geprüft und 3 davon in den Spam-Ordner sortiert. Keine Regel-Liste, kein Mensch — ein Modell hat für jede Mail eine **Ja/Nein-Entscheidung** getroffen. Genau das baust du jetzt selbst nach: **binäre Klassifikation** — die Kunst, aus Zahlen eine Entscheidung zu machen.

Was du hier lernst

- Warum „Spam oder nicht?“ eine ganz andere Aufgabe ist als „Wie viel?“ (Regression)
- Wie **Sigmoid** aus jeder Zahl eine **Wahrscheinlichkeit zwischen 0 und 1** macht
- Wie aus der Wahrscheinlichkeit per **Schwelle (0,5)** eine Entscheidung wird
- Womit man Entscheidungs-Fehler misst (**Binary Crossentropy**) — und wie du Treffer von Fehlalarmen unterscheidest

Für den Unterricht

Einstieg (5 Min): Kapitel gemeinsam lesen, Alltagsbeispiel aufgreifen.

Erkunden am Gerät (15 Min): die interaktiven Module auf herrki.de/220_binaere_klassifikation ausprobieren.

Sichern mit Quiz (5 Min): „Kurzer Check" besprechen, „Das nimmst du mit" festhalten.

Runter zum Tensor: Was sieht das Netz von einer E-Mail?

Ein Netz liest keine Mails — es rechnet (du kennst das aus „Excel/CSV → Tensoren“). Jede Mail wird deshalb zu einem kleinen **Feature-Tensor** aus 5 Zahlen. Nehmen wir Mail Nr. 13 aus der Trainings-Tabelle weiter unten — kurz, 8 Links, „dringend“, Geld-Symbol, 45 % Großbuchstaben:

[5] `[50, 8, 1, 1, 45]` → Wörter, Links, „dringend“?, Geld-Symbole?, Großbuchstaben-%

↓ Netz rechnet ↓

[1] `[0.995]` → **ein einziger Ausgabe-Tensor: 99,5 % Spam-Wahrscheinlichkeit**

Dazwischen passiert erstaunlich wenig: Jedes der 5 Merkmale bekommt ein **Gewicht** — wie wichtig ist es, und in welche Richtung? Alles wird aufaddiert, ein **Bias** dazu, fertig ist die rohe Ausgabe. Mit Gewichten, wie sie nach dem Training herauskommen könnten:

```
Gewichte [5]: [−0,02 +0,35 +1,20 +1,10 +0,06] Bias: −1,50

Mail 13, Spam [ 50, 8, 1, 1, 45]:
  Wörter −0,02 · 50 = −1,00 // viele Wörter sprechen GEGEN Spam
  Links +0,35 · 8 = +2,80
  dringend +1,20 · 1 = +1,20
  Geld +1,10 · 1 = +1,10
  CAPS +0,06 · 45 = +2,70
  Summe + Bias (−1,50) = +5,30 // rohe Ausgabe – jede Zahl wäre möglich
  sigmoid(+5,30) = 0,995 // über 0,5 -> SPAM

Mail 1, legitim [120, 1, 0, 0, 2]:
  −0,02·120 +0,35·1 +0 +0 +0,06·2 = −1,93
  Summe + Bias (−1,50) = −3,43
  sigmoid(−3,43) = 0,031 // unter 0,5 -> LEGITIM
```

Dieselbe Rechnung, zwei völlig verschiedene Antworten — allein wegen der Zahlen im Tensor. (Das Modul unten skaliert die 5 Merkmale vorher noch per Z-Score aus Kapitel 135 und rechnet mit 8 Neuronen in der Mitte statt einem. Am Prinzip ändert das nichts: multiplizieren, addieren, quetschen.)

Der Trick heißt Sigmoid: Das letzte Neuron könnte jede Zahl ausspucken (bei uns +5,30 und −3,43; genauso gut −7,3 oder 42). Die Sigmoid-Funktion *quetscht* jede Zahl sanft in den Bereich 0 bis 1 — aus +5,30 wird 0,995, aus −3,43 wird 0,031 — und plötzlich lässt sich das Ergebnis als Wahrscheinlichkeit lesen. Über 0,5? → „Ja, Spam.“ Darunter? → „Nein.“ Aus Rechnen wird Entscheiden.

Trainingsdaten

> Das Netz lernt aus **24 Beispiel-E-Mails** (12 normale, 12 Spam) — schau dir die Muster an, *bevor* du trainierst: Erkennst DU, woran man den Spam erkennt?

Nr.	Wörter	Links	⚠ Dringend?	Geld?	CAPS %	Kategorie

Training starten

Hyperparameter

Dieselben Stellschrauben wie bei der Regression — nur das Ziel ist neu: statt einer Zahl soll am Ende eine gute Entscheidung herauskommen.

Neu ist damit auch der **Fehler-Messer**. Dein Netz gibt ja keine Zahl aus, sondern eine Wahrscheinlichkeit p für „Spam“ — das Ergebnis des Sigmoids von oben. Der passende Fehler dazu heißt **Binary Cross-Entropy** (binäre Kreuzentropie), und ihre Idee ist erstaunlich schlicht: Sie schaut nur auf die Wahrscheinlichkeit, die das Netz der *richtigen* Antwort gegeben hat, und rechnet **−log(p)**. Je weniger Wahrscheinlichkeit auf die Wahrheit entfiel, desto größer die Strafe. War die Mail wirklich Spam und dein Netz sagt 0,9, kostet das fast nichts; bei 0,5 — reines Raten — schon das Sechsfache, bei 0,1 das Zwanzigfache:

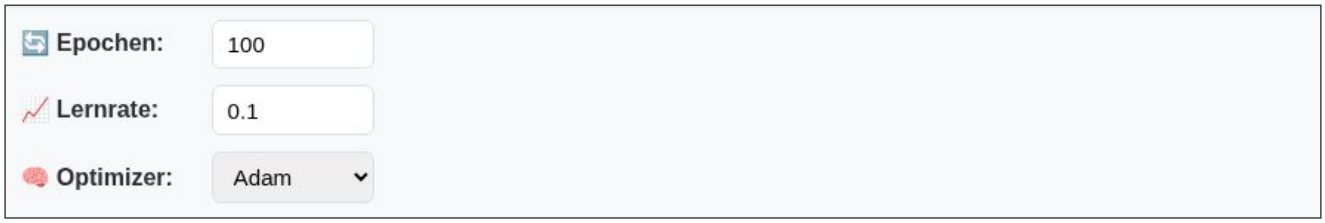
```
Richtige Antwort: SPAM Fehler = −ln(p)

p = 0,90 → −ln(0,90) = 0,105 // fast richtig, fast gratis
p = 0,50 → −ln(0,50) = 0,693 // reines Raten
p = 0,10 → −ln(0,10) = 2,303 // daneben – und teuer
p = 0,01 → −ln(0,01) = 4,605 // „99 % sicher“ und trotzdem falsch: brutal
```

Und warum nicht der MSE aus „Wie falsch ist falsch?“? Weil der bei selbstsicheren Fehlern viel zu milde ist: Zwischen 0 und 1 kann er sich höchstens um 1 verrechnen (Ziel 0, Ausgabe 1 → Fehler $1^2 = 1$), während die Cross-Entropy denselben Patzer mit 4,6 ahndet — und noch härter, je dreister das Netz wird. Dazu kommt ein handfester Rechen-Vorteil: Ist das Sigmoid schon fast gesättigt (Ausgabe dicht an 0 oder 1), schrumpft der Gradient — der Wegweiser, an dem das Netz seine Gewichte ausrichtet — beim MSE auf fast nichts, und das Lernen bleibt genau dort stehen, wo es am nötigsten wäre. Bei der Cross-Entropy bleibt er groß. Das Netz lernt also aus genau den Fehlern, die weh tun. Im Modul unten steht dafür eine einzige Zeile: `loss: 'binaryCrossentropy'` — und sobald es mehr als zwei Klassen zu unterscheiden gibt, wird daraus im nächsten Kapitel die kategoriale Variante.

Am Gerät ausprobieren: herrki.de/220_binaere_klassifikation#abschnitt-4

Direkt zum Modul



Interaktives Modul – am Gerät ausprobieren

0%

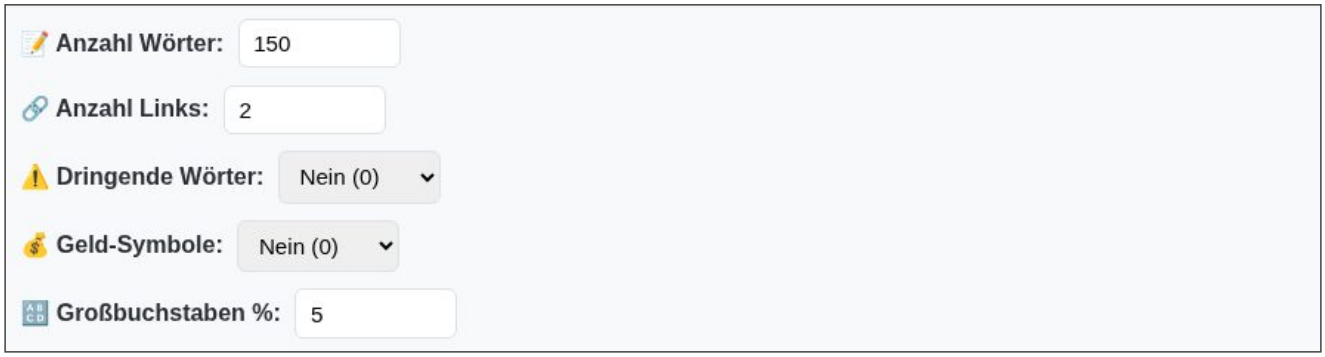
Beim Deuten der Ergebnisse gilt: Nicht jeder Fehler ist gleich schlimm. Eine Spam-Mail, die durchrutscht, nervt — eine *echte* Mail, die im Spam-Ordner landet (ein „Fehlalarm"), kann dich ein Jobangebot kosten. Profis zählen deshalb Treffer und Fehlalarme getrennt, statt nur auf „Genauigkeit" zu schauen. Wie man sauber prüft, ob das Modell wirklich kann, was es verspricht, vertiefst du in „Trainieren, prüfen, bestehen".

E-Mails testen

Jetzt bist du der Spam-Versender — oder der ehrliche Absender: Stell dir eine Mail zusammen und schau, ob dein Netz sie durchlässt. Probier beides: eine harmlose Alltagsmail (viele Wörter, kein Geld, wenig CAPS) und eine typische „GEWINNE JETZT!!!“-Mail.

Am Gerät ausprobieren: herrki.de/220_binaere_klassifikation#abschnitt-5

Direkt zum Modul



Interaktives Modul – am Gerät ausprobieren

Zurück zur Anwendung

Erinnerst du dich an die 14 Mails von heute Morgen, von denen drei im Spam-Ordner landeten? Jetzt weißt du, was dabei passiert ist: Jede Mail wurde zu einem kleinen Zahlen-Tensor, daraus wurde eine Wahrscheinlichkeit, und eine Schwelle machte daraus die Entscheidung. Genau nach diesem Muster arbeiten auch die Ja/Nein-Fragen, die dich sonst so treffen — Kredit bewilligt oder nicht, Bild freigegeben oder gemeldet, Termin wahrgenommen oder gerissen. Und weil du jetzt weißt, dass hinter „Spam“ eine Zahl wie 0,92 steckt, kannst du die richtige Frage stellen: *Wo liegt die Schwelle — und wer bezahlt die Fehlalarme?* Beim nächsten Griff in den Spam-Ordner siehst du kein Rätsel mehr, sondern eine bewusst gesetzte Grenze.

Zwei Möglichkeiten waren der Anfang. Was passiert, wenn es Apfel, Banane *oder* Orange heißt, siehst du im nächsten Kapitel: „Multi-Class Klassifikation“.

Zum Weiterlesen

- **Sigmoid, Schwelle, Metriken — kompakt erklärt:** Google Machine Learning Crash Course, *Logistic Regression* — developers.google.com → Logistic Regression (developers.google.com/machine-learning/crash-course/logistic-regression)
- **Die statistische Sicht auf Klassifikation:** James, Witten, Hastie, Tibshirani, *An Introduction to Statistical Learning*, Kapitel 4 (PDF frei) — statlearning.com
- **Warum Cross-Entropy und nicht MSE:** Michael Nielsen, *Neural Networks and Deep Learning*, Kapitel 3 — neuralnetworksanddeeplearning.com/chap3.html

Kurzer Check – mit Lösung

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

Frage 1: Die Sigmoid-Ausgabe für eine E-Mail ist 0,92. Was heißt das?

- Die E-Mail enthält 92 verdächtige Wörter.
- ✔ **Das Modell schätzt 92 % Wahrscheinlichkeit für Spam — mit Schwelle 0,5 wird sie als Spam einsortiert.**
- Die E-Mail wird zu 92 % gelöscht und zu 8 % zugestellt.

Sigmoid quetscht die rohe Netz-Ausgabe auf 0–1, damit du sie als Wahrscheinlichkeit lesen kannst. 0,92 heißt „ziemlich sicher Spam" — nicht „92 Wörter" und nicht „teilweise löschen".

Frage 2: Warum steht am Ausgang überhaupt ein Sigmoid und nicht einfach die rohe Zahl?

- Ohne Sigmoid könnte das Netz gar nicht trainieren.
- ✔ **Sigmoid presst jede beliebige Zahl in den Bereich 0–1 — erst dadurch wird daraus eine interpretierbare Ja/Nein-Wahrscheinlichkeit.**
- Sigmoid macht das Training zehnmal schneller.

Die rohe Ausgabe könnte −7 oder +42 sein — schwer zu deuten. Nach dem Sigmoid ist es eine Wahrscheinlichkeit, und die Schwelle (z. B. 0,5) macht daraus die Entscheidung.

Frage 3: Eine wichtige Bewerbung landet im Spam-Ordner. Welcher Fehlertyp ist das — und warum ist das mehr als eine Statistik-Frage?

- Ein Treffer — das Modell hat ja etwas gefunden.
- ✔ **Ein Fehlalarm (falsch positiv) — und weil seine Kosten hoch sind, wählt man die Schwelle je nach Anwendung bewusst.**
- Egal welcher Typ — Hauptsache, die Gesamt-Genauigkeit stimmt.

Verpasste Bewerbung vs. eine Spam-Mail im Posteingang: Die beiden Fehlerarten kosten unterschiedlich viel! Deshalb schaut man nie nur auf Genauigkeit, sondern auf Treffer UND Fehlalarme — und schiebt die Schwelle entsprechend.

Das nimmst du mit

- **Entscheiden ist etwas anderes als schätzen.** Nicht „wie viel?“, sondern „ja oder nein?“ — dieselben Tensoren, ein anderes Ziel.
- **Sigmoid quetscht jede Zahl in 0–1.** Aus +5,30 wird 0,995, aus −3,43 wird 0,031 — erst dadurch lässt sich die Ausgabe als Wahrscheinlichkeit lesen.
- **Die Schwelle macht die Entscheidung.** Über 0,5 heißt „Spam“ — und diese Grenze ist eine bewusste Wahl, kein Naturgesetz.
- **Binary Crossentropy bestraft selbstsichere Fehler.** „99 % sicher“ und trotzdem falsch kostet 4,605 — genau daraus lernt das Netz am meisten.
- **Nicht jeder Fehler kostet gleich viel.** Eine durchgerutschte Spam-Mail nervt, eine verlorene Bewerbung schmerzt — deshalb zählt man Treffer und Fehlalarme getrennt.

Und wenn es mehr als zwei Möglichkeiten gibt? Spam/kein Spam war einfach — aber deine Foto-App unterscheidet Katze, Hund, Essen, Strand … Dafür braucht es einen neuen Baustein: Softmax — im nächsten Kapitel.

So zitierst du dieses Kapitel

HerrKI: Ja oder Nein? (Binäre Klassifikation). herrki.de/220_binaere_klassifikation, ein privates Hobbyprojekt von Terragon.de, abgerufen am 09.09.2026.