Lern-Nugget: Sigmoid als Spezialfall von Softmax

🎯 Ziel

Den eleganten Zusammenhang zwischen Sigmoid und Softmax verstehen - warum Sigmoid nur ein Spezialfall von Softmax für binäre Klassifikation ist!

🤔 Das Problem: Zwei scheinbar verschiedene Funktionen

Sigmoid (binäre Klassifikation):

- $\sigma(z) = 1/(1 + e^{(-z)})$
- Ausgabe: Eine Wahrscheinlichkeit zwischen 0 und 1
- Verwendung: "Hund oder Katze?", "Spam oder nicht?"

Softmax (Multi-Klassen-Klassifikation):

- $\text{softmax}(z_i) = e^{(z_i)} / \Sigma(e^{(z_j)})$
- Ausgabe: Wahrscheinlichkeitsverteilung über alle Klassen
- Verwendung: "Hund, Katze, oder Vogel?", "0, 1, 2, ..., 9?"

Frage: Sind das wirklich verschiedene Funktionen? 🤔

🔍 Der Schlüssel: Softmax mit 2 Klassen

Gegeben: Zwei Klassen mit Logits z_1 und z_2

Softmax berechnen:

- $P(\text{Klasse } 1) = e^{(z_1)} / (e^{(z_1)} + e^{(z_2)})$
- $P(\text{Klasse } 2) = e^{(z_2)} / (e^{(z_1)} + e^{(z_2)})$

Schritt 1: Trick anwenden

Dividiere Zähler und Nenner durch $e^{(z_2)}$:

$P(\text{Klasse } 1) = e^{(z_1)} / (e^{(z_1)} + e^{(z_2)}) = (e^{(z_1)}/e^{(z_2)}) / ((e^{(z_1)} + e^{(z_2)})/e^{(z_2)}) = e^{(z_1-z_2)} / (e^{(z_1-z_2)} + 1)$

Definiere: $z = z_1 - z_2$ (der "relative" Logit)

$P(\text{Klasse } 1) = e^z / (e^z + 1) = e^z / (e^z + 1) \cdot e^{(-z)}/e^{(-z)} = 1 / (1 + e^{(-z)})$

🎉 Überraschung: Das ist Sigmoid!

$P(\text{Klasse } 1) = \sigma(z_1 - z_2) = 1/(1 + e^{(-(z_1-z_2))})$

Was bedeutet das?

1. **Sigmoid ist Softmax für 2 Klassen!**
2. **$z = z_1 - z_2$** ist der entscheidende "Unterschied" zwischen den Klassen

3. **P(Klasse 2) = 1 - P(Klasse 1)** automatisch erfüllt

📊 Konkrete Zahlenbeispiele

Beispiel 1: Klare Entscheidung

Logits: $z_1 = 2$, $z_2 = -1$

- **Softmax:**
  - P(Klasse 1) = $e^2/(e^2 + e^{-1})$ = 7.39/(7.39 + 0.37) ≈ 0.95
  - P(Klasse 2) = $e^{-1}/(e^2 + e^{-1})$ = 0.37/(7.39 + 0.37) ≈ 0.05
- **Sigmoid:** $\sigma(2-(-1)) = \sigma(3) = 1/(1 + e^{-3}) \approx 0.95$ ✓

Beispiel 2: Unklare Entscheidung

Logits: $z_1 = 0.5$, $z_2 = 0.3$

- **Softmax:**
  - P(Klasse 1) = e^0.5/(e^0.5 + e^0.3) = 1.65/(1.65 + 1.35) ≈ 0.55
  - P(Klasse 2) = e^0.3/(e^0.5 + e^0.3) = 1.35/(1.65 + 1.35) ≈ 0.45
- **Sigmoid:** $\sigma(0.5-0.3) = \sigma(0.2) = 1/(1 + e^{-0.2}) \approx 0.55$ ✓

🧠 Intuitive Interpretation

Was sagt $z = z_1 - z_2$?

z > 0: Klasse 1 ist "stärker" → P(Klasse 1) > 0.5 **z < 0:** Klasse 2 ist "stärker" → P(Klasse 1) < 0.5 **z = 0:** Beide gleich stark → P(Klasse 1) = 0.5

Praktische Bedeutung:

Bei binärer Klassifikation brauchen wir nur einen Logit!

- Statt z_1 und z_2 zu berechnen
- Berechnen wir direkt z = "Evidenz für Klasse 1 vs. Klasse 2"
- Sigmoid gibt uns P(Klasse 1) = σ(z)
- P(Klasse 2) = 1 - σ(z) ergibt sich automatisch

🏗 Implementierung: Von Softmax zu Sigmoid

Multi-Klassen Netzwerk:

```
Input → Hidden → [z₁, z₂, z₃, z₄] → Softmax → [p₁, p₂, p₃, p₄]
```

Binäres Netzwerk (ineffizient):

```
Input → Hidden → [z₁, z₂] → Softmax → [p₁, p₂]
```

Binäres Netzwerk (elegant):

```
Input → Hidden → z → Sigmoid → p₁
                              (p₂ = 1-p₁)
```

🎯 Wichtige Erkenntnisse

1. **Mathematische Eleganz**

- Sigmoid ist nicht "anders" als Softmax
- Es ist die **optimierte Version** für den 2-Klassen-Fall
- Ein Parameter weniger (z statt z_1, z_2)

2. **Praktische Konsequenzen**

- **Binäre Klassifikation:** Verwende Sigmoid (effizienter)
- **Multi-Klassen:** Verwende Softmax (notwendig)
- **Übergang:** Von binär zu multi-class ist nahtlos

3. **Loss-Funktionen**

- **Binary Cross-Entropy:** -[y log σ(z) + (1-y) log(1-σ(z))]
- **Categorical Cross-Entropy:** -Σ y_i log(softmax_i(z))
- Für 2 Klassen sind beide **identisch**!

🎲 Übung: Verständnis testen

Gegeben: Ein Netzwerk klassifiziert E-Mails als "Spam" oder "Ham"

Szenario A: Zwei Output-Neuronen mit Softmax

- Output: [2.1, -0.3] → Softmax → [0.89, 0.11]

Szenario B: Ein Output-Neuron mit Sigmoid

- Output: z = ? → Sigmoid → 0.89

Frage: Welcher Wert z ergibt sich in Szenario B?

Lösung:

- z = z_spam - z_ham = 2.1 - (-0.3) = 2.4
- σ(2.4) = 1/(1 + e^(-2.4)) ≈ 0.89 ✓

📝 Zusammenfassung

Kernbotschaft:

> Sigmoid ist Softmax für 2 Klassen. Statt zwei Logits z_1, z_2 zu berechnen, berechnen wir einen "relativen" Logit $z = z_1 - z_2$ und wenden Sigmoid darauf an.

Praktisch bedeutet das:

- ☑ **Effizienz:** Ein Parameter weniger bei binärer Klassifikation
- ☑ **Klarheit:** Der Output z hat direkte Interpretation (Evidenz-Unterschied)
- ☑ **Flexibilität:** Nahtloser Übergang zwischen binär und multi-class
- ☑ **Verständnis:** Beide Funktionen sind Teil derselben mathematischen Familie

---

💡 **Tipp:** Wenn Sie Sigmoid verstehen, verstehen Sie auch Softmax - und umgekehrt! Sie sind verschiedene Perspektiven auf dasselbe fundamentale Konzept der Wahrscheinlichkeitsmodellierung.