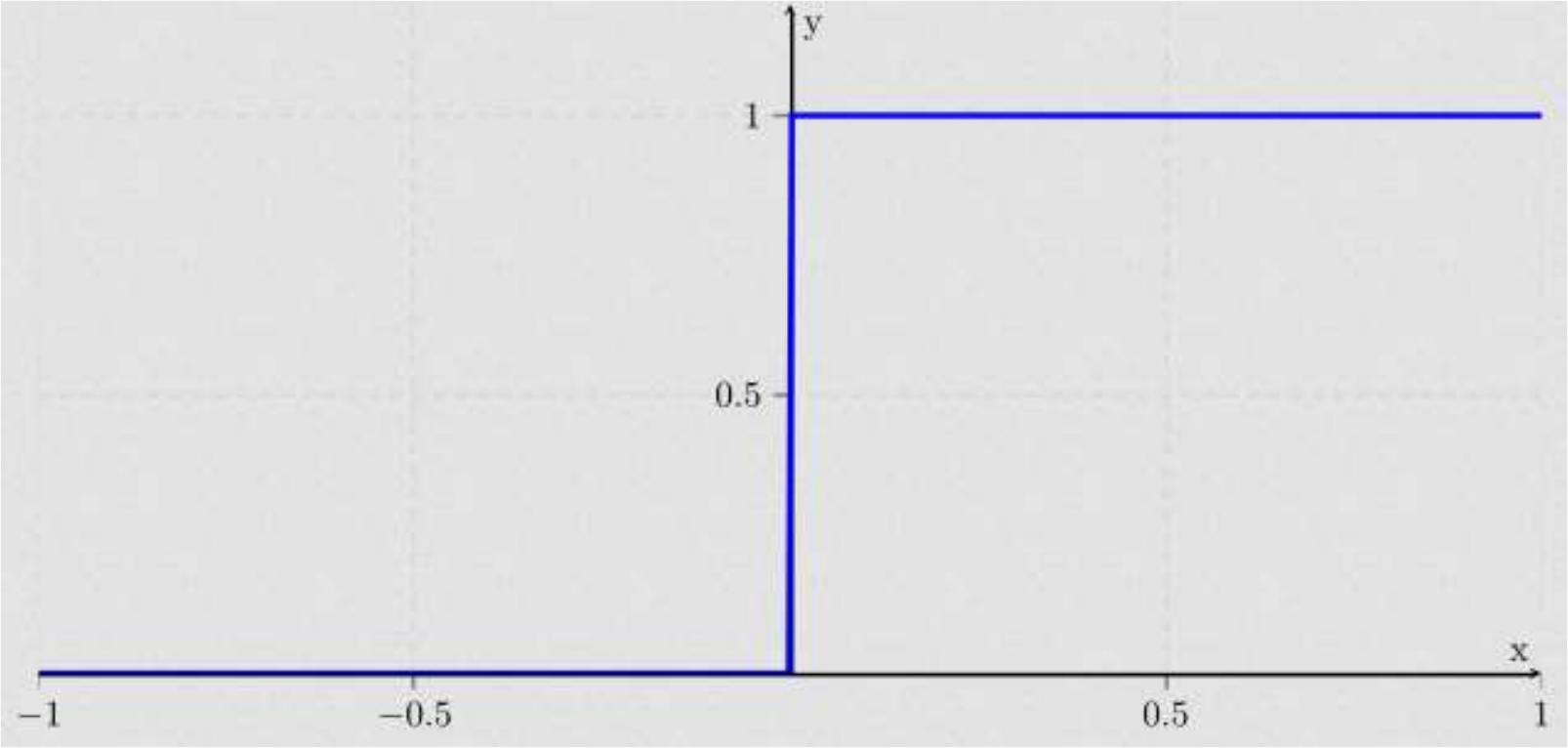
Aktivierungsfunktion - Binäre Step Funktion

Bisher: **Binäre Step Funktion**



Quelle: MartinThoma, CC BY 3.0, via Wikimedia Commons

"Ganz oder gar nicht" $\rightarrow$ Kennt als Output nur 0 oder 1

Aktivierungsfunktion - Binäre Step Funktion

Sinnvoll nutzbar bei Funktionen die als Ergebnis nur ein klares Wahr/Falsch kennen

Aktivierungsfunktion - Binäre Step Funktion

Sinnvoll nutzbar bei Funktionen die als Ergebnis nur ein klares Wahr/Falsch kennen

Nachteile:

- Schwaches aber korrektes Input-Signal wird u. U. nicht beachtet, wenn der Output z. B. 0 bleibt, anstatt wenigstens 0,3 anzuzeigen.

Aktivierungsfunktion - Binäre Step Funktion

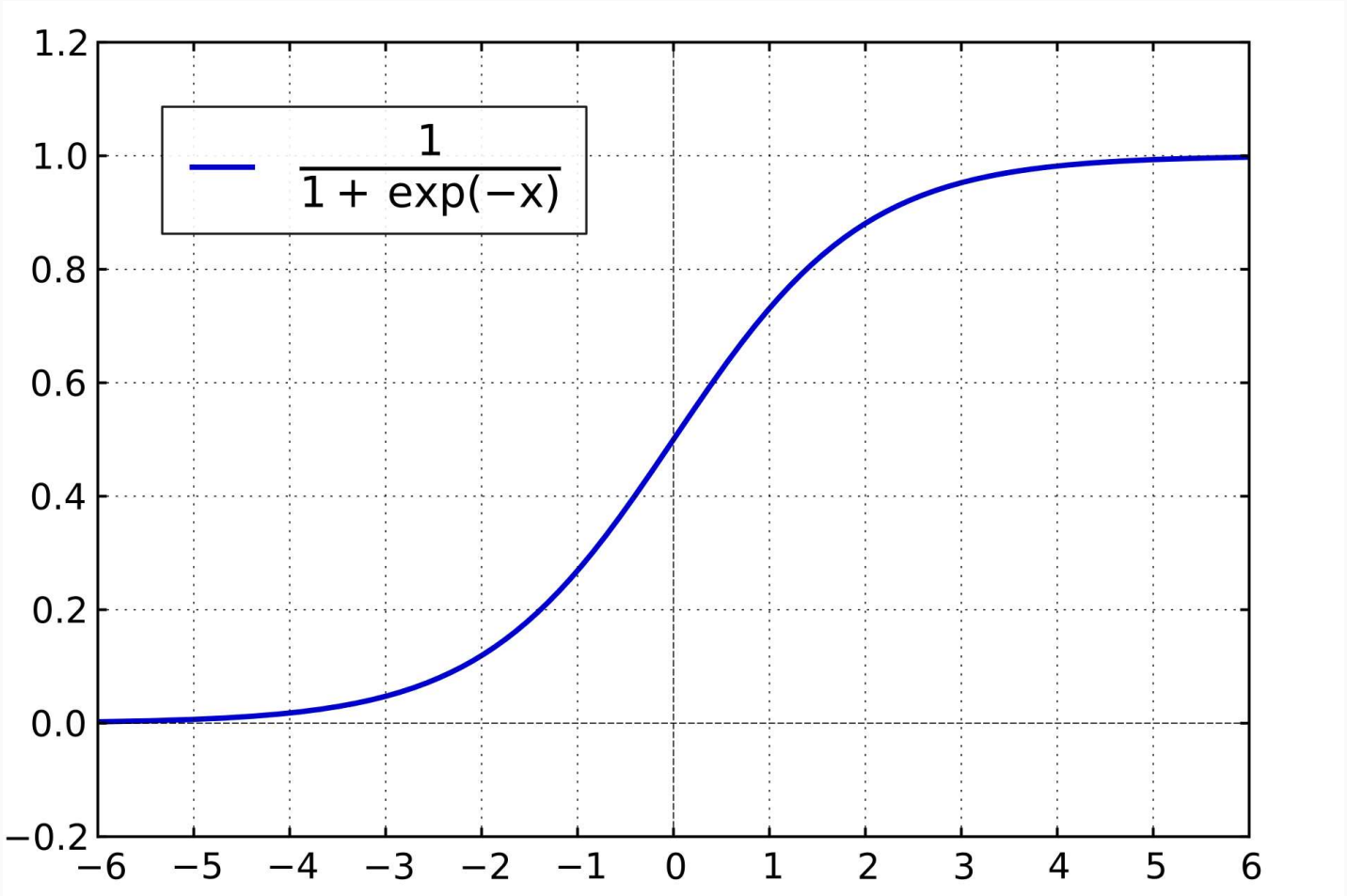
Sinnvoll nutzbar bei Funktionen die als Ergebnis nur ein klares Wahr/Falsch kennen

Nachteile:

- Schwaches aber korrektes Input-Signal wird u. U. nicht beachtet, wenn der Output z. B. 0 bleibt, anstatt wenigstens 0,3 anzuzeigen.
- Es gibt keinen Gradienten (keine Steigung), welcher für die Backpropagation nötig ist ($\rightarrow$ man weiß nicht, wie stark das Gewicht einer Verbindung angepasst werden muss)

Aktivierungsfunktion - Logistische (sigmoide) Funktion

Meist genutzte Alternative: **Logistische (sigmoide) Funktion**



Quelle: Geek3, CC BY 3.0, via Wikimedia Commons

Werte zwischen 0 und 1 möglich $\Rightarrow$ **Wahrscheinlichkeiten**

Aktivierungsfunktion

Aufgaben 2b

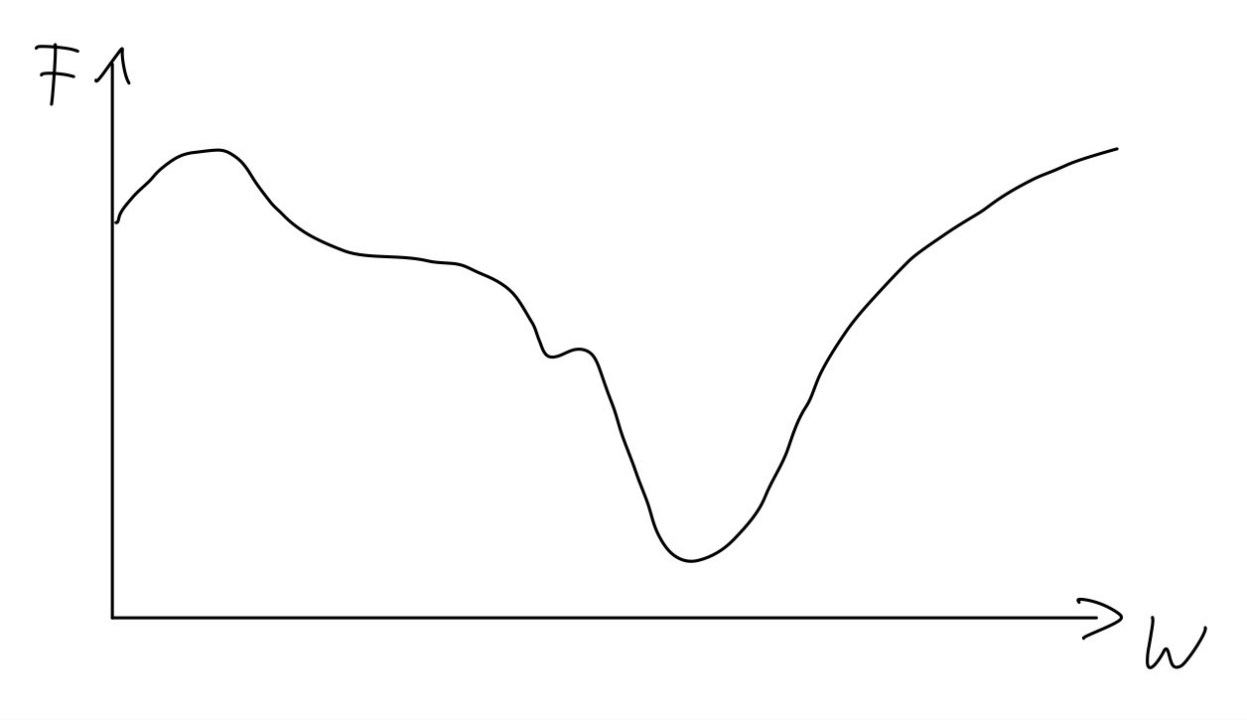
Gradientenabstiegsverfahren und Lernrate

Was für einen Einfluss hat die Lernrate?

Gradientenabstiegsverfahren und Lernrate

Was für einen Einfluss hat die Lernrate?

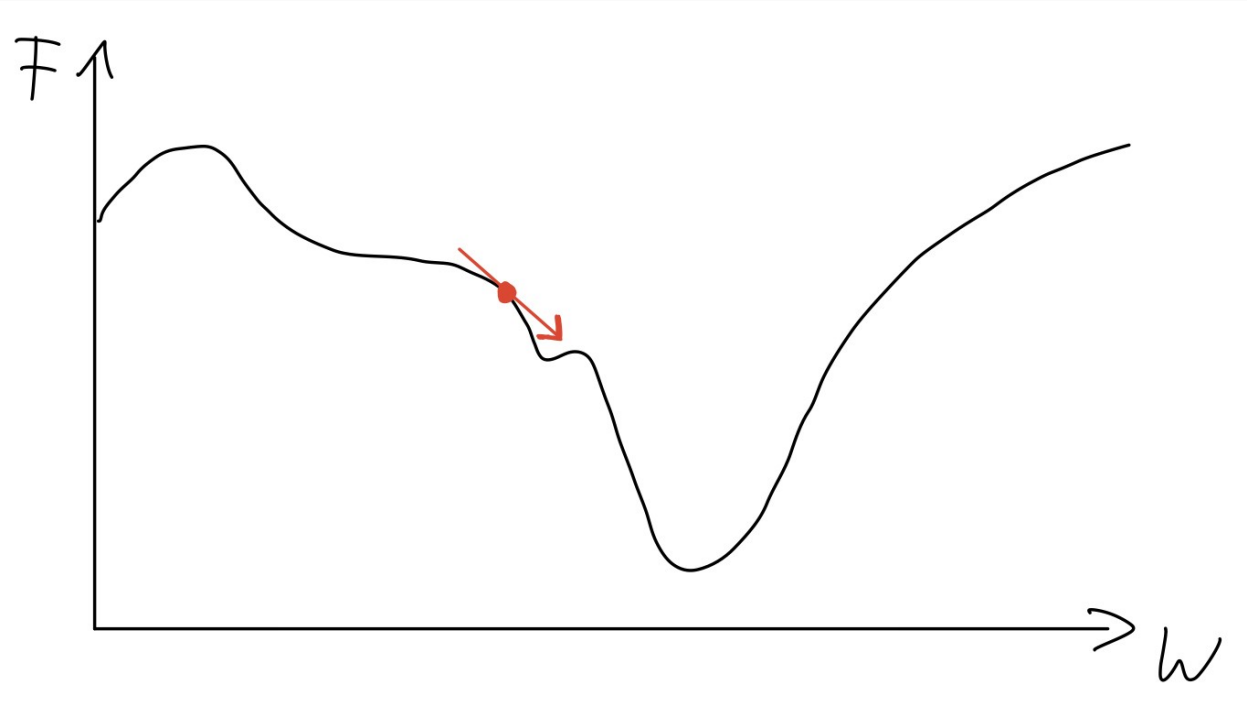
Für jede einzelne Verbindung kann man sich den Fehler (bzw. die Abweichung vom gewünschten Ergebnis) in Abhängigkeit des Gewichts anzeigen lassen:



Gesucht ist natürlich das Minimum (lokal oder besser global).

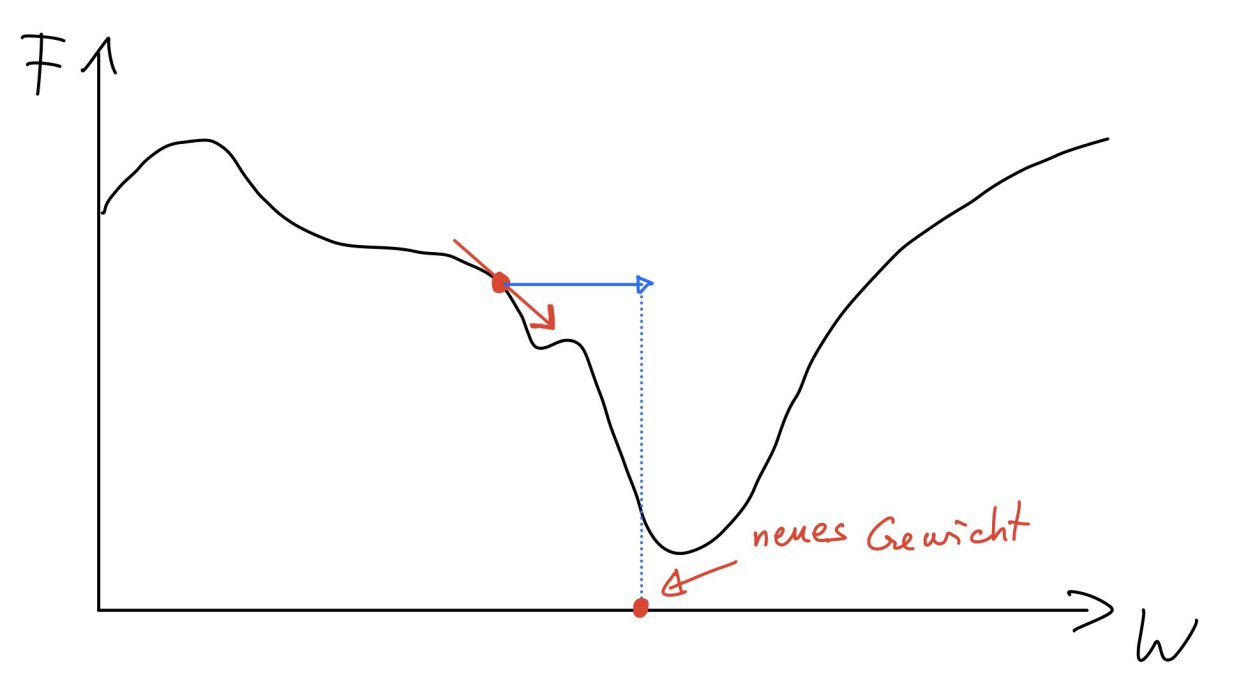
Gradientenabstiegsverfahren und Lernrate

Mit dem ersten Trainingsdatensatz und zufälligen Initialgewichten erfährt man den Fehler und außerdem die Richtung der Abweichung (= *muss das Gewicht vergrößert oder verkleinert werden?*). Die Richtung erfährt man hier grafisch über die Steigung (Gradient).



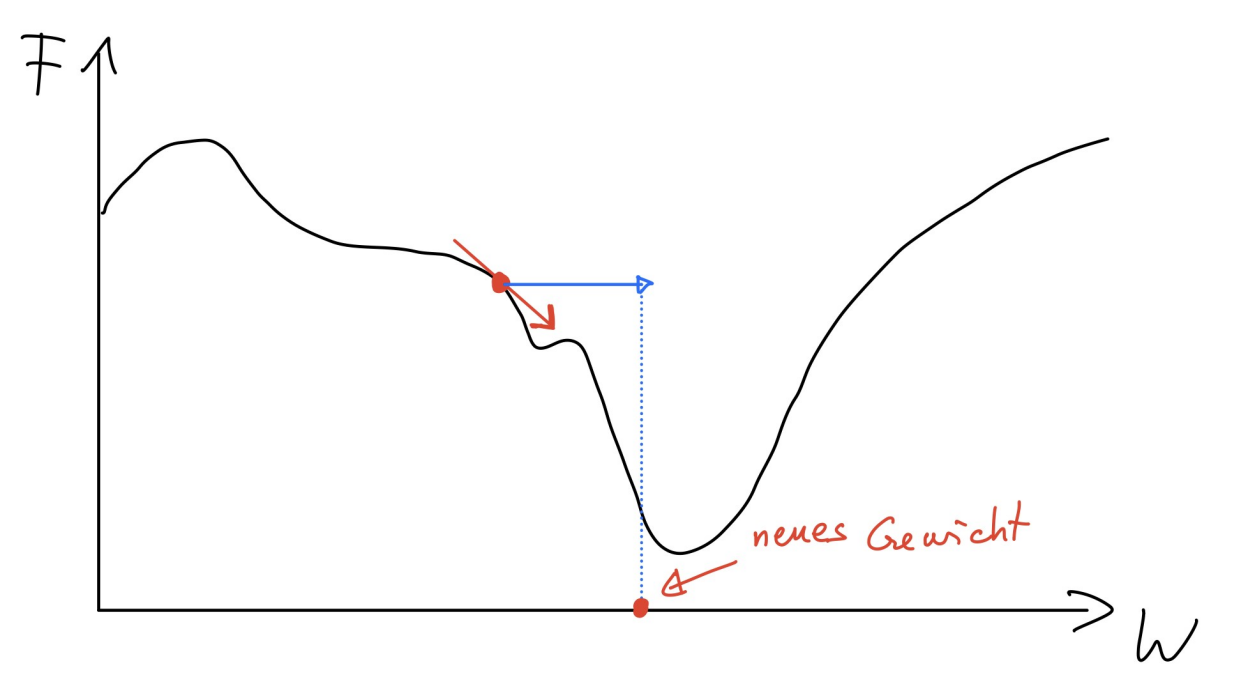
Gradientenabstiegsverfahren und Lernrate

Die Stärke der Korrektur ist abhängig von der Steigung des Gradienten und der Größe der Lernrate.



Gradientenabstiegsverfahren und Lernrate

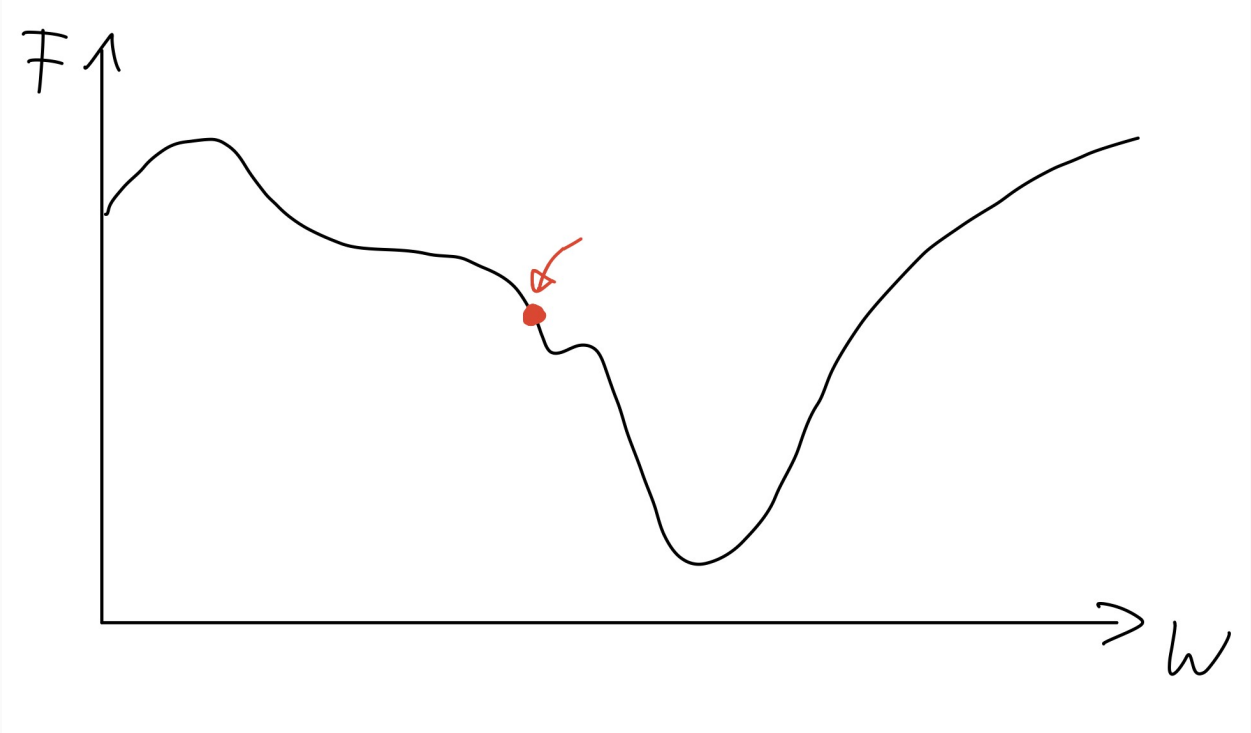
Die Stärke der Korrektur ist abhängig von der Steigung des Gradienten und der Größe der Lernrate.



Je näher man dem flachen Minimum kommt, desto kleiner wird die Korrektur.

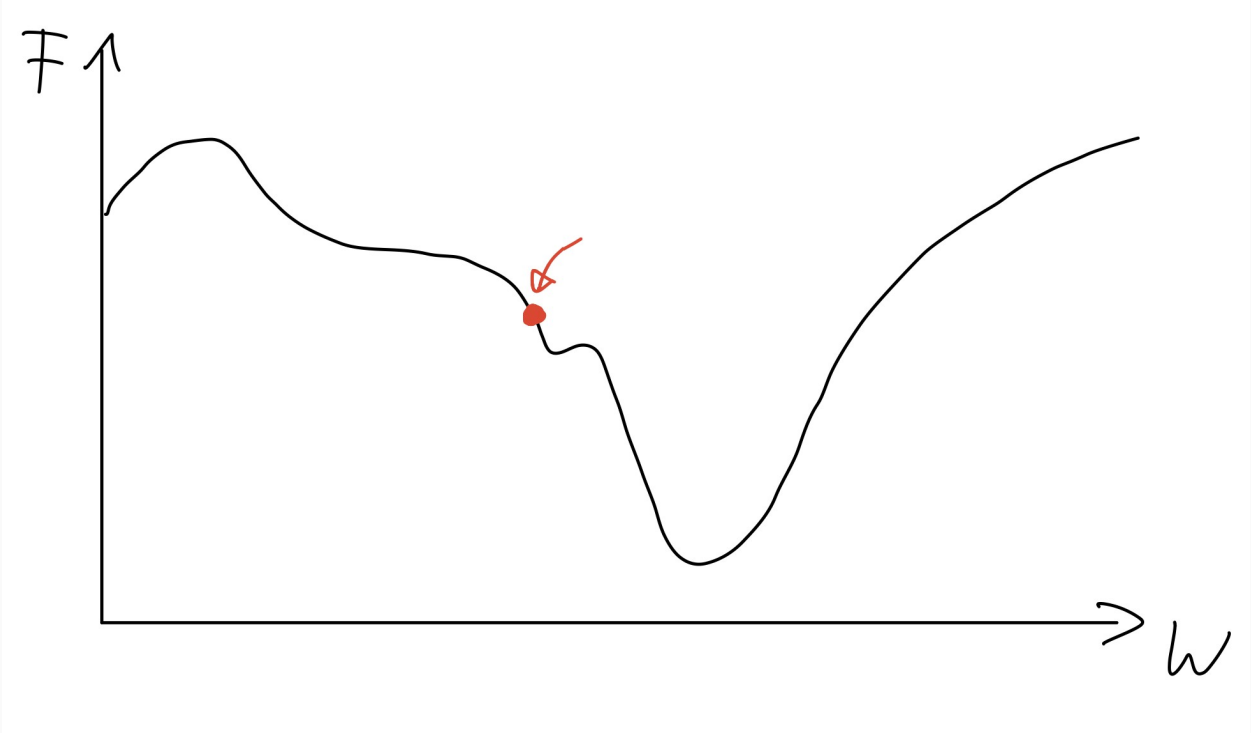
Gradientenabstiegsverfahren und Lernrate

Was geschieht hier, wenn die Lernrate zu klein ist?



Gradientenabstiegsverfahren und Lernrate

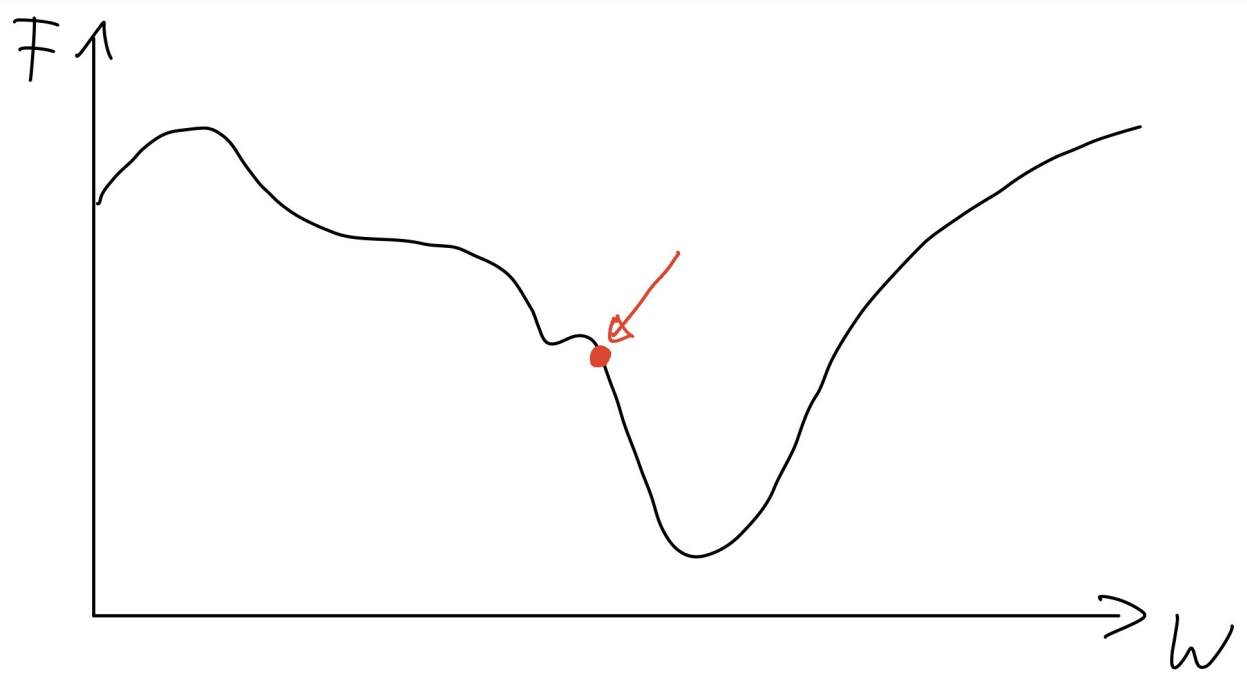
Was geschieht hier, wenn die Lernrate zu klein ist?



$\Rightarrow$ Es wird nur das lokale Minimum gefunden

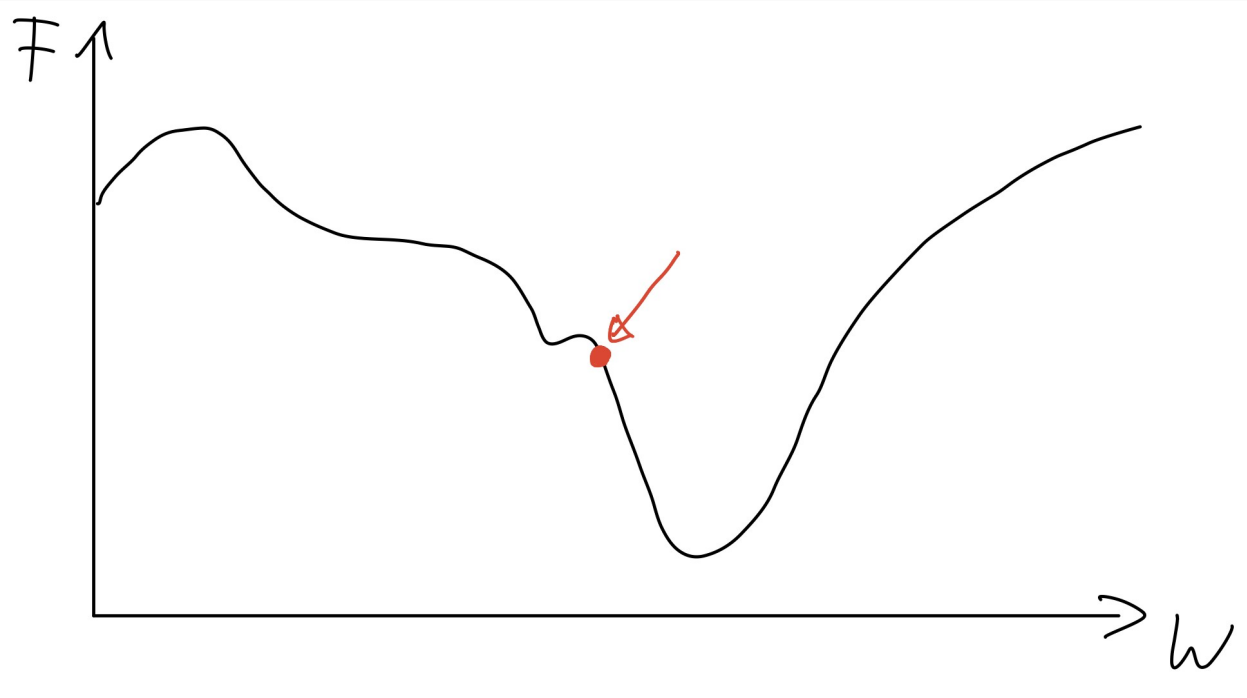
Gradientenabstiegsverfahren und Lernrate

Was geschieht hier, wenn die Lernrate zu groß ist?



Gradientenabstiegsverfahren und Lernrate

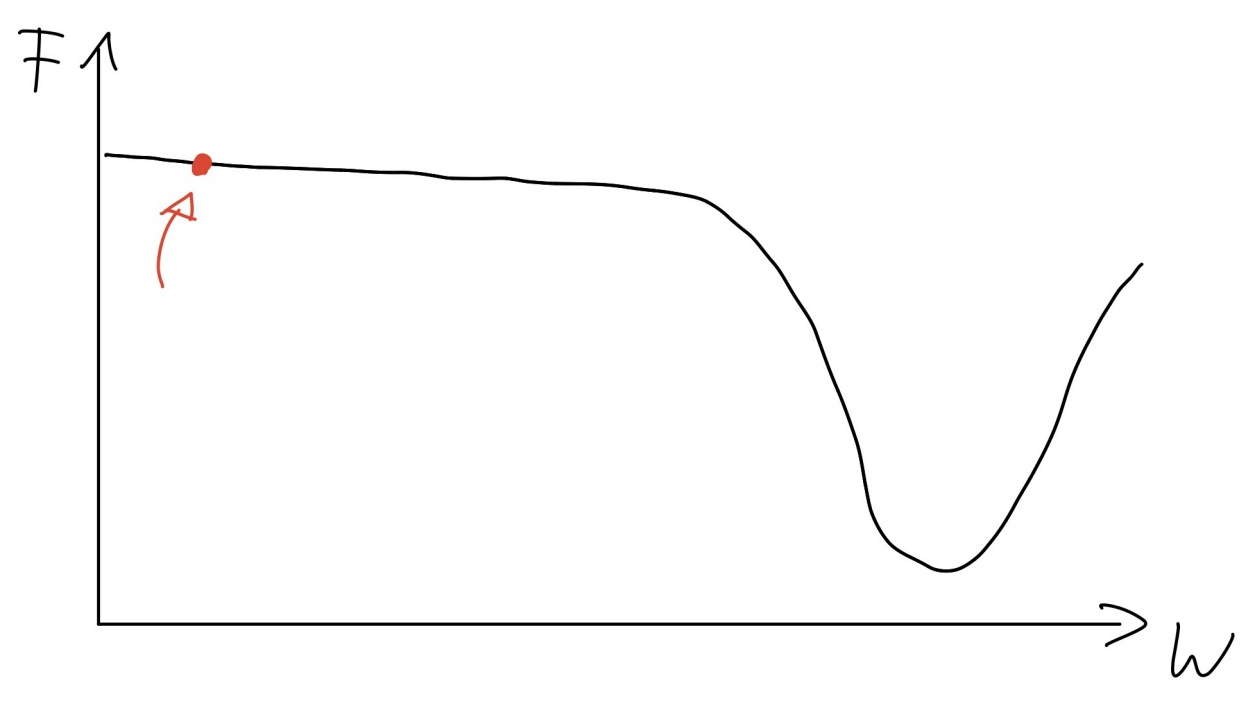
Was geschieht hier, wenn die Lernrate zu groß ist?



⇒ Das globale Minimum wird immer wieder von rechts und links übersprungen (Oszillation / Schwingung)

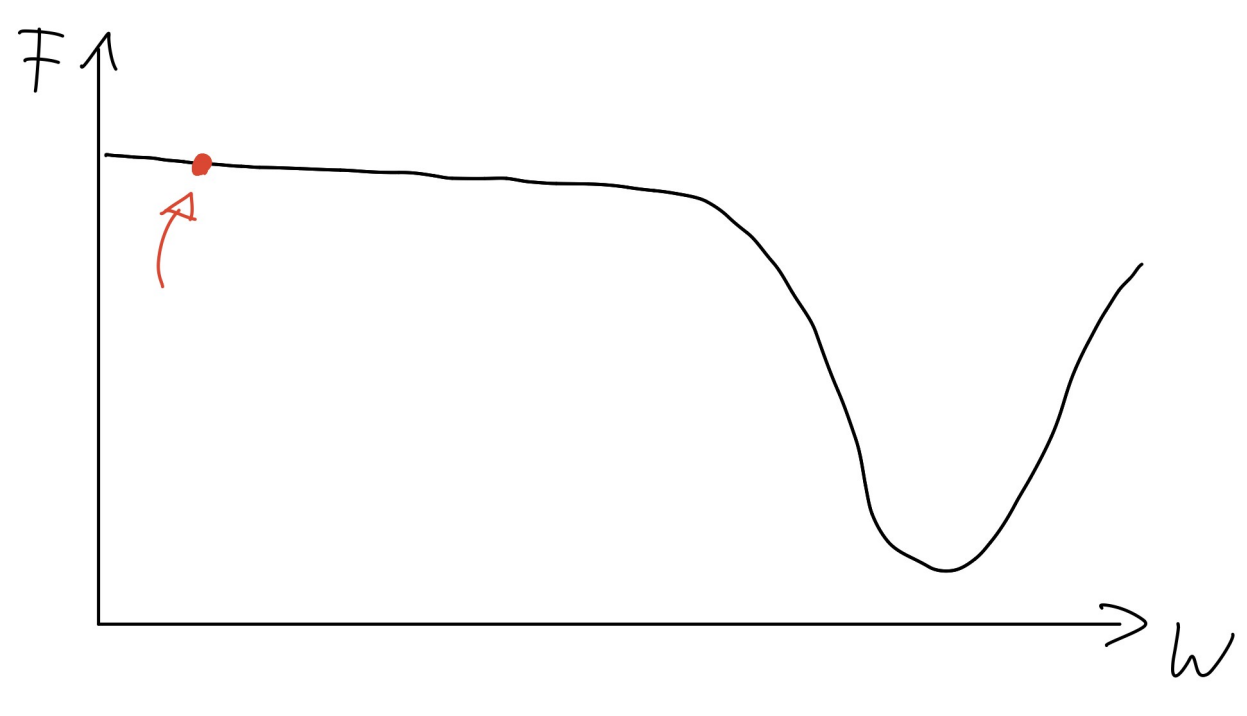
Gradientenabstiegsverfahren und Lernrate

Was geschieht hier, wenn die Lernrate zu klein ist?



Gradientenabstiegsverfahren und Lernrate

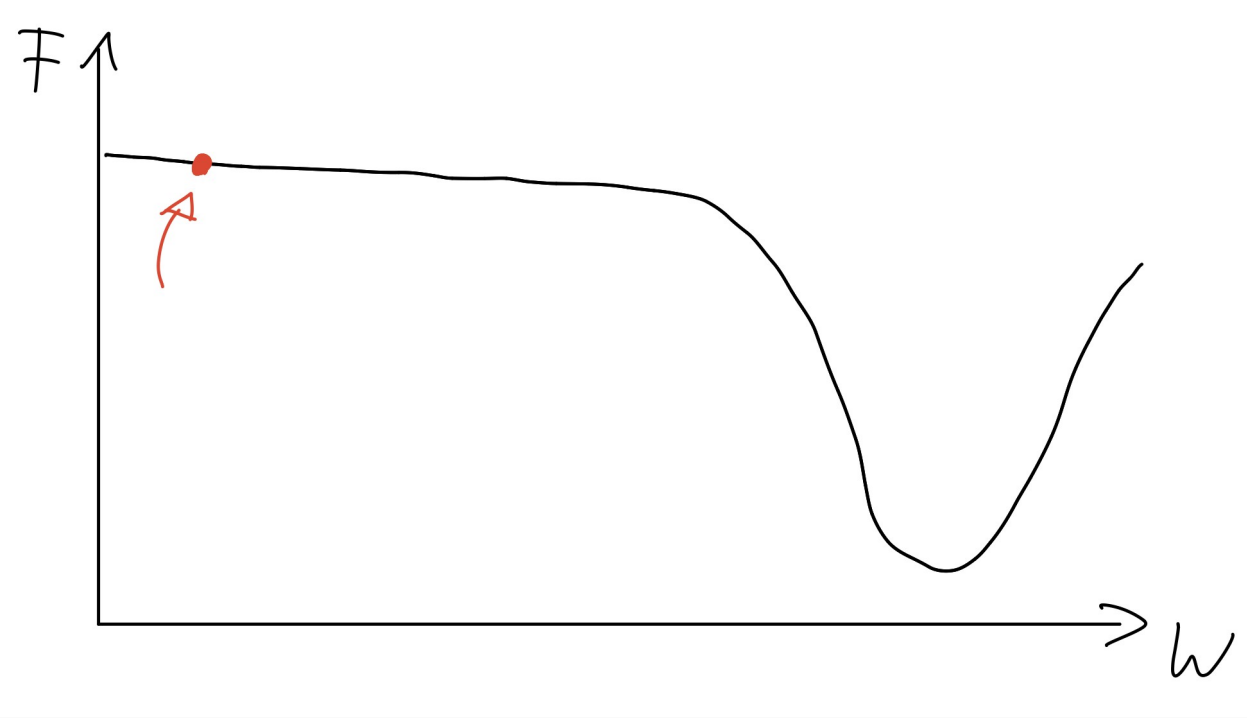
Was geschieht hier, wenn die Lernrate zu klein ist?



$\Rightarrow$ Es dauert Ewigkeiten bis das flache Plateau verlassen wird

Gradientenabstiegsverfahren und Lernrate

Was geschieht hier, wenn die Lernrate zu klein ist?



$\Rightarrow$ Es dauert Ewigkeiten bis das flache Plateau verlassen wird

$\rightarrow$ **Restliche Aufgaben zu MemBrain**

Fachbegriffe

Overfitting:

Frage: Was kann passieren, wenn ein neuronales Netzwerk zu lange bzw. zu perfekt auf einen Trainingsdatensatz trainiert wird?

Fachbegriffe

Overfitting:

Frage: Was kann passieren, wenn ein neuronales Netzwerk zu lange bzw. zu perfekt auf einen Trainingsdatensatz trainiert wird?

→ Es wird weniger flexibel gegenüber neuen Eigenschaften in den Testdaten...

→ ... und liefert dadurch wahrscheinlich nur schlechte Ergebnisse

Fachbegriffe

Bias-Unit:

Bisherige Netzwerke haben verschiedene Einschränkungen, z. B.:

Bei bekannter Aktivierungsfunktion

$$\sum_n x_n \cdot w_n \geq a$$

ist es nicht möglich zu feuern, wenn die Inputs 0 sind und die Aktivierungsschwelle über 0 liegt.

Fachbegriffe

Bias-Unit:

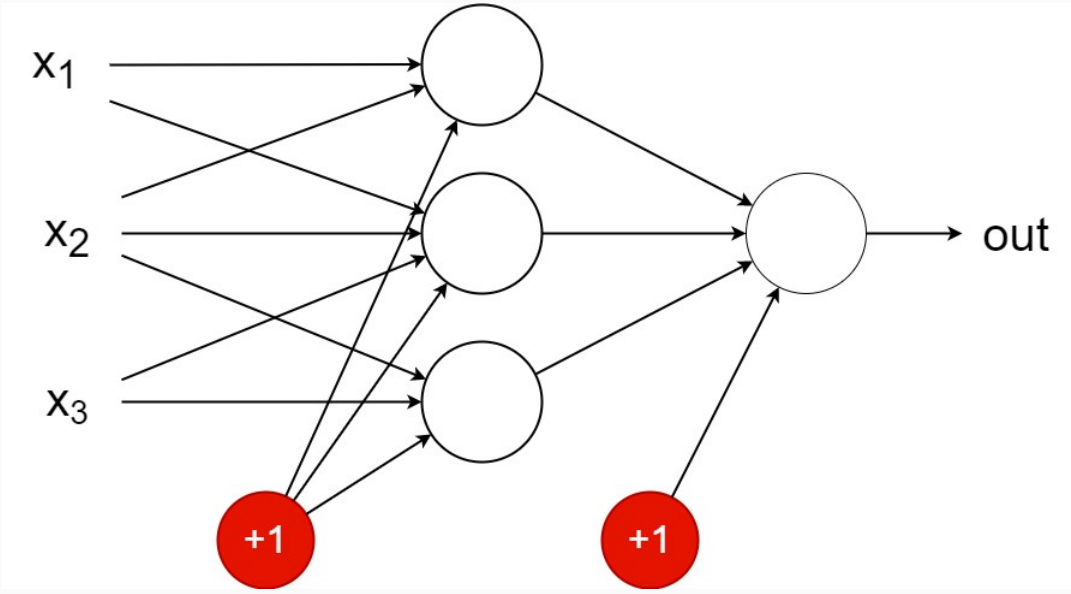
Bisherige Netzwerke haben verschiedene Einschränkungen, z. B.:

Bei bekannter Aktivierungsfunktion

$$\sum_n x_n \cdot w_n \geq a$$

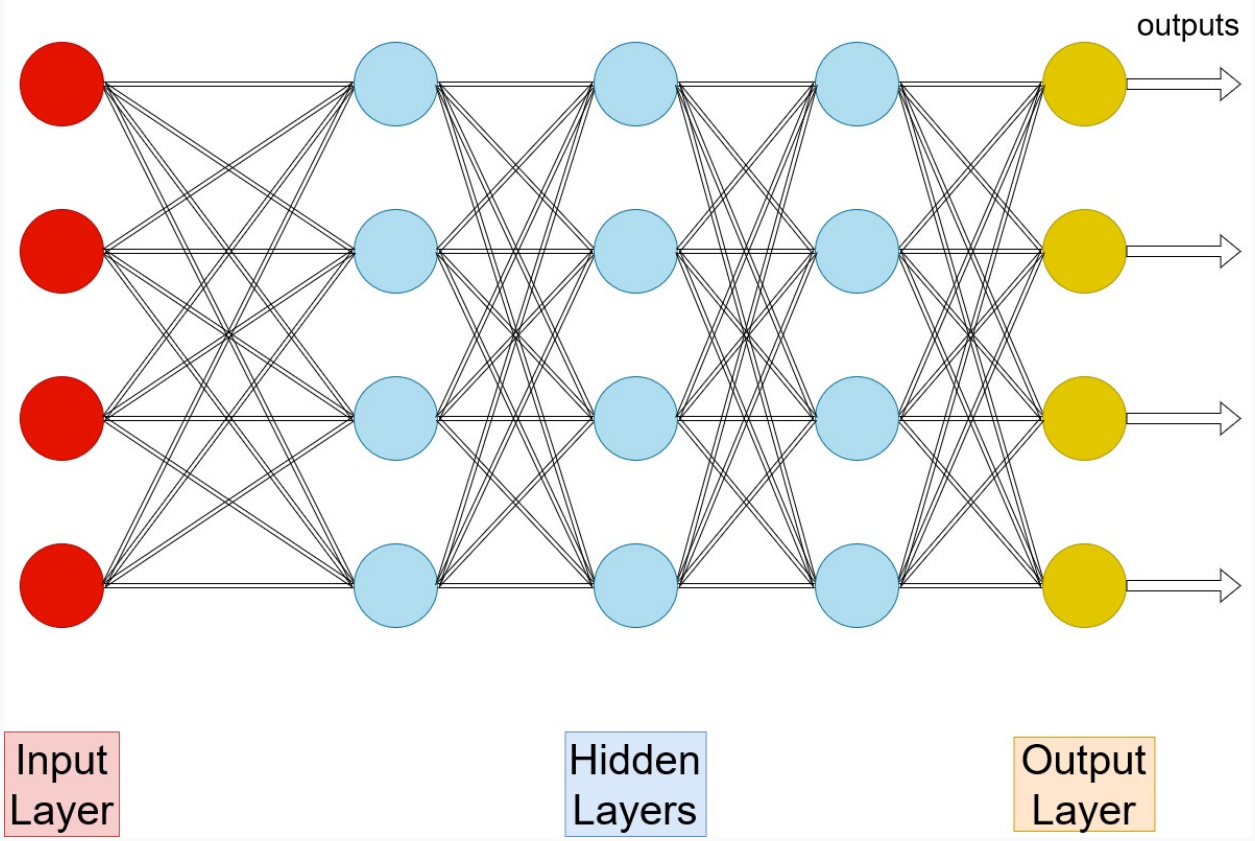
ist es nicht möglich zu feuern, wenn die Inputs 0 sind und die Aktivierungsschwelle über 0 liegt.

Lösung: Konstanter Input „1“ mit eigenem Gewicht für jedes Neuron.



Autoencoder

In diesem Neuronalen Netz sind die Trainingsdaten beim Input und Output komplett identisch

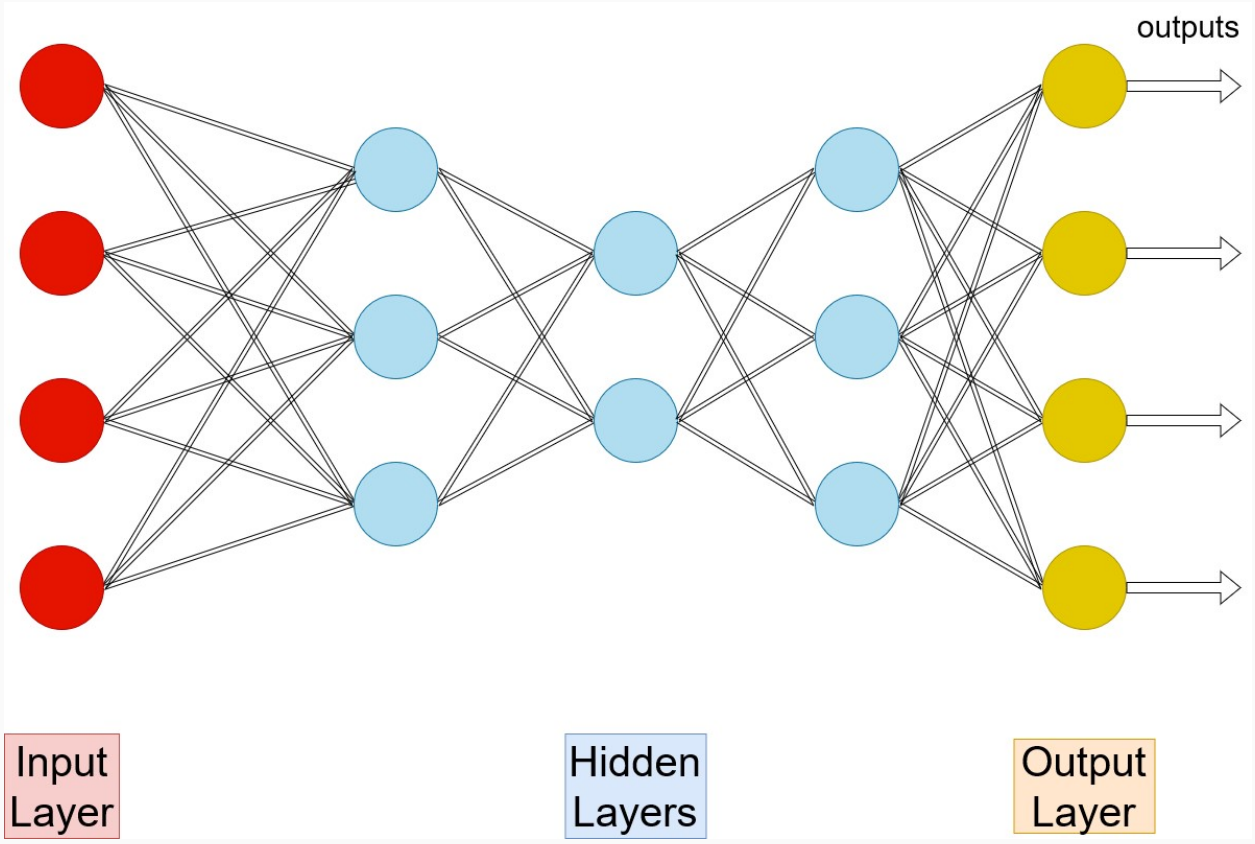


$\Rightarrow$ Dieses Netz ist komplett sinnfrei, da es keinen Informationsgewinn gibt

Autoencoder

Auch hier sind die Trainingsdaten am Input und Output identisch.

Was ist der Sinn des Ganzen?

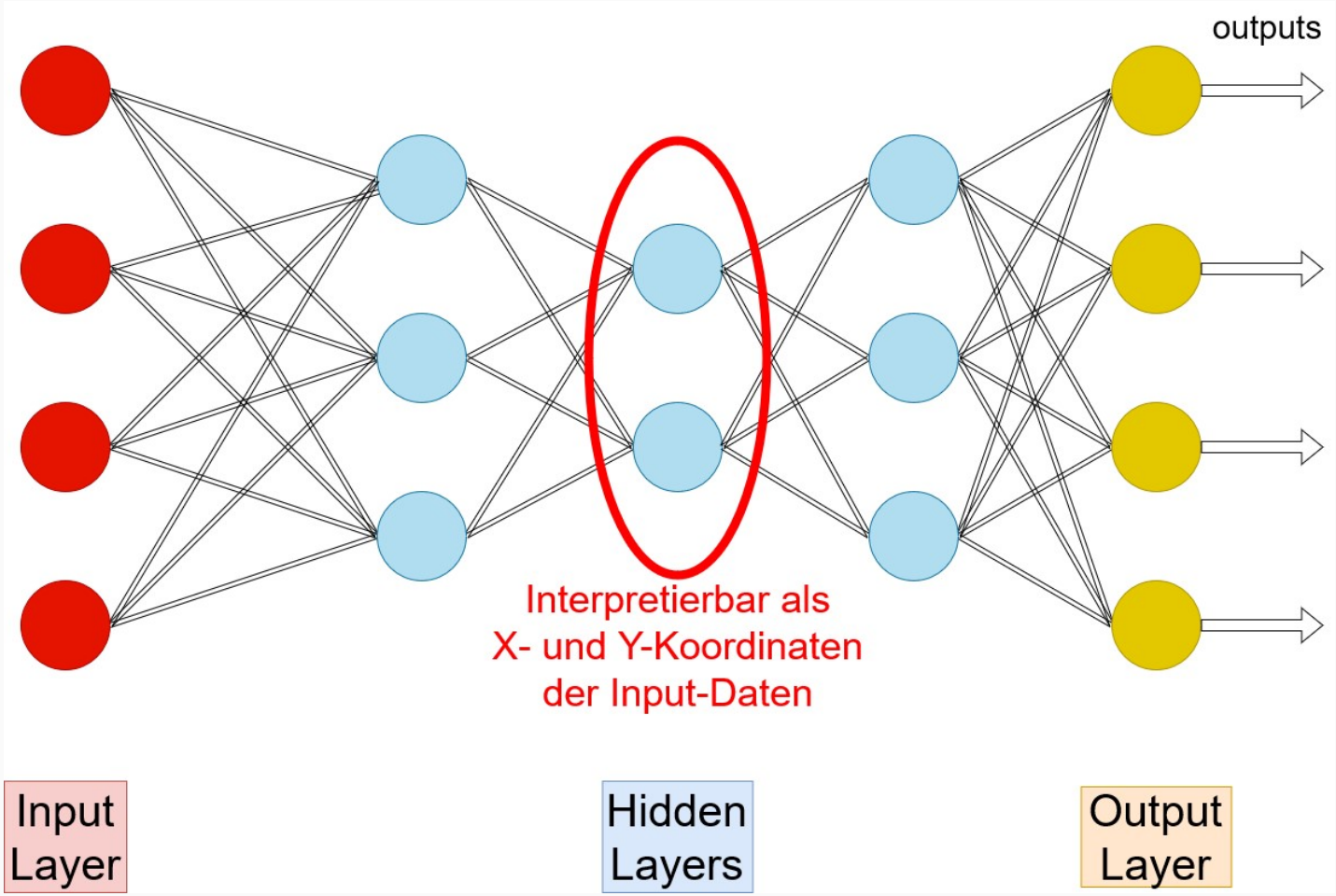


Autoencoder

Im Optimalfall gehen keine Informationen verloren (Output = Input) - man schafft es aber, die Daten zu komprimieren auf nur zwei Neuronen.

Autoencoder

Im Optimalfall gehen keine Informationen verloren (Output = Input) - man schafft es aber, die Daten zu komprimieren auf nur zwei Neuronen.



Autoencoder

Wenn die Aktivierungswerte beider Neuronen als Koordinaten interpretiert werden, dann kann man diese grafisch auswerten und man kann möglicherweise für alle Trainingsdaten unterschiedliche Gruppierungen entdecken.

Übersicht über maschinelles Lernen

Man kann maschinelles Lernen grob in folgende 3 Kategorien unterteilen:

- **Supervised Learning** (überwacht)

  Unsere Vorgehensweise: man hat Trainingsdaten, bei denen man den Input und den erwarteten Output kennt. Der Algorithmus passt die Parameter anhand des erwarteten Outputs an.

Übersicht über maschinelles Lernen

Man kann maschinelles Lernen grob in folgende 3 Kategorien unterteilen:

- **Supervised Learning** (überwacht)

  Unsere Vorgehensweise: man hat Trainingsdaten, bei denen man den Input und den erwarteten Output kennt. Der Algorithmus passt die Parameter anhand des erwarteten Outputs an.

- **Unsupervised Learning** (unüberwacht)

  Es gibt nur Input-Daten ohne erwarteten Output. Der Algorithmus muss zunächst selbstständig mithilfe statistischer Modelle Kategorien oder Zusammenhänge erkennen.

Übersicht über maschinelles Lernen

Man kann maschinelles Lernen grob in folgende 3 Kategorien unterteilen:

- **Supervised Learning** (überwacht)

  Unsere Vorgehensweise: man hat Trainingsdaten, bei denen man den Input und den erwarteten Output kennt. Der Algorithmus passt die Parameter anhand des erwarteten Outputs an.

- **Unsupervised Learning** (unüberwacht)

  Es gibt nur Input-Daten ohne erwarteten Output. Der Algorithmus muss zunächst selbstständig mithilfe statistischer Modelle Kategorien oder Zusammenhänge erkennen.

- **Reinforcement Learning** (bestärkend)

  Weder gekennzeichnete Input- noch Output-Daten. Es gibt nur Regeln, nach denen der Algorithmus eine positive oder negative "Belohnung" bekommt $\rightarrow$ das Netz muss selbstständig Strategien entwickeln (*vgl. Mutation/Evolution*) und diese verwerfen/bestärken.

Übersicht über maschinelles Lernen

Beispiel für Reinforcement Learning:

`https://www.youtube.com/watch?v=gn4nRCC9TwQ`

Zusammenfassung aktueller Probleme mit ML

„Wie (und warum) Künstliche Intelligenz diskriminiert... und wie wir es verhindern können“

`https://www.youtube.com/watch?v=9WPyD2BLooo`