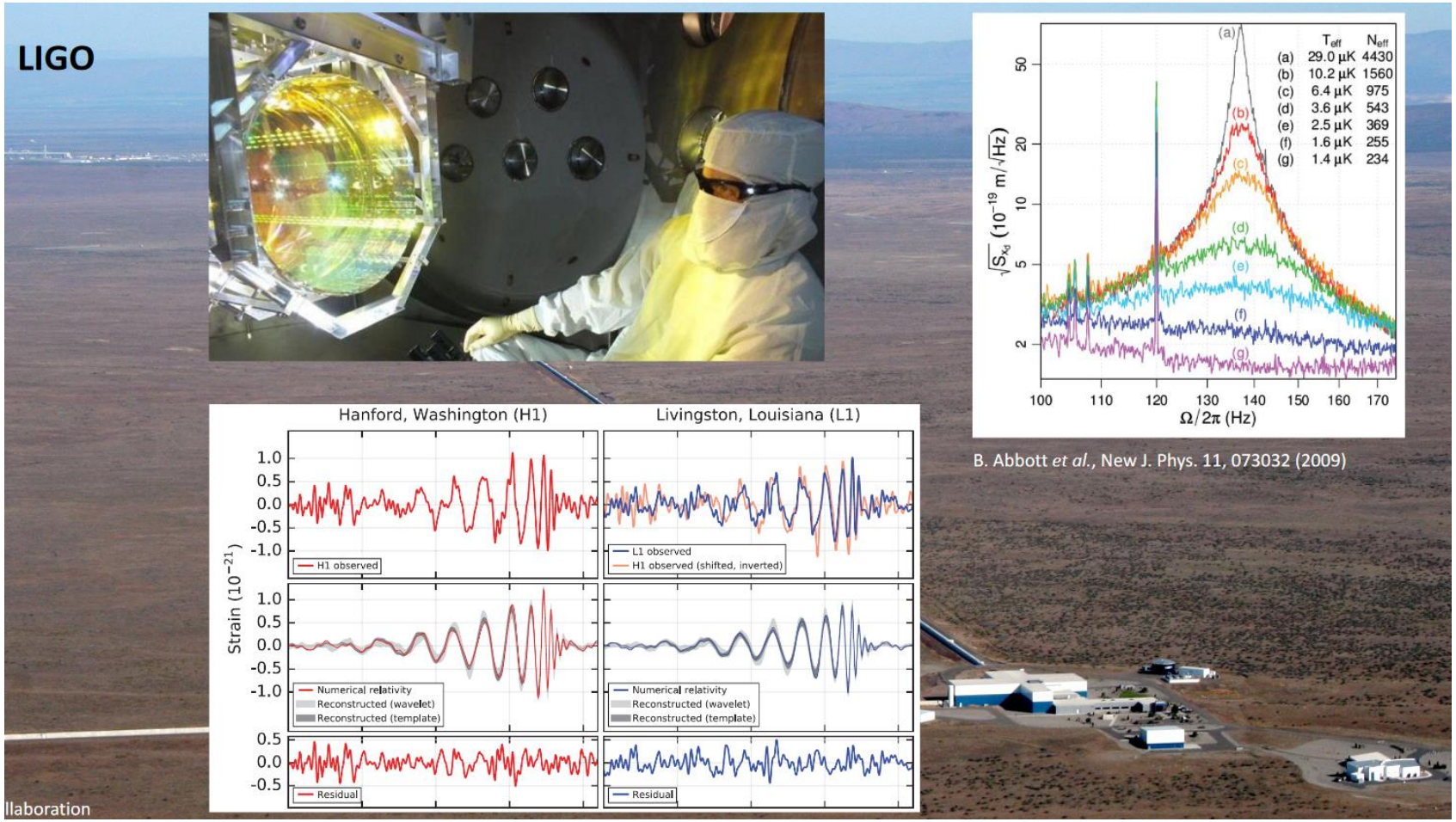
ETH zürich

Datenanalyse in der Physik

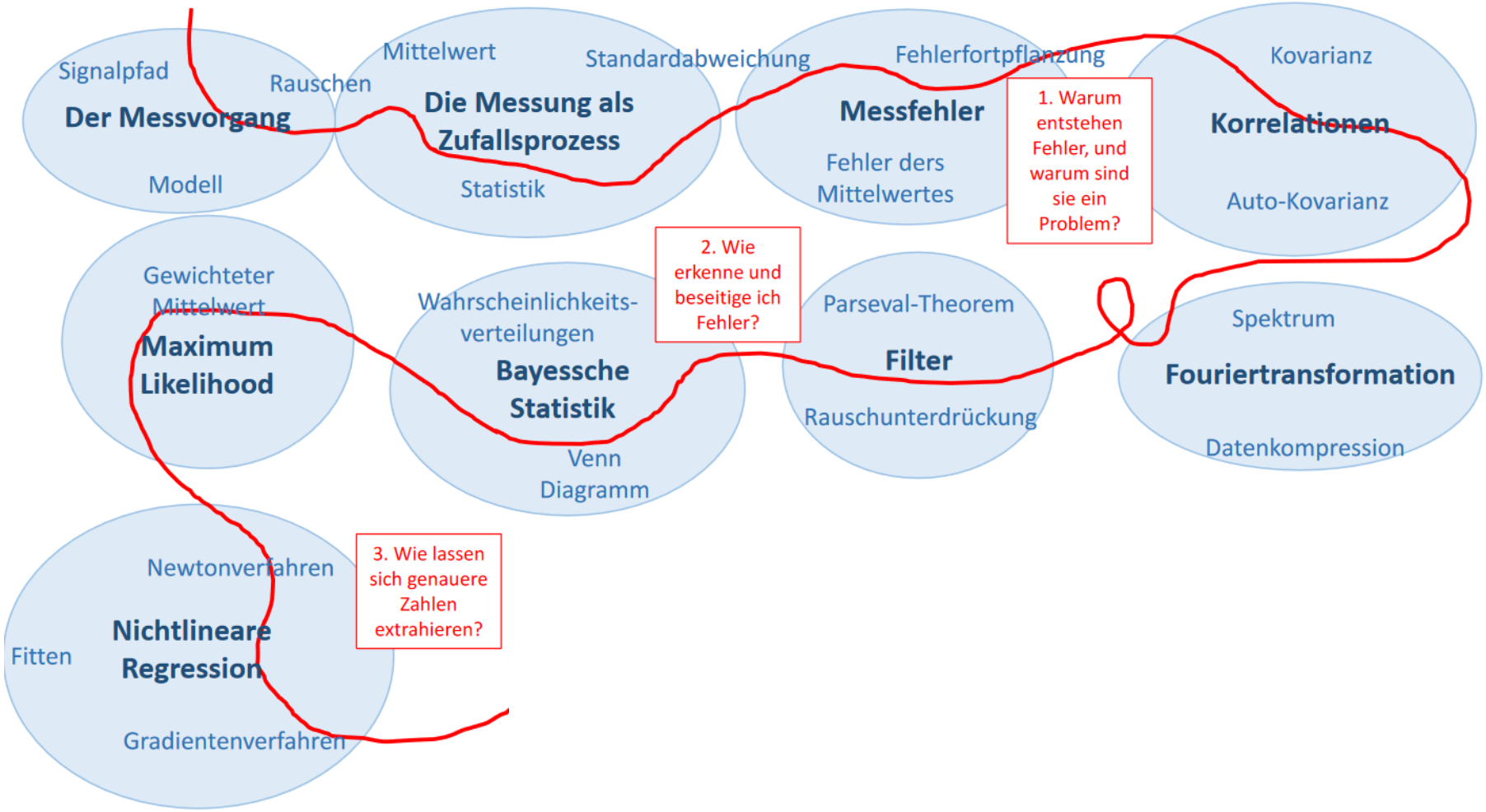
Carlos Cotrini, Marcel Lüthi (D-INFK)
Vorlesung am D-Phys der ETH Zürich

Einführung erste Woche



B. Abbott *et al.*, New J. Phys. 11, 073032 (2009)

Der rote Faden



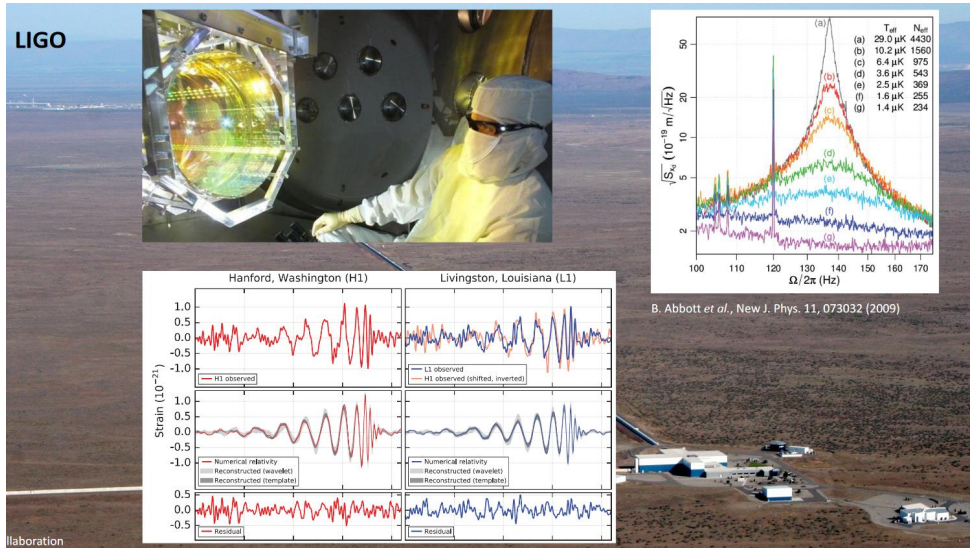
Diskussion

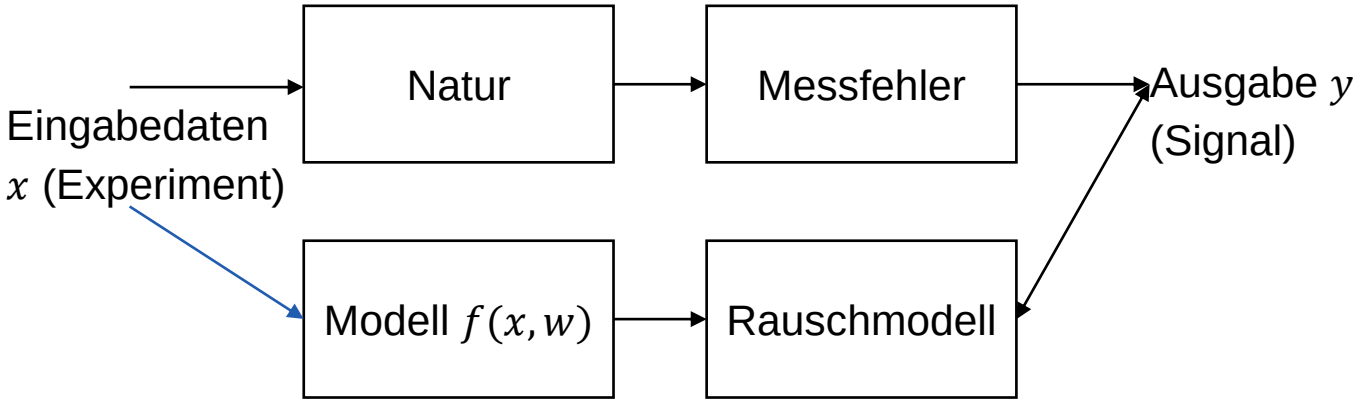
- Können Sie in einem Satz erklären, wozu Sie all diese Methoden/Techniken brauchen? Was wollen Sie als Physiker:in mit diesen Methoden wirklich erreichen?

- Unser nächstes Thema in der Vorlesung ist Maschinelles Lernen, also Lernen aus Daten. Welche Methoden die Sie bisher gelernt haben, könnten als *Maschinelles Lernen* bezeichnet werden?

Maschinelles Lernen (T. Mitchell)
*A computer program is said to **learn** from experience with respect to some class of tasks, if its performance at these tasks improves with experience.*

Datenanalyse in der Physik





Gegeben

- Mathematisches Modell (Hypothese / Theorie)
- Daten eines Experiments

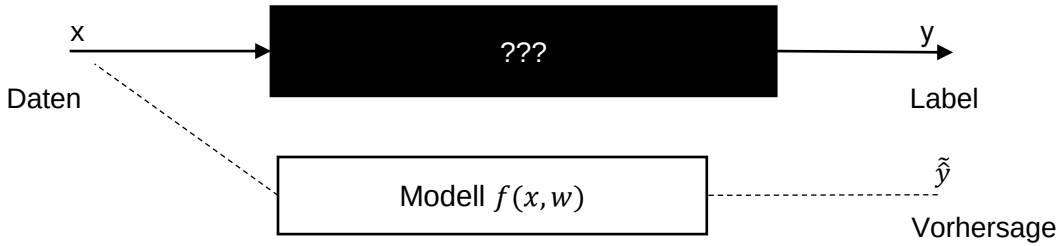
Aufgabe

- Verifizieren des Modells mit den Daten aus den Experimenten

Maschinelles Lernen – Gleich und doch anders

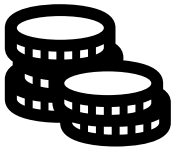
- Regisseur: Tom Fox
- Länge: 93 Minuten
- Darsteller: Eichhörnchen, Hasen, Füchse
- FSK: 14

…



?

IMDb



- **Gegeben** Daten (Messwerte / Beobachtungen) – Oft nicht systematisch gesammelt
- **Aufgabe** Modell finden, welches neue Vorhersagen machen kann
- Physikalischer Zusammenhang oft nicht gegeben oder nicht von primären Interesse

Agenda der letzten 3 Vorlesungen

- **Heute**: Linearer Regression zu Neuronalen Netzen
  - Anknüpfung an Bestehendes.
  - Aufbauen der Intuition für neuronale Netze.

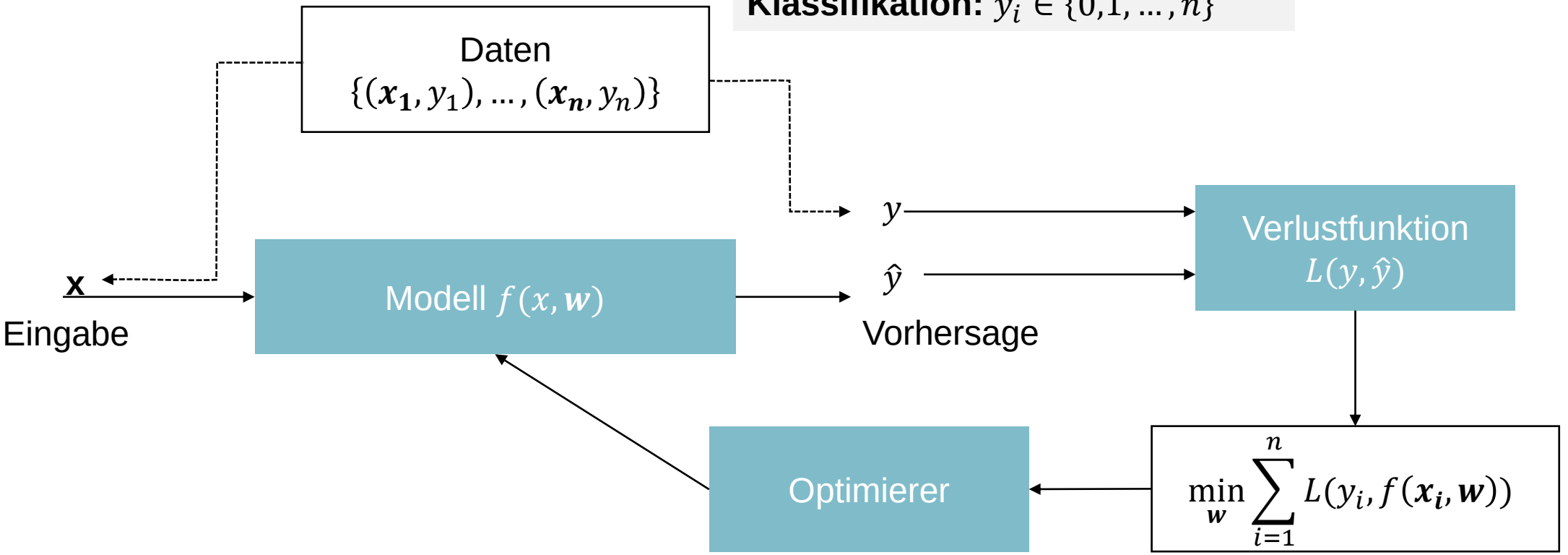
- **Nächste Woche**: Trainieren und Validieren von Modellen – Deep Learning
  - Wie trainieren und validieren wir richtig? Wie bestimmen wir Parameter?
  - Deep Learning Anwendungen und Architekturen

- **Übernächste Woche**: Weitere Themen im Maschinellen Lernen
  - Wie arbeite ich mit nicht numerischen Daten
  - Was gibt es noch für Arten von Maschinellem Lernen ausser Überwachtes Lernen

Überwachtes Lernen (supervised learning)

Eingaben $\boldsymbol{x_i} \in \mathbb{R}^d$

Regression: Ausgabe $y_i \in \mathbb{R}$
Klassifikation: $y_i \in \{0,1,\ldots,n\}$



Einfachstes Beispiel: Lineare Regression

Daten

$$D = \{(x_1, y_1), \ldots, (x_n, y_n)\}, x \in \mathbb{R}, y \in \mathbb{R}$$

Modell

$$f(x, \boldsymbol{w}) = f(x, a, b) = ax + b$$

Parameter

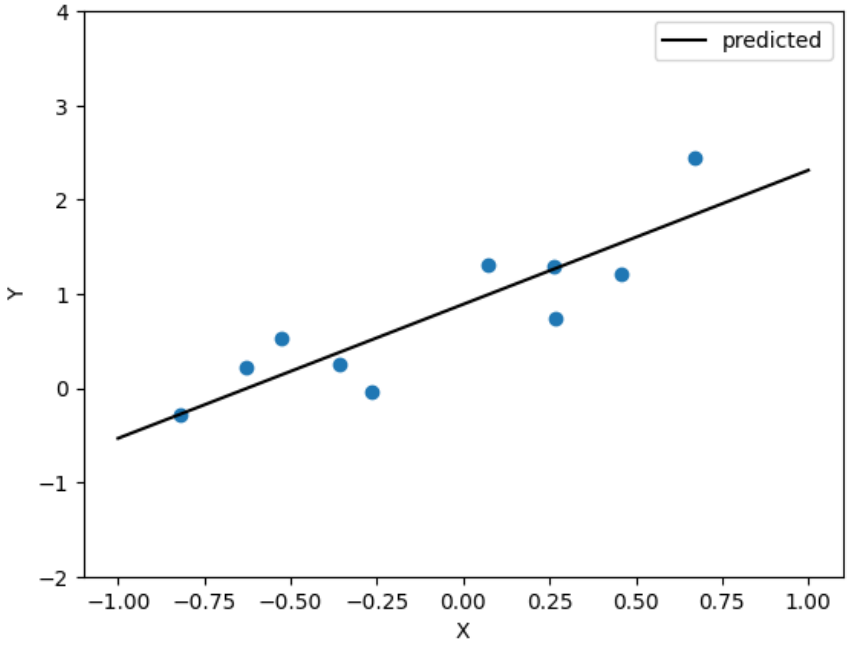
- Steigung a, Achsenabschnitt b
  Parameter vector $\boldsymbol{w} = \begin{pmatrix} a \\ b \end{pmatrix}$

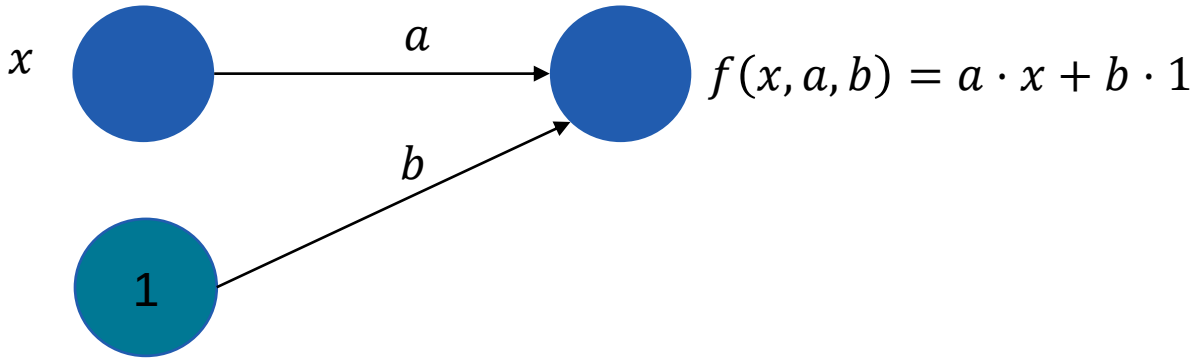
Verlustfunktion

$$L(y, \hat{y}) = (y - f(x, a, b))^2$$

Optimieren

- Optimale Lösung für $\min_{a,b} \sum_{i=1}^{n} (y_i - f(x_i, a, b))^2$
  durch Normalgleichungen.





Lineare Regression in höheren Dimensionen

Daten

$$D = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_n, y_n)\}, \boldsymbol{x} \in \mathbb{R}^d, y \in \mathbb{R}$$

Parameter:

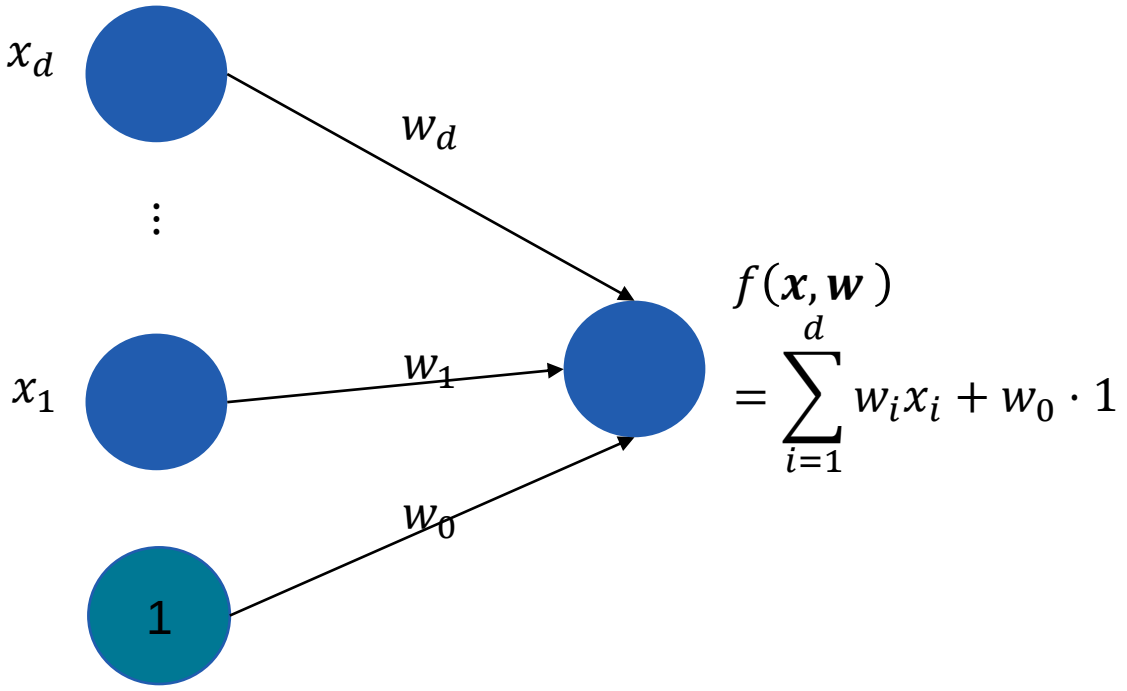
- $\mathbf{w} = (\mathrm{w}_0, w_1, w_2, \dots, w_d)^T$

Modell

$$f(\boldsymbol{x}, \boldsymbol{w}) = w_1 x_1 + \cdots + w_d x_d + w_0 \cdot 1$$

Verlustfunktion

$$L(y, \hat{y}) = \big(y - f(\boldsymbol{x}, \boldsymbol{w})\big)^2$$



Optimierung

- Optimale Lösung für $\min_{\boldsymbol{w}} \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i, \boldsymbol{w}))^2$ durch Normalgleichungen.

Lineare Basisfunktionsmodelle

Daten

$$D = \{(x_1, y_1), \dots, (x_n, y_n)\}, x \in \mathbb{R}, y \in \mathbb{R}$$

Basisfunktionen

$$\phi_1, \dots, \phi_m, \phi_i : \mathbb{R} \to \mathbb{R}$$

Merkmalstransformation

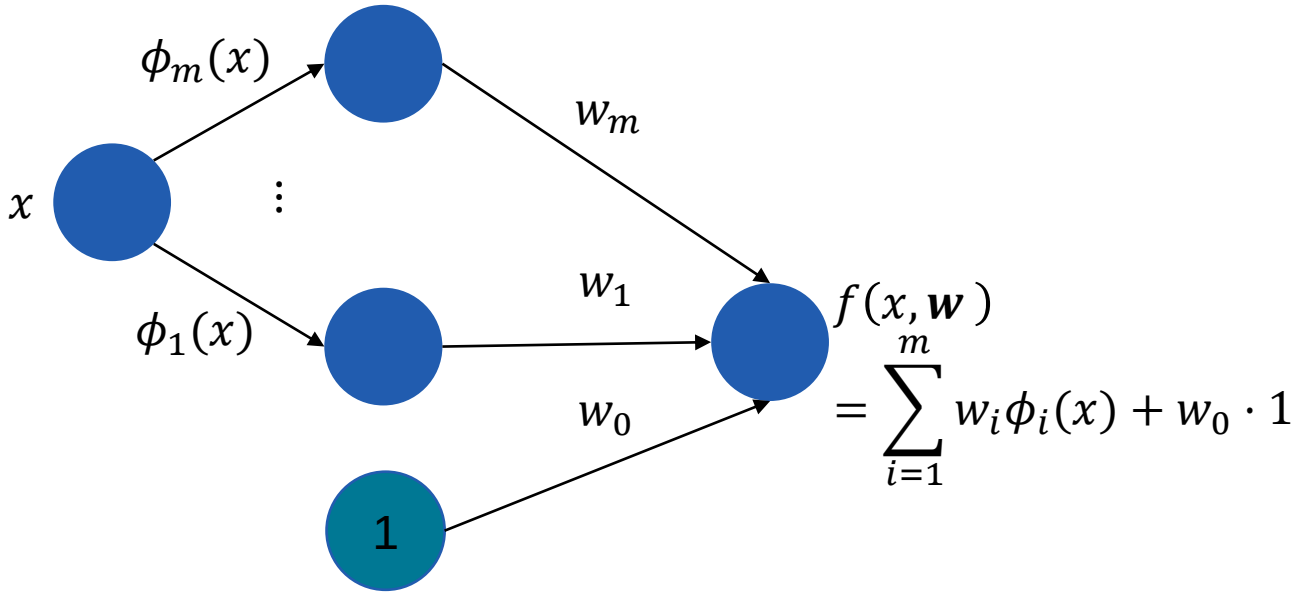
$$\boldsymbol{\phi}(x) = (\phi_1(x), \cdots, \phi_m(x))^T : \mathbb{R} \to \mathbb{R}^m$$

Model

$$f(x, \boldsymbol{w}) = w_1\phi_1(x) + \cdots + w_m\phi_m(x) + w_0 1$$

Optimierung

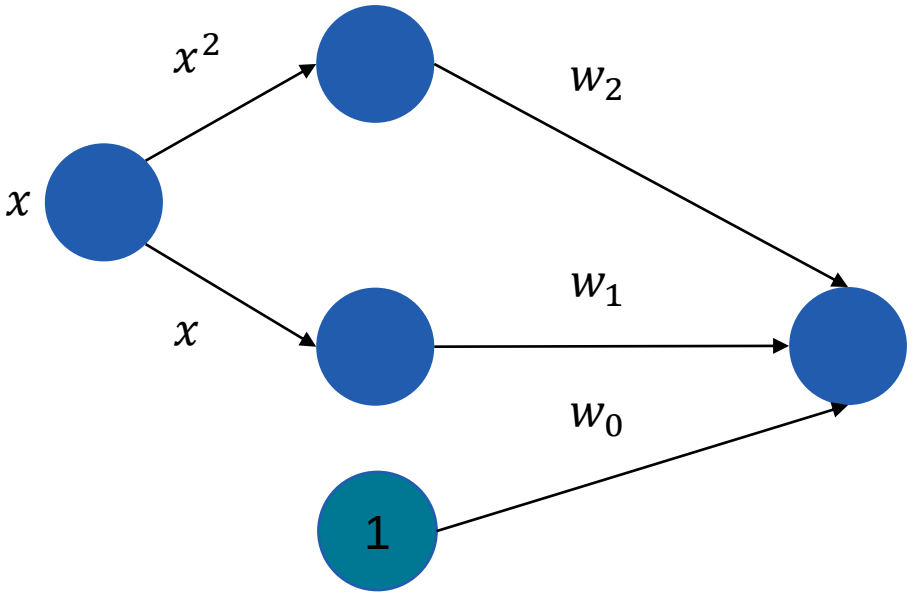
- Optimale Lösung für $\min_{\boldsymbol{w}} \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i, \boldsymbol{w}))^2$ durch Normalgleichungen.



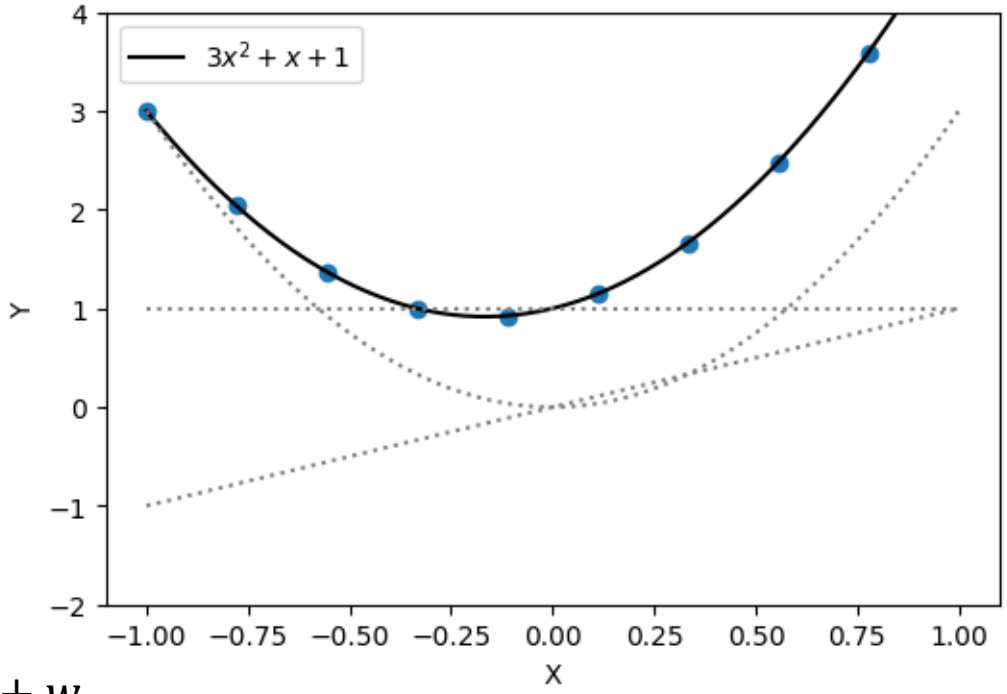
Beispiel: Polynomiale Regression

Merkmalstransformation

$$\boldsymbol{\phi}(x) = (\phi_1(x), \phi_2(x))^T = (x, x^2)$$

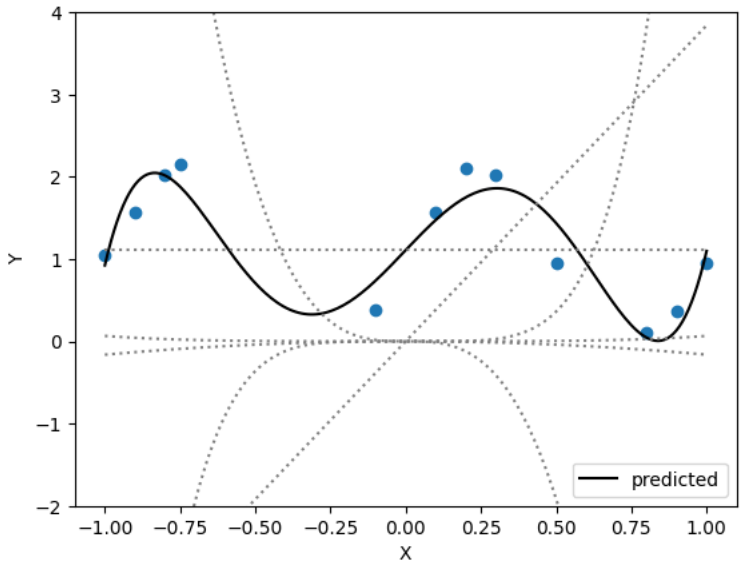


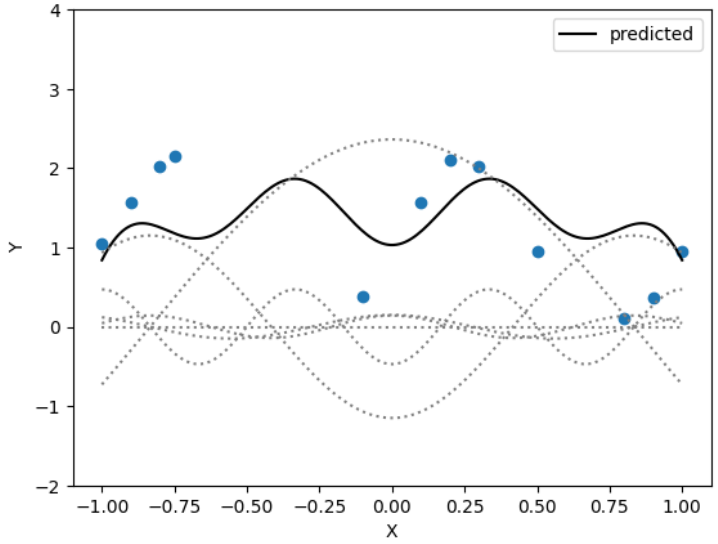
$$f(x, \boldsymbol{w}) = w_1 x + w_2 x^2 + w_0$$

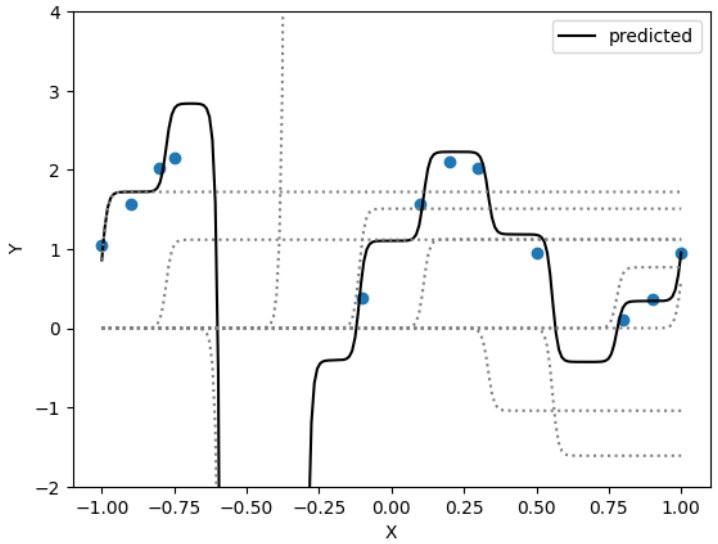


Beispiele von Basisfunktionen

- Polynome $\phi_i(x) = x^i$
- Fourierbasis $\phi_{2i-1}(x) = \sin(2\pi i\omega x)$, $\phi_{2i}(x) = \cos(2\pi i\omega x)$
- Sigmoid Funktionen $\sigma_i(x) = \frac{1}{1+e^{-x}}$



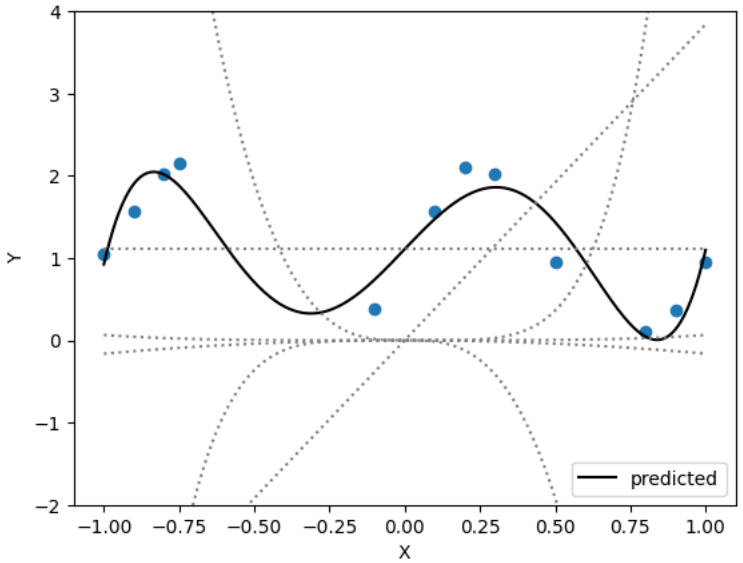


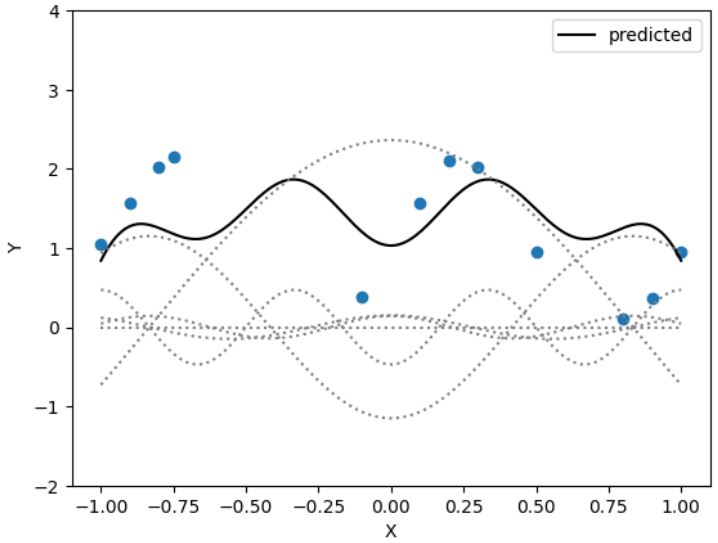


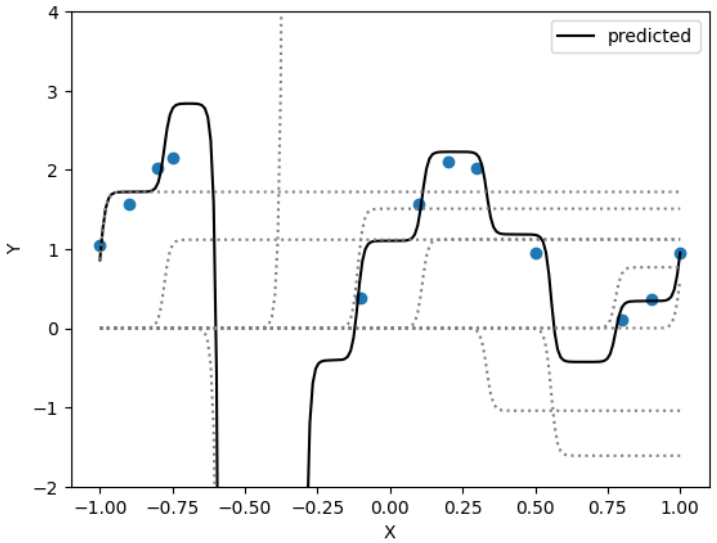
Diskussion

Gegeben Datenpunkte (blau) und Vorhersagen (schwarze Linie) mit verschiedenen Basisfunktionen (graue Linien)

- Welche Vorhersage würden Sie bevorzugen? Weshalb?
- Sind die benutzten Basisfunktionen sinnvoll gewählt für die Daten?
- Was sind Eigenschaften, die gute Vorhersagen haben sollte?

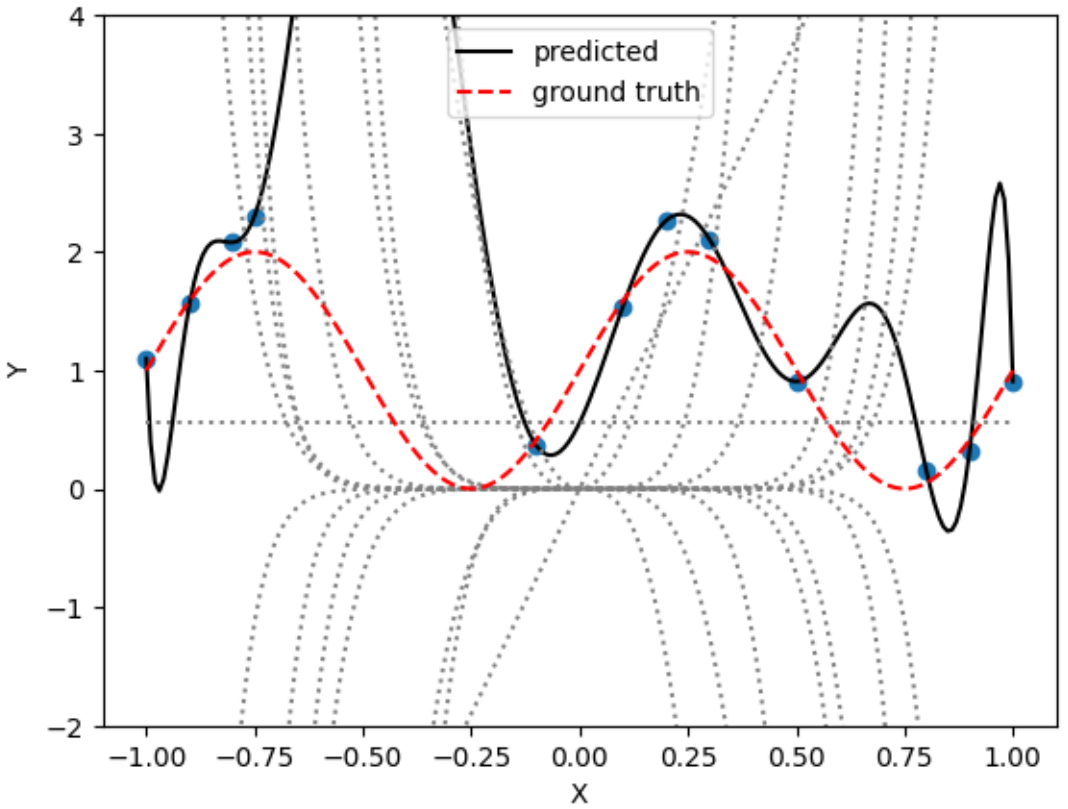






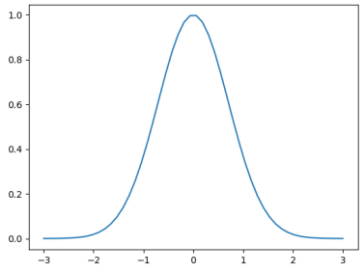
Probleme von obigem Ansatz

- Schwierig, korrektes Modell (Basisfunktionen) zu finden
- Basisfunktionen sind global definiert
  - Jeder Punkt beeinflusst Lösung
- Anzahl Basisfunktionen muss im Voraus festgelegt werden – Unabhängig von Daten



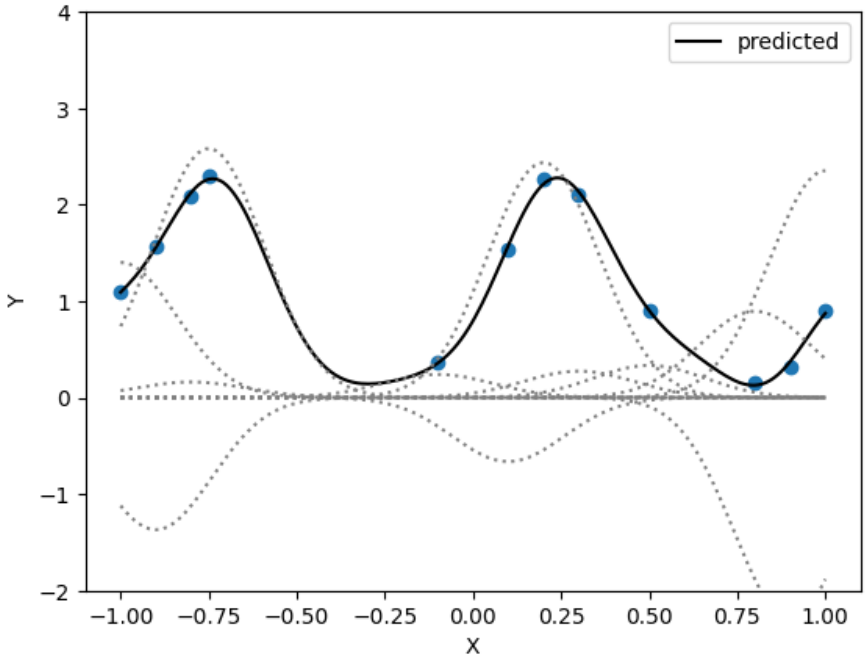
Lösungsansatz: Radiale Basisfunktionen (RBF)

$$\phi_i(x) = \exp\left(\frac{-\|x - \mu_i\|^2}{s^2}\right)$$



Parameter:

- μ_i - Mean der Funktion
- s^2- Breite der Funktion



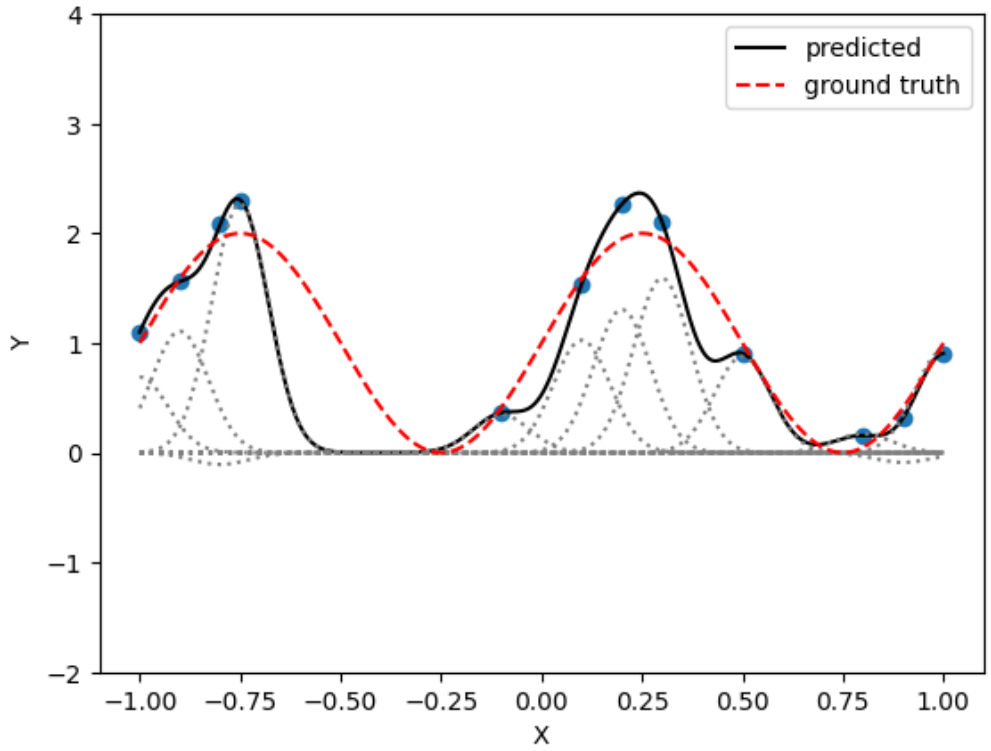
- **Lokal**: Wert nahe 0 weit Weg von Zentrum
  - Vorhersagen werden am stärksten von benachbarten Punkten beeinflusst.
- **Datenabhängig**:
  μ_i kann als Zentrum der Daten gewählt werden
  → Mehr Daten führen zu komplexeren Erklärungen

→ Jupyter-Notebook

Overfitting

Modell lernt nicht nur relevantes Muster, sondern auch Rauschen.

- Tritt bei “komplexen” Modellen auf.



Regularisierung

Einfache Lösungen sollen bevorzugt werden

Einfache Umsetzung: Ridge regression

- Idee: Bestrafe Parameter mit grossen Werten

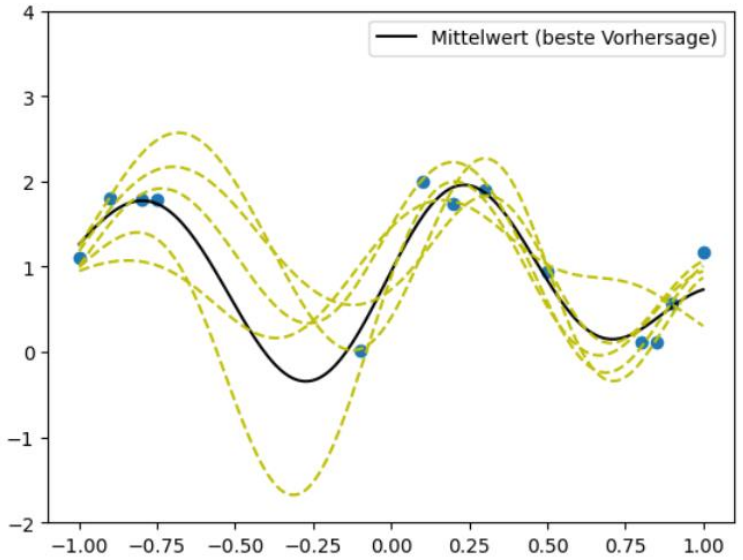
$$\min_{\boldsymbol{w}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x, \boldsymbol{w}))^2 + \lambda \|\boldsymbol{w}\|^2$$

- Lösung: Gegeben durch lineares Gleichungssystem
  (ähnlich Normalgleichungen)

Quelle: Wikipedia

Einschub: Bayes'sche Lineare regression

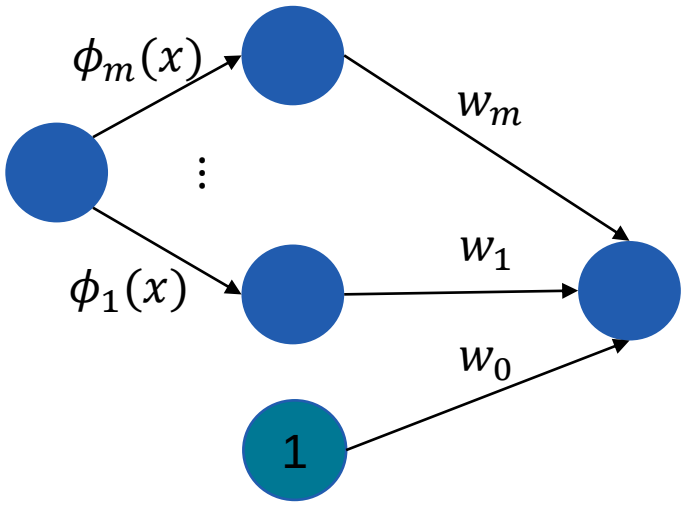
- Definiere Wahrscheinlichkeitsverteilung $w_i \sim N(0, \alpha), i = 1 \ldots, n$ auf Parameter
  - Bevorzugt wie Ridge regression kleine Parameter
- Posteriorverteilung der Parameter gegeben der Daten ist wieder Normalverteilt: $p(\boldsymbol{w}|D) \sim N(\mu, \Sigma)$
  - Mittelwert μ (beste Vorhersage) und Kovarianz Σ (Unsicherheit) durch Gleichungssystem bestimmt.

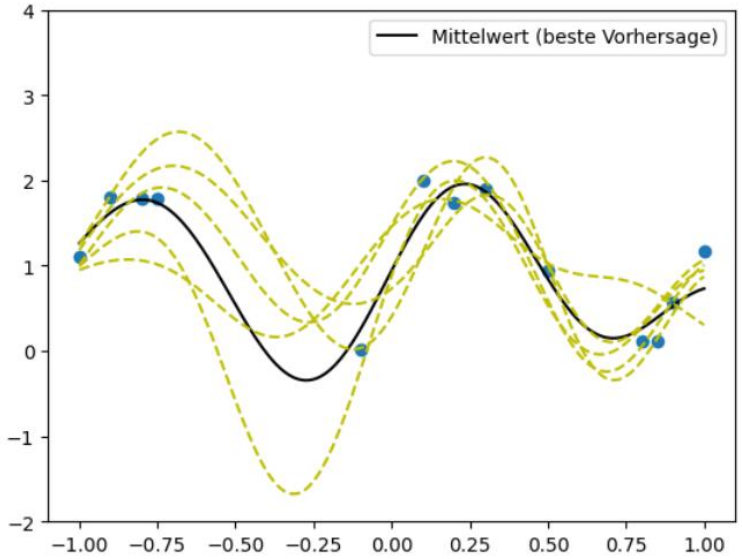


→Notebook

Lineare Basisfunktionsmodelle: Zwischenfazit

- ✓ Komplexe Zusammenhänge in Daten können gelernt werden.
- ✓ Verschiedene Möglichkeiten, Funktionen zu wählen. Lösung jeweils durch Gleichungssystem gegeben.
- ✓ Komplexität der Funktionen regulierbar und kann von Daten abhängen.
- ✓ Verteilung über mögliche Erklärungen der Daten mit Bayes'schen linearen Regression.





Limitierungen von RBFs (und anderen Basisfunktionmodellen)

- Funktionen sind nicht angepasst auf Daten.
- In hochdimensionalen Räumen werden zu viele Basisfunktionen benötigt.
  - Lösung nicht mehr effizient berechenbar.

Quelle: Hubblesite.org

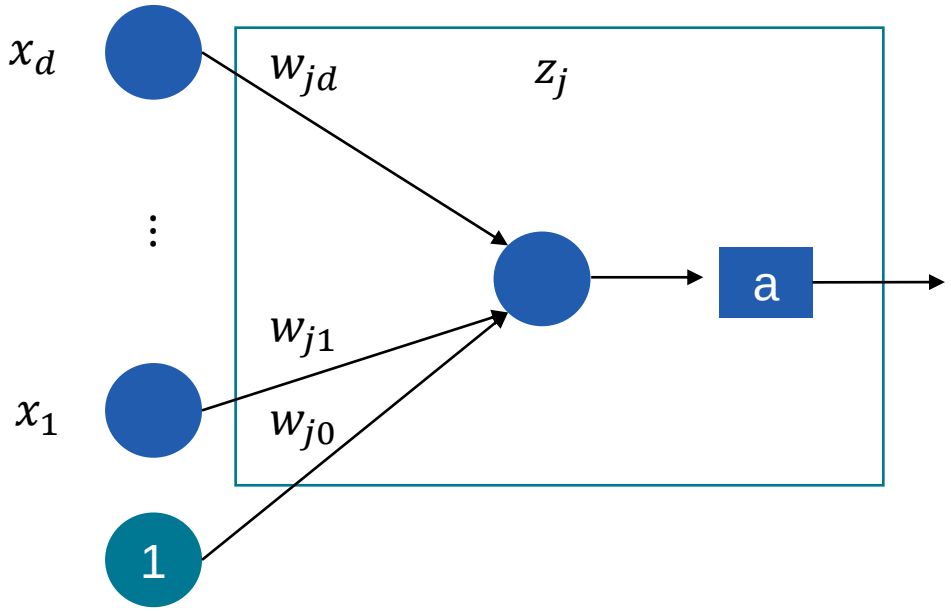
Neuronale Netze:

- Basisfunktionen werden aus Daten gelernt und somit an Daten angepasst.

Neuronen: Lernbare Basisfunktionen

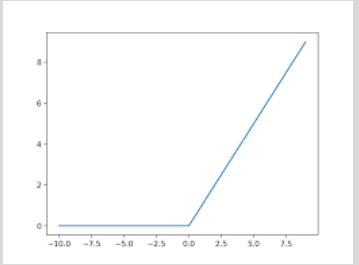
Neuron: Linearkombination der Eingaben, gefolgt von einer Aktivierungsfunktion

$$z_j(x, \boldsymbol{w_j}) = a(\sum_{i=1}^{n} w_{ji} x_i + w_{j0})$$

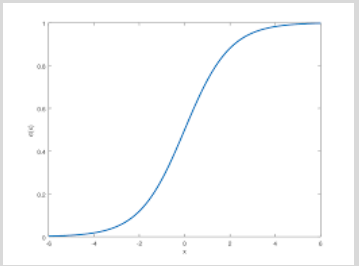


Aktivierungsfunktionen

Relu

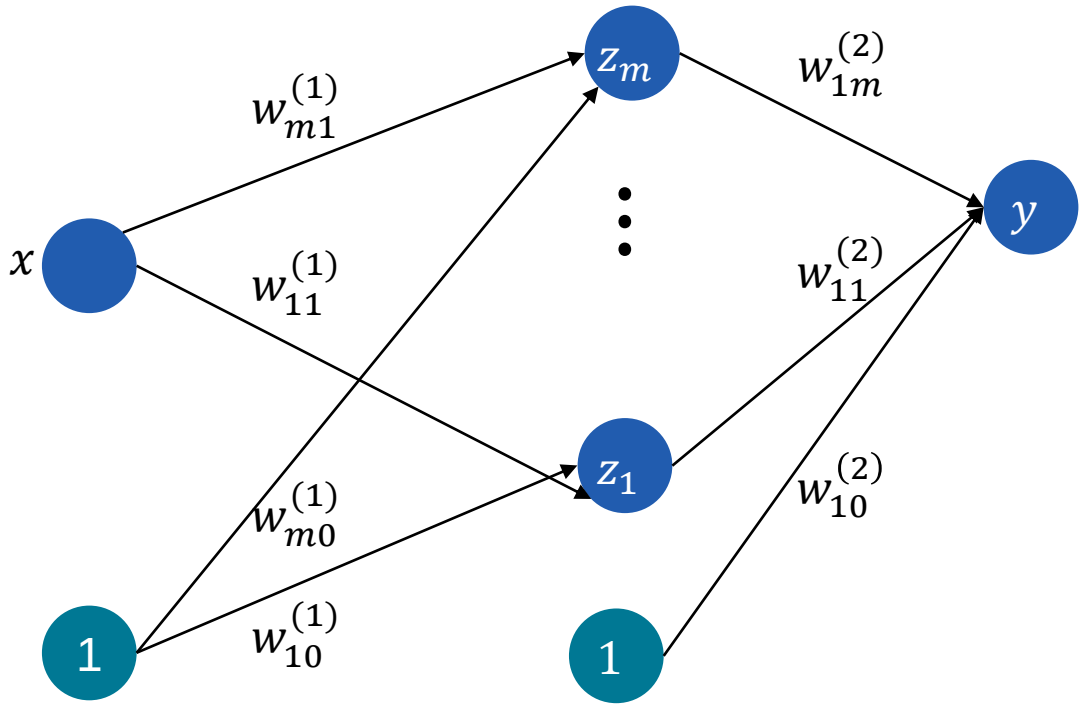


Sigmoid



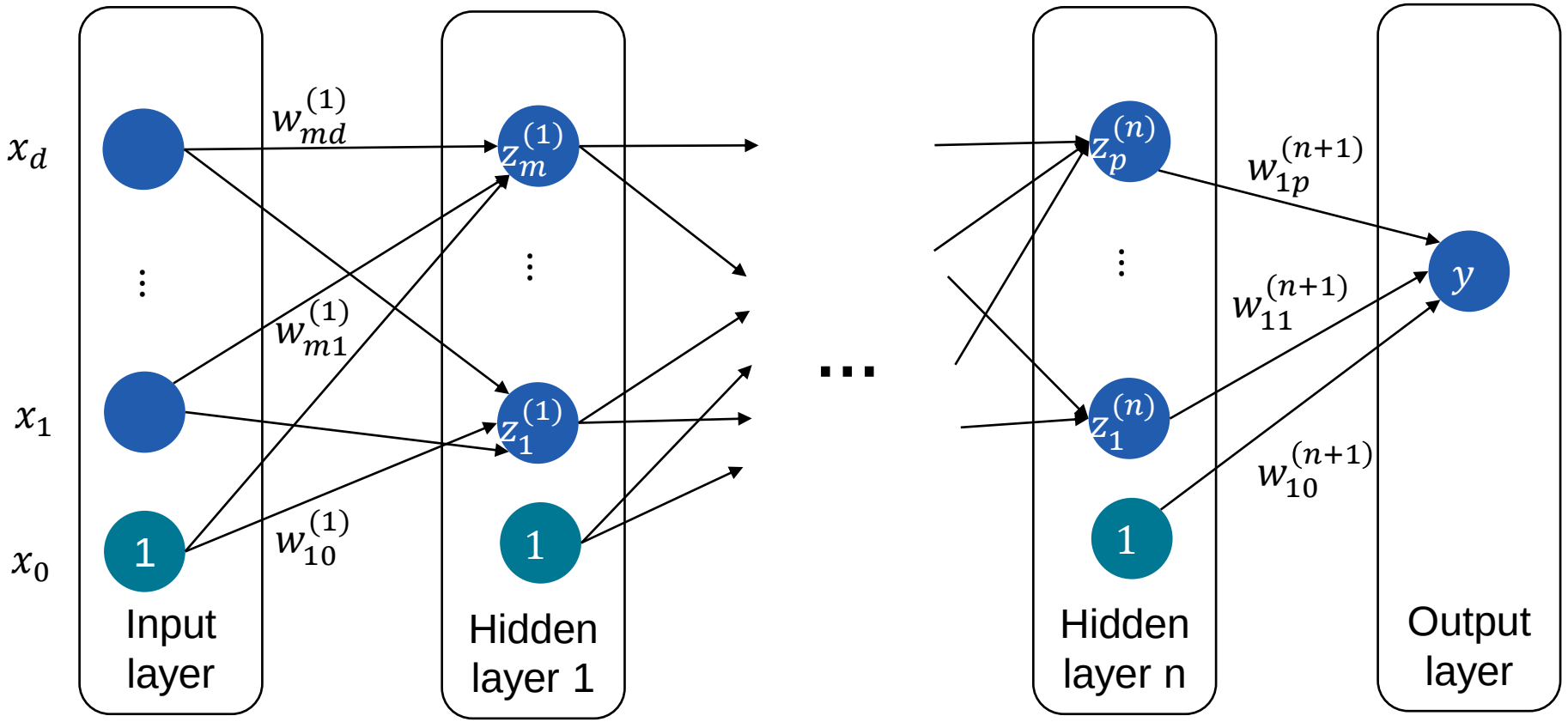
... ...

Einfaches neuronales Netz



- $z_1, \ldots, z_m$ entsprechen gelernten Basisfunktionen $\phi_1, \ldots, \phi_m$
- Wegen nichtlinearer Aktivierungsfunktion keine analytische Lösung möglich

Multilayer neural networks



- Beliebig tiefe Verschachtelung möglich
- Beliebig hochdimensionale Eingaben möglich
- In jeder Ebene werden Merkmale etwas besser an Problem angepasst

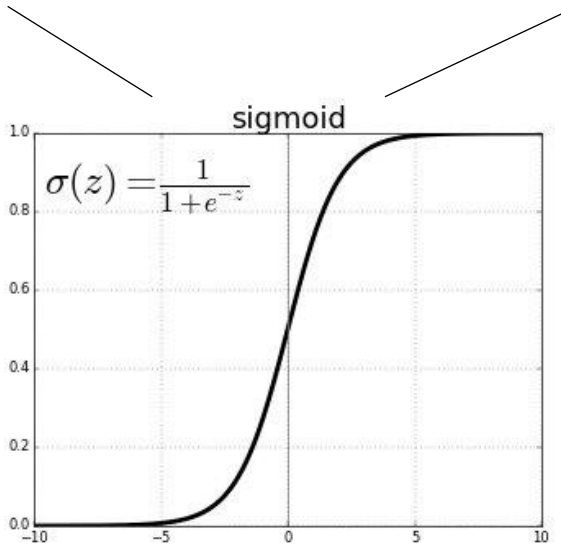
Multilayer neural networks für Klassifikation

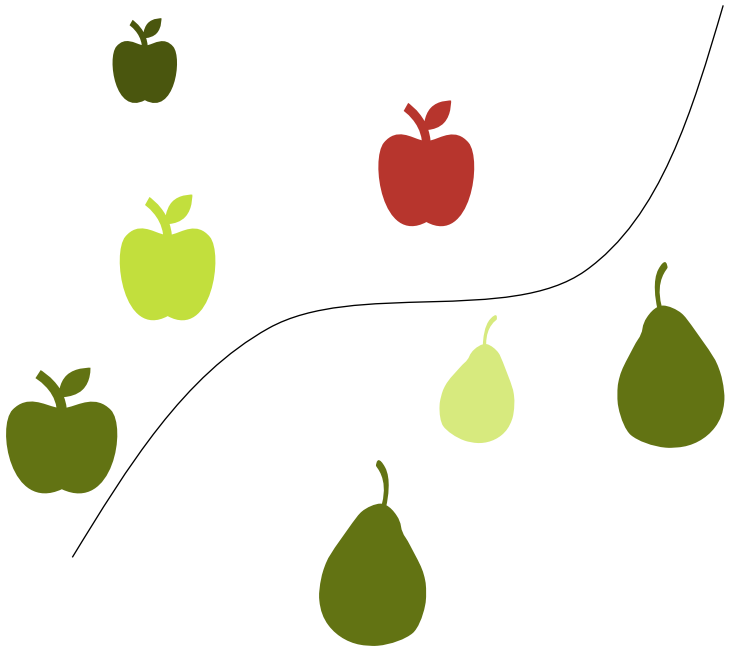
- Alles bleibt gleich aber:
  - Fehlerquadrate keine ideale Verlustfunktion
  - Bessere Wahl: Cross entropy Loss

Cross Entropy Loss für $y \in \{0, 1\}$

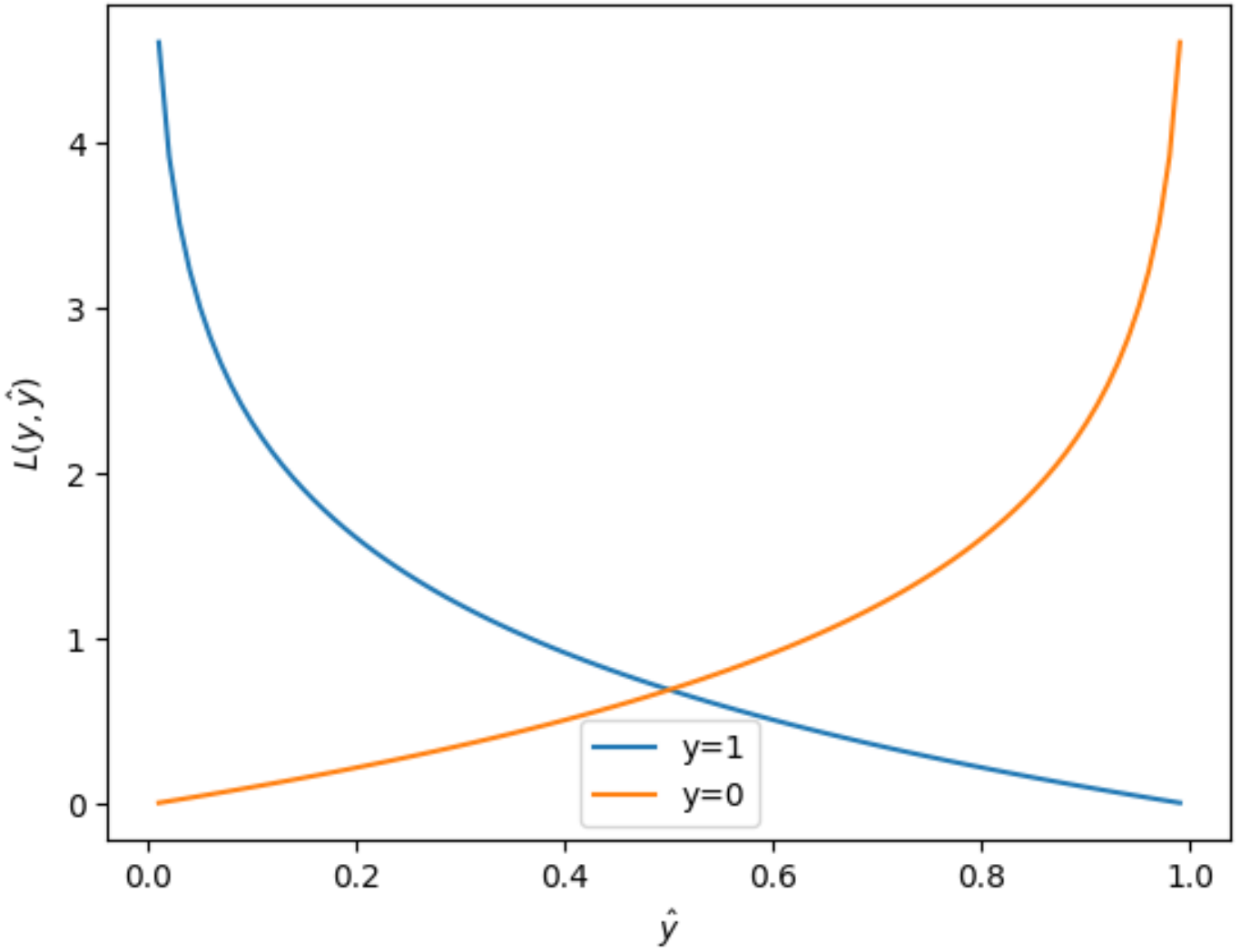
$$L(y, \hat{y}) = L\left(y, \sigma\left(f(x, \boldsymbol{w})\right)\right)$$

$$= -y \log(\sigma(f(x, \boldsymbol{w}))) - (1 - y)\log(1 - \sigma(f(x, \boldsymbol{w})))$$





Verhalten – Cross entropy loss



Übungen und Ausblick

Übungen

- Anwendung von linearer Regression und Neuronalen Netzen in Scikit Learn

Nächstes Mal

- Wie trainieren wir Modelle?
- Anwendungen von Neuronalen Netzen – Deep Learning