08.12.20

Einführung in Data Science und maschinelles Lernen mit R

Einführung in maschinelles Lernen

OPENCAMPUS.sh

- Charakteristika des maschinellen Lernens
- Definition der linearen Regression
- Kostenfunktionen
- Optimierungsfunktionen
- Overfitting
- Regularisierung

ML ALGORITHMEN

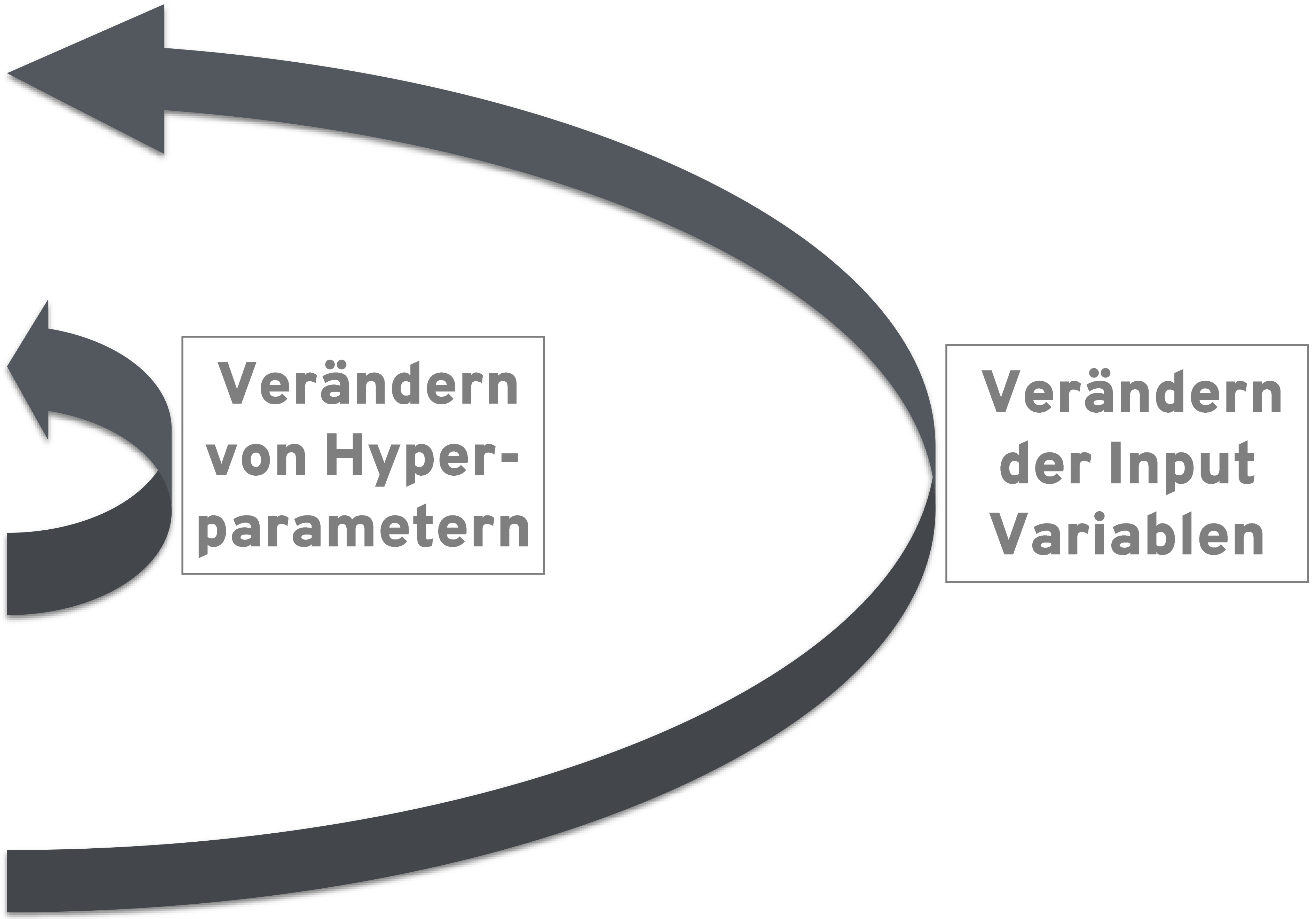
(1) Wahl eines Prognose-Modells

(2) Teilung der vorhanden Daten in einen Trainings- und einen Validierungsdatensatz

(3) Optimierung der Modellparameter anhand des Trainingsdatensatzes

(4) Überprüfung des Modell anhand des Validierungsdatensatzes

(5) Erweiterung/Verbesserung des vorhandenen Datensatzes



PROGNOSEMODELLE

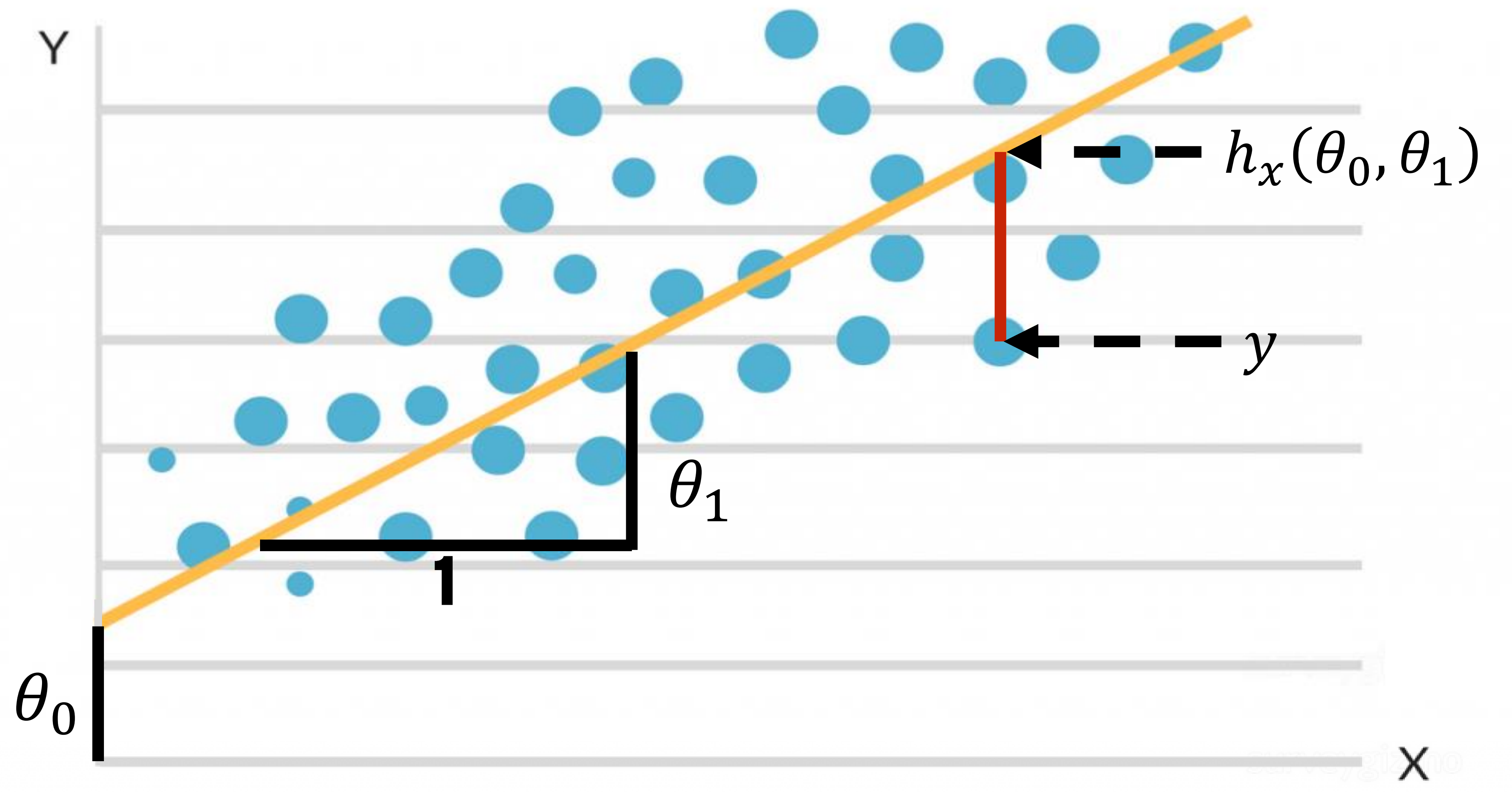
Modelle der kommenden Sessions:

- **Lineares Modell**
- **Support Vektor Maschine**
- **(Tiefes) Neuronales Netz**

LINEARES MODELL

x ist die beobachtete Eingabe

y ist die beobachtete Ausgabe



Lineares Model:

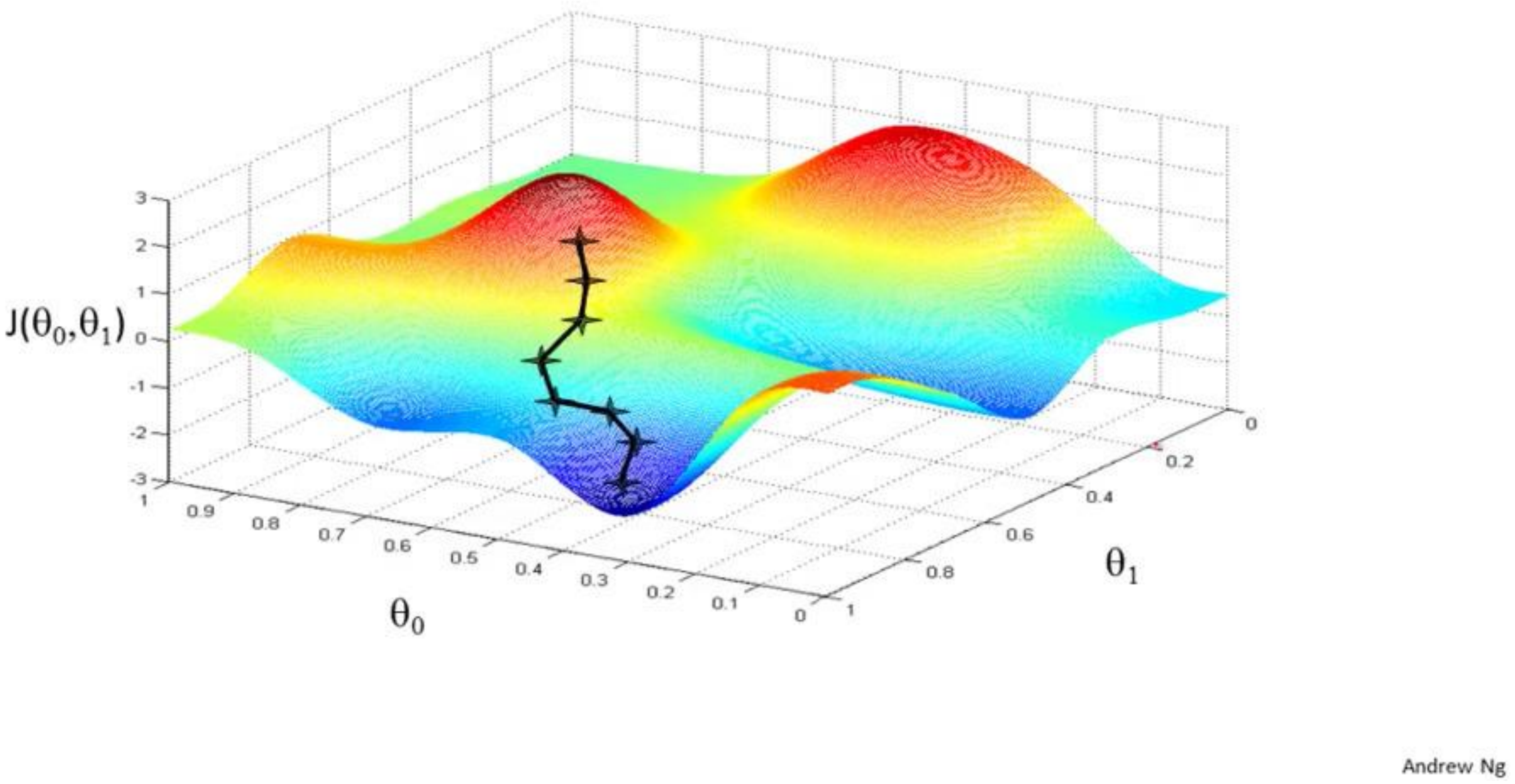
$h_x(\theta_0, \theta_1) = \theta_0 + \theta_1 x$

Kostenfunktion zur Berechnung der optimalen Parameter θ_0 und θ_1:

$$J_x(\theta_0, \theta_1) = \qquad (h_x(\theta_0, \theta_1) - y)$$

Mean Squared Error (MSE)

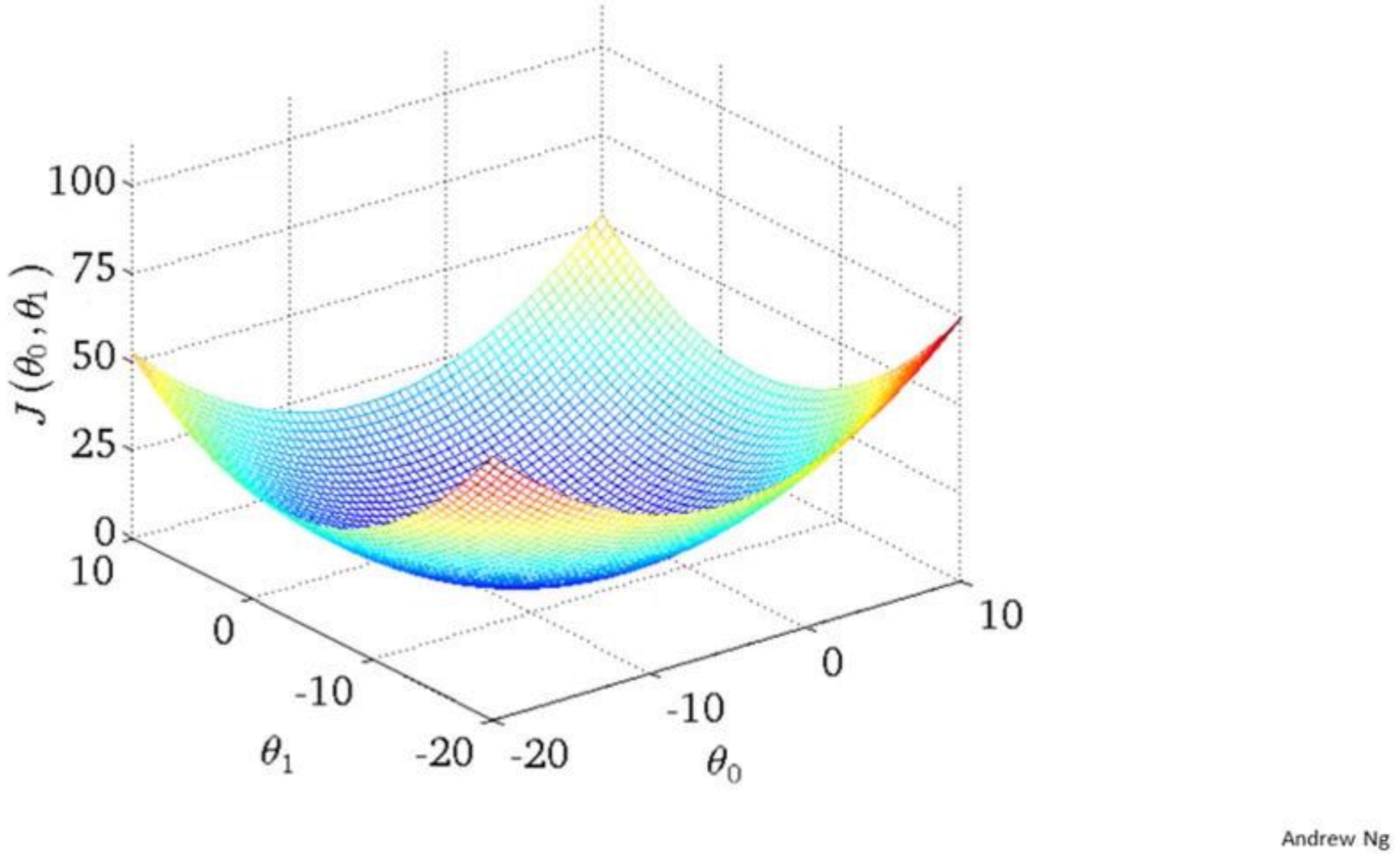
MINIMIERUNG DER KOSTENFUNKTION



Quelle: https://www.coursera.org/learn/machine-learning

- **Minimierung der Kostenfunktion durch ein iteratives Verfahren, dass sich dem Minimum annähert.**
- **Die Schrittgröße für die Annäherung kann durch den „Learning Parameter“ Alpha kontrolliert werden.**

GRADIENT DESCENT FUNKTION



Quelle: https://www.coursera.org/learn/machine-learning

- **Verfahren, das für viele Kostenfunktionen eingesetzt werden kann.**
- **Für lineare Modelle ist die Kostenfunktion konvex und besitzt keine lokalen Minima.**

SCHÄTZEN EINES LINEAREN MODELLS

```
library(readr)
house_pricing <-
read_csv("https://raw.githubusercontent.com/opencampus-sh/sose20-
datascience/master/house_pricing_train.csv")

mod <- lm(price ~ sqft_lot15 + as.factor(condition), house_pricing)
summary(mod)
```

ERGEBNIS DES LINEAREN MODELLS

```
Call:
lm(formula = price ~ sqft_lot15 + as.factor(condition), data = house_pricing)

Residuals:
    Min      1Q  Median      3Q     Max 
-795388 -214555  -85989  101761 7183941 

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)    
(Intercept)            3.261e+05  7.049e+04   4.625 3.77e-06 ***
sqft_lot15             1.138e+00  1.035e-01  11.002  < 2e-16 ***
as.factor(condition)2 -2.305e+04  7.690e+04  -0.300 0.764379    
as.factor(condition)3  2.031e+05  7.057e+04   2.878 0.004008 ** 
as.factor(condition)4  1.800e+05  7.070e+04   2.546 0.010915 *  
as.factor(condition)5  2.762e+05  7.118e+04   3.880 0.000105 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 366300 on 17284 degrees of freedom
Multiple R-squared:  0.01409,	Adjusted R-squared:  0.0138 
F-statistic: 49.39 on 5 and 17284 DF,  p-value: < 2.2e-16
```

KENNWERTE DER REGRESSION

p-Wert

- Signifikanzwert
- Wahrscheinlichkeit, dass der zugehörige Wert in der Regression gleich null ist.
- Werte unter 0,05 (entspricht 5%) werden üblicherweise als signifikant betrachtet

(Adjustiertes) R^2

- Kennzahl zur Beurteilung der Güte einer Regression
- Wert zwischen 0 und 1, der der dem Anteil der erklärten Variation entspricht (1 entspricht 100%):

$$\boldsymbol{R^2} = \frac{\text{Erklärte Varianz}}{\text{Gesamtvarianz}}$$

- Das adjustierte R^2 bestraft das Hinzufügen zusätzlicher Variablen/Parameter

DataCamp Tutorial zur Linearen Modellierung in R :
https://www.datacamp.com/community/tutorials/linear-regression-R

EIGENSCHAFTEN DES LINEAREN MODELLS

- Die Funktion lm() liefert optimierte Parameter für das lineare Modell ohne Regularisierung (Angabe eines Lernparameters ist hier nicht nötig)

  → Für einfache Modelle ist es einfach optimierte Parameter zu erhalten.

- Einfacher zu schätzende Modelle haben stärkere Annahmen über den Zusammenhang der Variablen (hier linearer Zusammenhang)

  → Die optimale Kodierung/Kategorisierung der Variablen entsprechend der Annahmen ist sehr wichtig und ggf. schwierig.

AUFGABEN

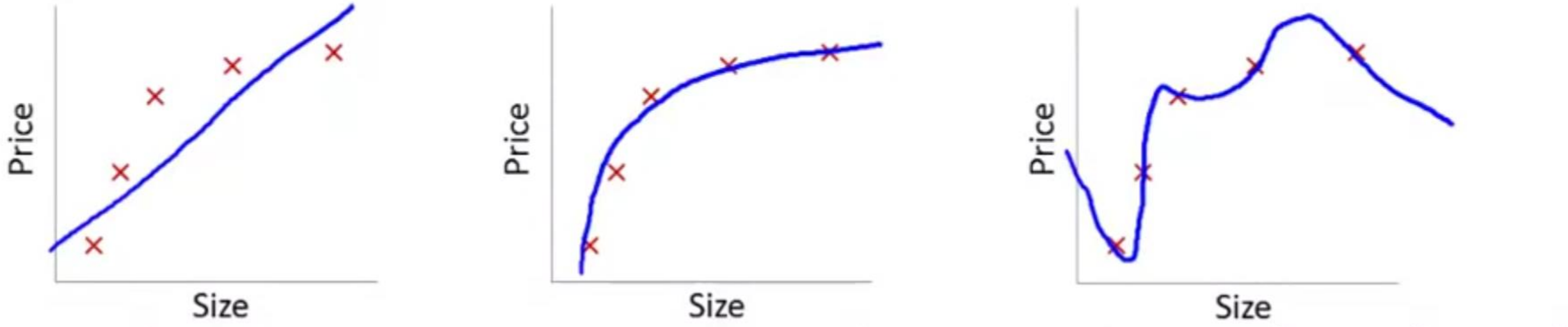
- Stellt für Euren Datensatz eine Modellgleichung auf, die das adjustierte R^2 auf Basis eines linearen Modells maximiert.

Schaut das folgende Video an:

- Overfitting (10 Minuten): https://www.coursera.org/lecture/machine-learning/the-problem-of-overfitting-ACpTQ

OVERFITTING

Example: Linear regression (housing prices)



$\rightarrow \theta_0 + \theta_1 x$ "Underfit" "High bias"

$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2$ "Just right"

$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ "Overfit" "High variance"

Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).

Computer Science Engineering. (2017). *3.4.1 The Problem of Overfitting by Andrew Ng.*
Abgerufen von https://www.youtube.com/watch?v=OSd30QGMl88

BEISPIEL ZU OVERFITTING

```
---
title: "Linear Regression"
output: html_notebook
---

```{r}
# Importing Function Packages
library(dplyr)
library(readr)
library(lubridate)
library(broom)
library(Metrics)
```


```{r}
# Importing Training and Test Data
house_pricing_train <- read_csv("https://raw.githubusercontent.com/opencampus-sh/ws1920-datascience/master/house_pricing_train.csv")
house_pricing_test <- read_csv("https://raw.githubusercontent.com/opencampus-sh/ws1920-datascience/master/house_pricing_test.csv")
```


```{r}
# Estimating (Training) Models
mod1 <- lm(price ~ bathrooms, house_pricing_train)
mod2 <- lm(price ~ as.factor(bathrooms), house_pricing_train)
mod3 <- lm(price ~ as.factor(bathrooms) + as.factor(zipcode), house_pricing_train)
mod4 <- lm(price ~ as.factor(bathrooms) + as.factor(zipcode) + condition, house_pricing_train)
mod5 <- lm(price ~ as.factor(bathrooms) + as.factor(zipcode) + as.factor(condition), house_pricing_train)
mod6 <- lm(price ~ as.factor(bathrooms) + as.factor(zipcode) + as.factor(condition) + sqft_living15, house_pricing_train)
mod7 <- lm(price ~ as.factor(bathrooms) + as.factor(zipcode) + as.factor(condition) + sqft_living15 + floors + view + grade +
as.factor(zipcode)*as.factor(bathrooms), house_pricing_train)
```
```

STRATEGIEN UM OVERFITTING ZU VERMEIDEN

Options:

1. Reduce number of features.
   — Manually select which features to keep.
   — Model selection algorithm (later in course).
2. Regularization.
   — Keep all the features, but reduce magnitude/values of parameters θ_j.
   — Works well when we have a lot of features, each of which contributes a bit to predicting y.

Computer Science Engineering. (2017). ***3.4.1 The Problem of Overfitting by Andrew Ng.***
Abgerufen von https://www.youtube.com/watch?v=OSd30QGMl88

REGULARISIERUNG

„Bestrafen“ der Verwendung von Variableninformation im Rahmen der Kostenfunktion

Lineares Modell mit mehreren Variablen x_1, x_2 und vielen möglichen weiteren:

$$\boldsymbol{h_x(\theta_0, \theta_1, \theta_2, \ldots) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots}$$

(mit $\theta_0, \theta_1, \theta_2, \ldots$ als den zu schätzenden Modellparametern)

Kostenfunktion mit Regularisierung:

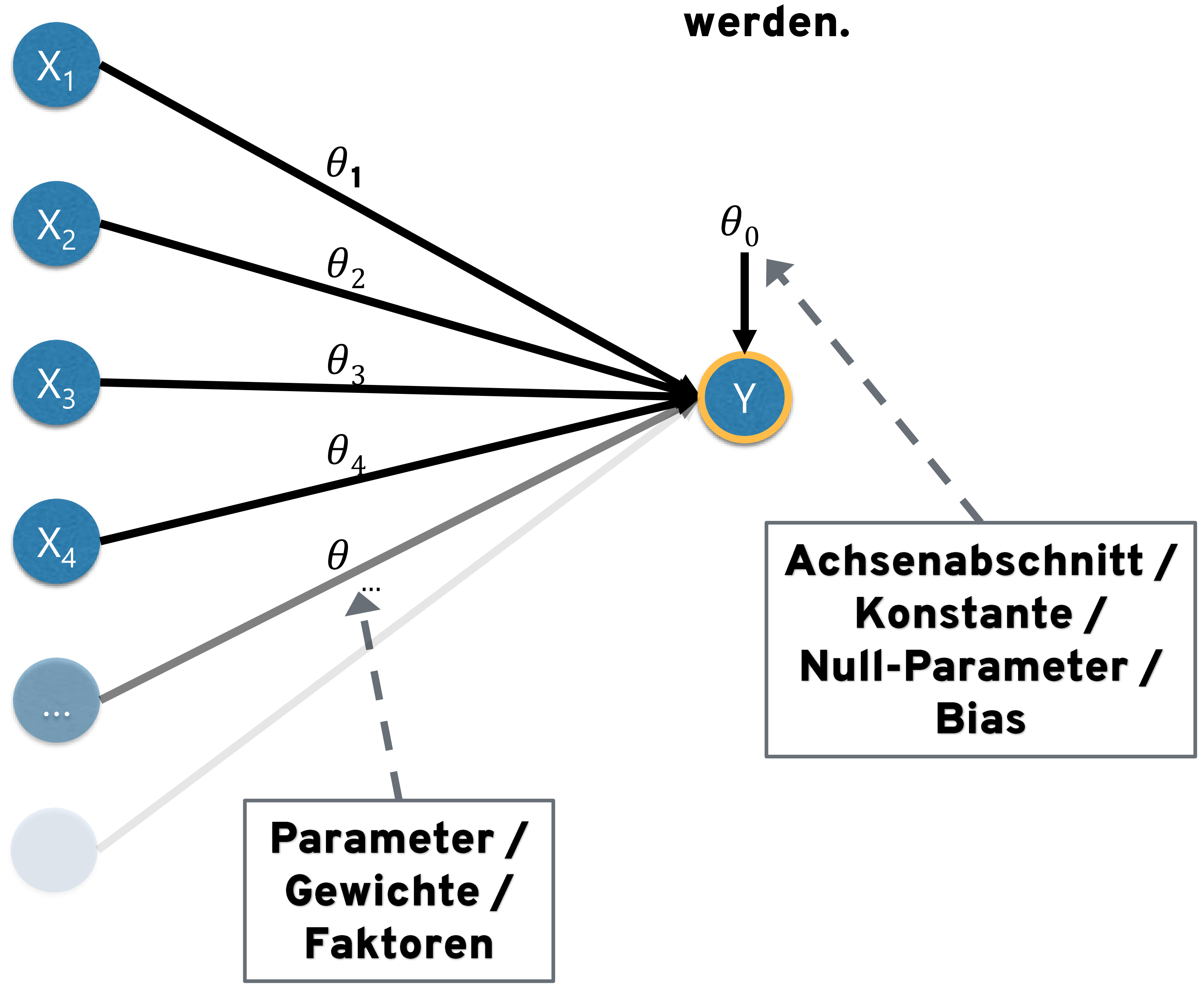
$$\boldsymbol{J_x(\theta_0, \theta_1, \theta_2, \ldots) = \frac{1}{m}\left[\sum_m (h_x(\theta_0, \theta_1, \theta_2, \ldots) - y)^2 + \lambda(\theta_0 + \theta_1 + \theta_2 + \cdots)\right]}$$

(mit λ als Regularisierungsparameter)

ZUSAMMENFASSUNG LINEARE REGRESSION

Input Layer
Elemente sind die Input-Variablen; auch genannt: Input-Features oder Input-Dimensionen.

Output Layer
Nutzt eine *„Aktivierungsfunktion“* (hier lineare Funktion) mit der die Parameter θ der eingehenden Schicht zusammengefasst werden.



- ❑ **Ziel ist es, anhand des Trainingsdatensatzes die Parameter θ der Aktivierungsfunktion für eine bestmögliche Vorhersage des Testdatensatzes zu optimieren.**

- ❑ **Die Verwendung der Gradient Descent Optimierung mit Regularisierung erlaubt, alle Variablen in das Modell eingehen zu lassen und über einen einzelnen Regularisierungsparameter (oder auch Shrinkage-Parameter) den Umfang des Einsatzes der Variablen zu kontrollieren, um so das zur Vorhersage optimale Modell bestimmen zu können.**

AUFGABEN

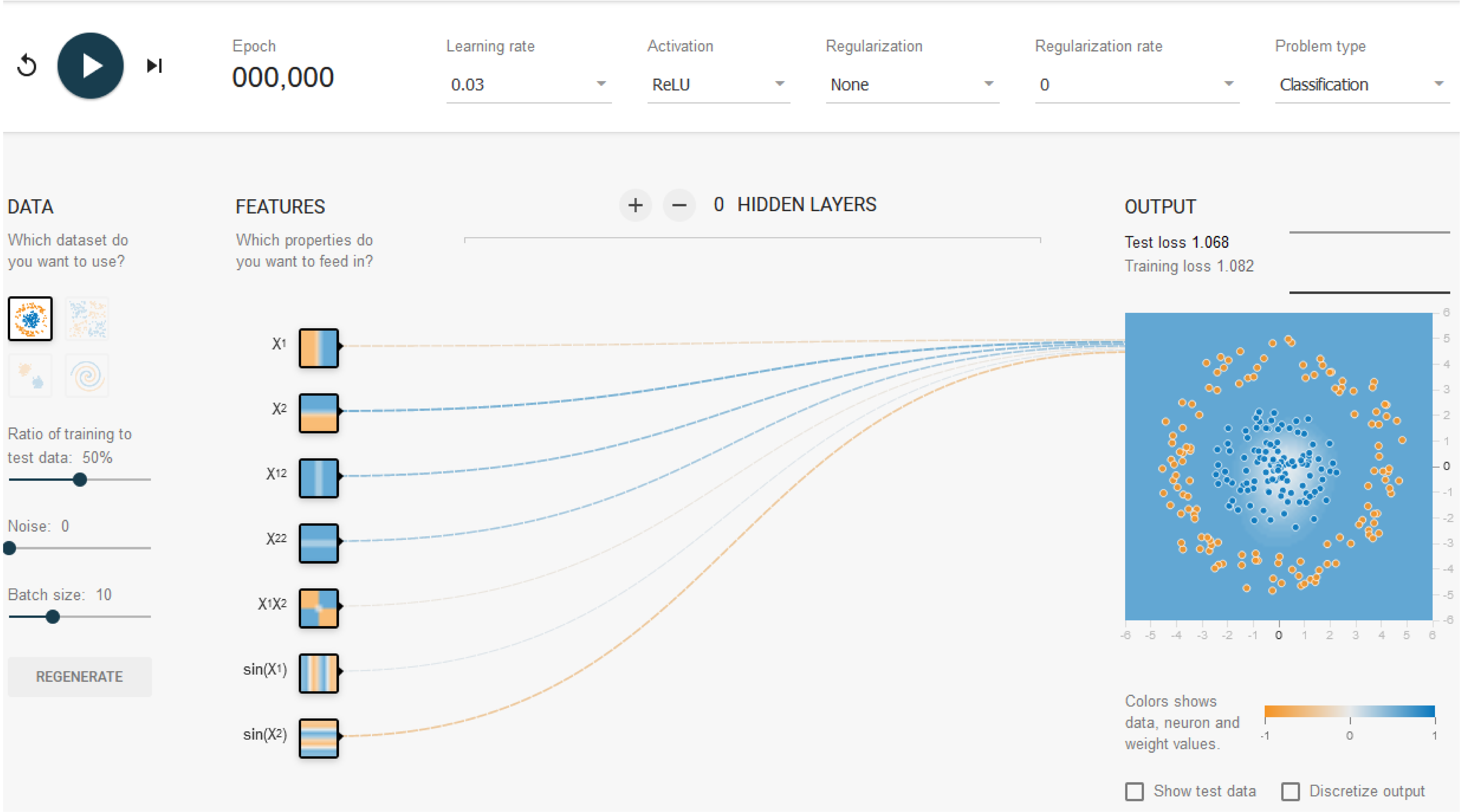
Schaut Euch folgendes Tool an: https://playground.tensorflow.org/

Definiert eine lineare Regression ohne Hidden Layer

- Welche Datensätze könnt Ihr mit der linearen Regression erfolgreich vorhersagen?
- Inwieweit könnt Ihr die Ergebnisse hinsichtlich der verwendeten Feature interpretieren?

Definiert zwei Hidden Layer und probiert die Anzahlen der Neuronen so zu ändern, dass Ihr den spiralförmigen Datensatz vorhersagen könnt.

- Welche Verteilungen könnt Ihr erfolgreich vorhersagen?
- Inwieweit könnt Ihr die Ergebnisse hinsichtlich der verwendeten Feature interpretieren?



OPENCAMPUS.sh

WICHTIGE KONZEPTE

- Aktivierungsfunktion
- Kostenfunktion
- Optimierungsfunktion (Minimierung der Kostenfunktion)
- Lernrate (Eigenschaft der Optimierungsfunktion)
- Regularisierung (Bestrafung der Verwendung von Variablen/großen Parametern)

RELEVANZ VON KATEGORISIERUNGEN