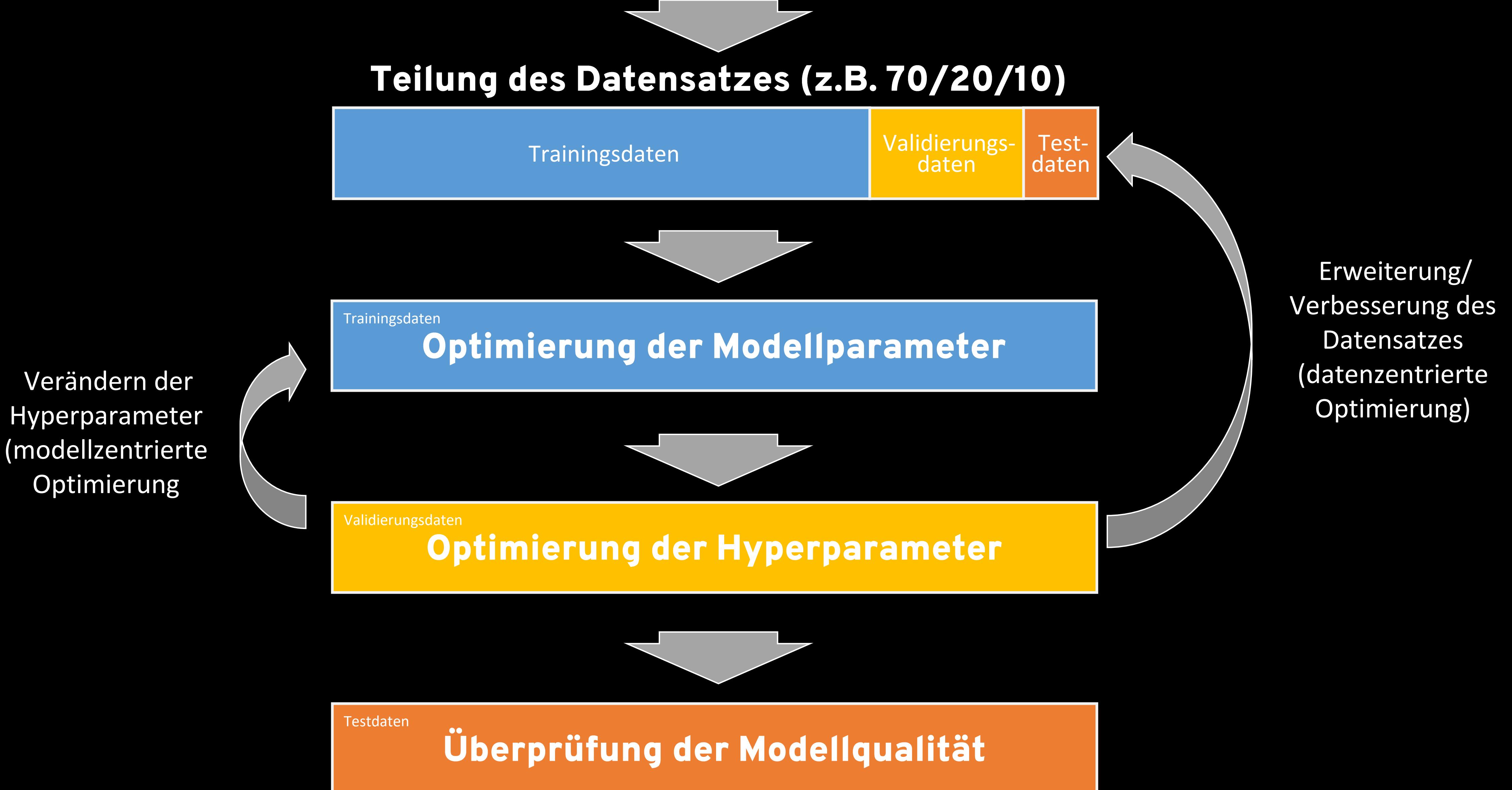


# Einführung in Data Science und maschinelles Lernen

EINFÜHRUNG IN MASCHINELLES  
LERNEN

- **Vorgehen zum Trainieren von maschinellen Lernalgorithmen**
- **Definition der linearen Regression**
- **Kostenfunktionen**
- **Optimierungsfunktionen**

# Wahl eines Prognosemodells



# **TEILUNG EINES DATENSATZES (OHNE ZEITREIHEN!)**

- (1) Mischung der Reihenfolge (Randomisierung)**
- (2) Definition der Größen von Trainings-,  
Validierungs- und Testdatensatz in Prozent**
- (3) Teilung des Datensatzes**

# **TEILUNG EINES DATENSATZES MIT ZEITREIHEN**

- (1)    Ordnen der Zeitreihe**
- (2)    Definition der Zeitreihenfenster für  
            Validierungs- und Testdatensatz**
- (3)    Teilung des Datensatzes**

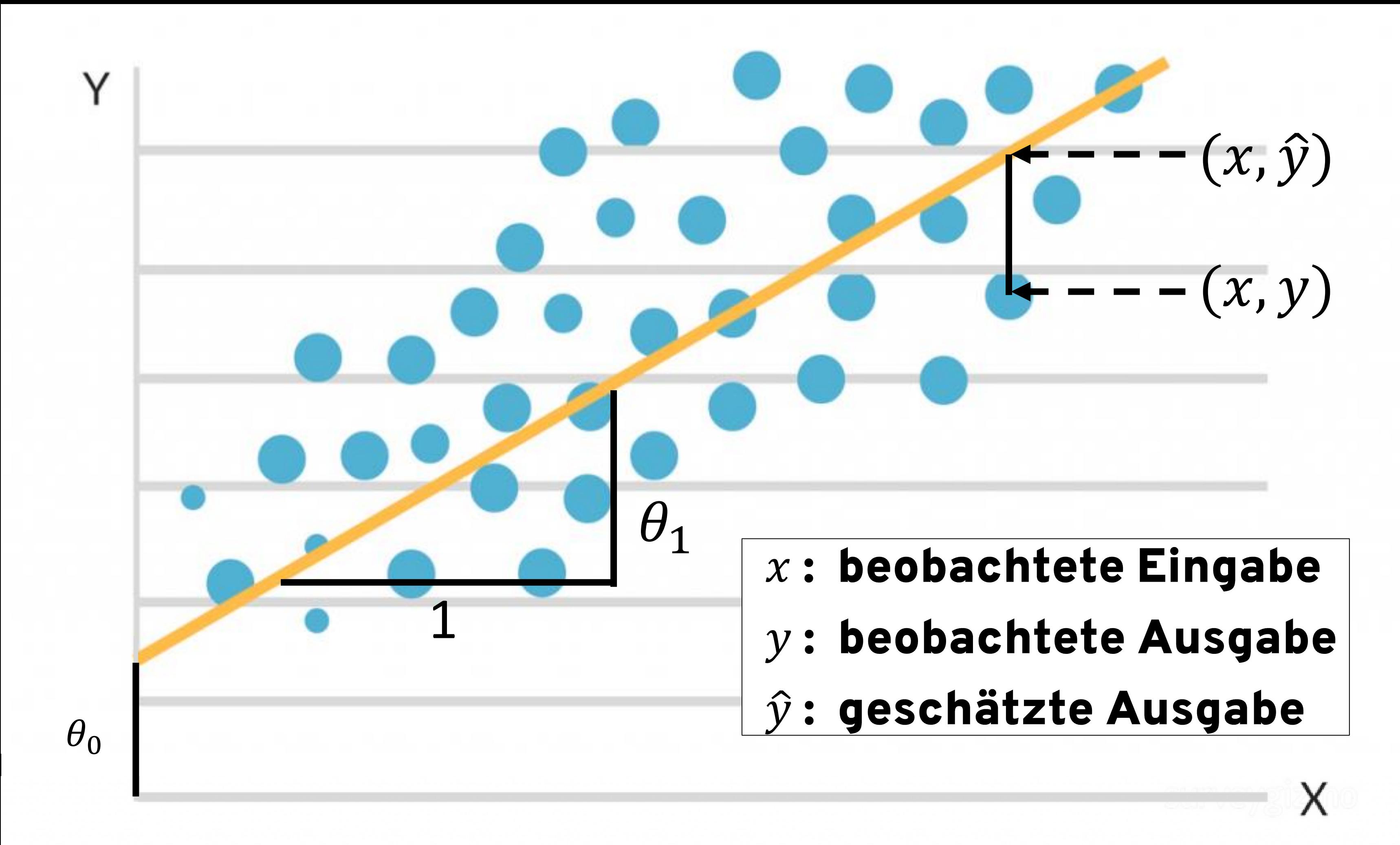
# PROGNOSEMODELLE

In diesem Kurs behandelte:

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

# LINEARES MODELL

$$\hat{y} = \theta_0 + \theta_1 x$$
$$= h_x(\theta_0, \theta_1)$$



# KOSTENFUNKTION (LOSS FUNCTION)

Zur Berechnung der Funktion mit den optimalen Parametern  
 $\theta_0$  und  $\theta_1$ :

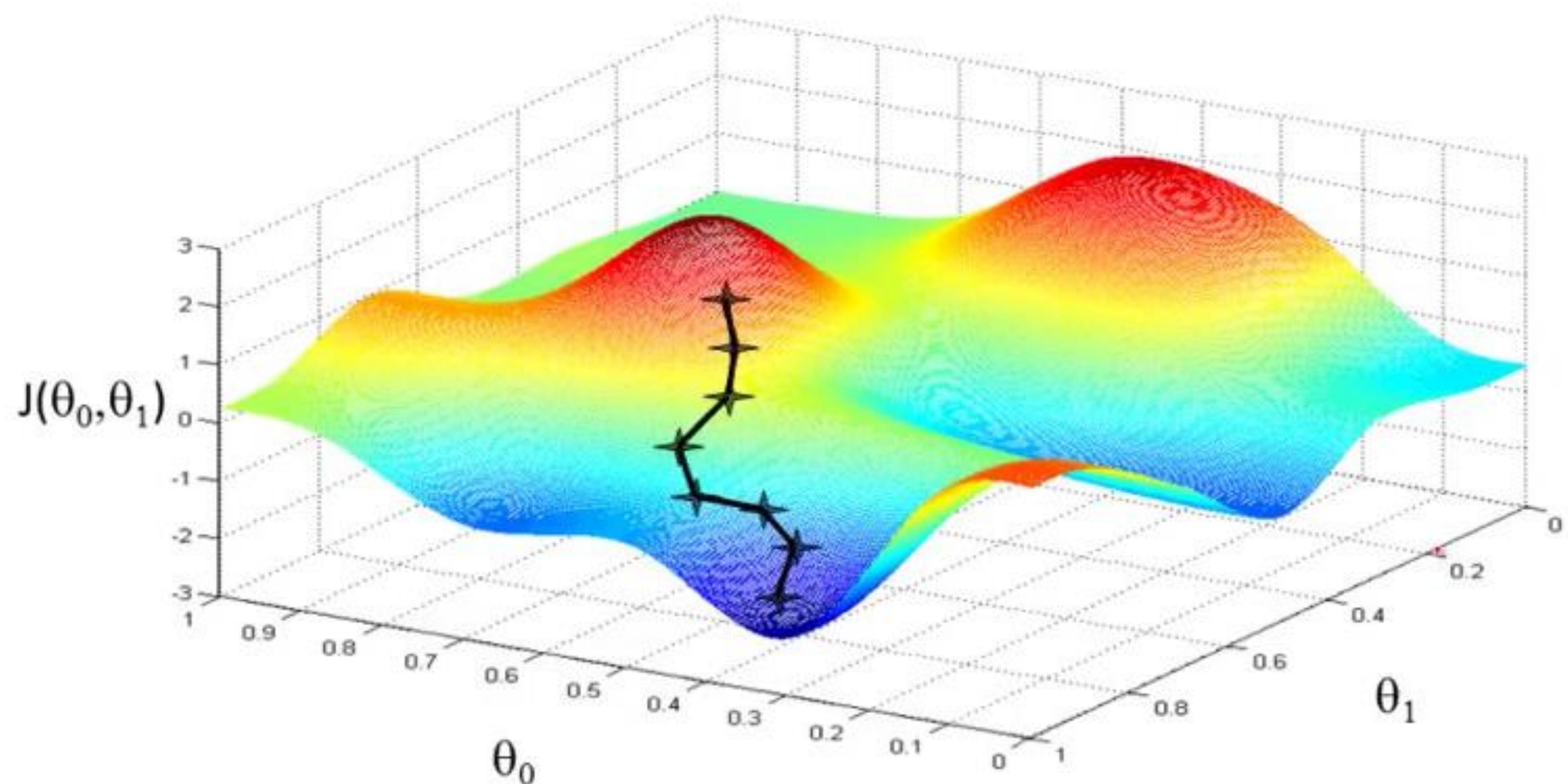
$$J_x(\theta_0, \theta_1) = h_x(\theta_0, \theta_1) - y$$

**Mean Absolute Error (MAE)**

$$J_x(\theta_0, \theta_1) = \frac{1}{m} \sum (h_x(\theta_0, \theta_1) - y)^2$$

**Mean Squared Error (MSE)**

# OPTIMIERUNGSFUNKTION (OPTIMIZER)



- Iteratives Verfahren (Gradient Descent), um das Minimum der Kostenfunktion zu finden.
- Die Lernrate („Learning Parameter“) beschreibt die Schrittgröße zur Annäherung an das Minimum.

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

# OPTIMIERUNG DER MODELLPARAMETER

## *Forward Propagation*

- Berechnung des vorhergesagten Wertes auf Basis der aktuellen Modellgleichung
- Berechnung der Kosten bzw. des Loss

## *Backward Propagation*

- Berechnung des Gradienten (d.h. aller partiellen Ableitungen), um die Richtung des Minimums zu bestimmen.
- Anpassung der Modellparameter im Ausmaß der definierten Lernrate:

$$\text{Neuer Wert} = \text{Alter Wert} - \text{Lernrate} \times \text{Partieller Gradient}$$

# MODELLPARAMETER VS. HYPERPARAMETER

## *Modellparameter*

- **Teile des Modells, die sich während des Trainingsprozesses automatisch anpassen, wie Gewichte und Biases**

## *Hyperparameter*

- **Einstellungen, die du vor dem Training festlegst, wie etwa die Lernrate oder die Anzahl der Modell-Parameter.**

# FEATURES AND LABELS

**Machine Learning**

**Feature**

**Label**

**Statistik**

**Unabhängige Variable**

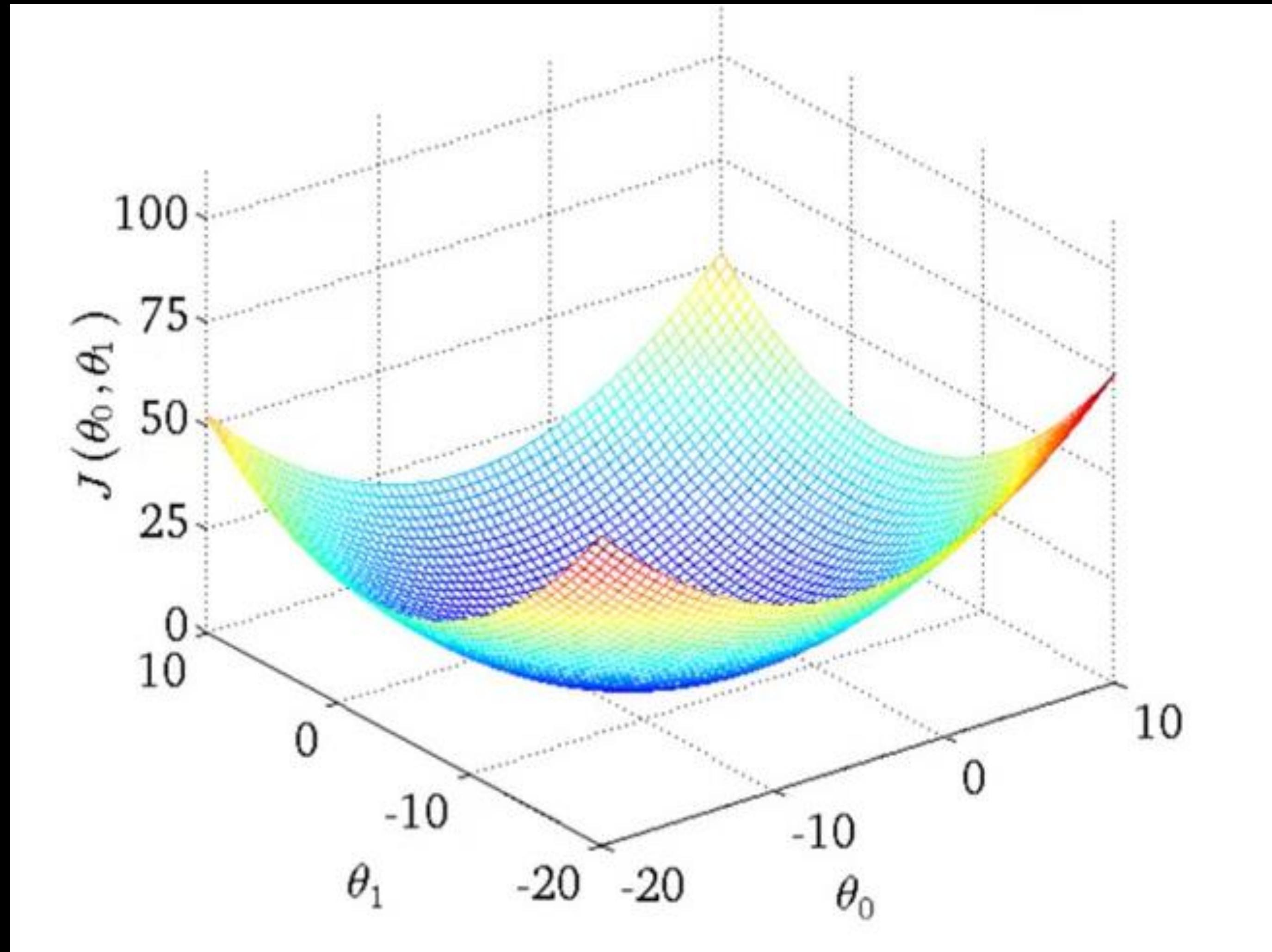
**Abhängige Variable**

**Beschreibung**

**Werte, die zur Vorhersage  
genutzt werden**

**Vorherzusagende Werte**

# MINIMIERUNG BEI LINEAREN MODELLEN



- Für lineare Modelle ist die Kostenfunktion konvex und besitzt keine lokalen Minima.
- Hier werden häufig auch andere statistische Verfahren als Gradient Descent genutzt.  
*(Insbesondere wenn das Modell nur wenige Variablen umfasst.)*

# BEISPIEL EINES LINEAREN MODELLS

```
library(readr)  
house_pricing <- read_csv("https://raw.githubusercontent.com/  
opencampus-sh/einfuehrung-in-data-  
science-und-ml/main/house_pricing_data/  
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<sup>2</sup>

- **Kennzahl zur Beurteilung der Güte einer Regression**
- **Wert zwischen 0 und 1, der dem Anteil der erklärten Variation entspricht (1 entspricht 100%):**
$$R^2 = \frac{\text{Erklärte Varianz}}{\text{Gesamtvarianz}}$$
- **Das adjustierte R<sup>2</sup> bestraft das Hinzufügen zusätzlicher Variablen/Parameter**

# BEISPIEL DATENSATZTEILUNG (KEINE ZEITREIHE)

```
12
13 * ##### Teilen des Datensatzes in Trainings-, Validierungs- und Testdatensatz
14 *   ``{r}
15 # Load the dplyr library
16 library(dplyr)
17
18 # Set a random seed for reproducibility
19 set.seed(42)
20 # Shuffle the data
21 data_shuffled <- data %>% sample_frac(1)
22
23 # Calculate the number of rows for each dataset
24 n_total <- nrow(data)
25 n_train <- floor(0.7 * n_total)
26 n_validation <- floor(0.20 * n_total)
27
28 # Split the data into training, validation, and test datasets
29 train_data <- data_shuffled %>% slice(1:n_train)
30 validation_data <- data_shuffled %>% slice((n_train + 1):(n_train + n_validation))
31 test_data <- data_shuffled %>% slice((n_train + n_validation + 1):n_total)
32
33 # Check the dimensions of the datasets
34 cat("Training dataset dimensions:", dim(train_data), "\n")
35 cat("Validation dataset dimensions:", dim(validation_data), "\n")
36 cat("Test dataset dimensions:", dim(test_data), "\n")
37
38 * ...
```

# BEISPIEL LINEARE MODELLIERUNG UND VORHERSAGE

```
42 ## Beispiel einer einfachen linearen Regression
43 ### {r}
44 mod <- lm(price ~ sqft_lot15 + as.factor(condition), train_data)
45 summary(mod)
46
47 ###
48
49 ## Nutzung des resultierenden Modells für eine Vorhersage
50 ### {r}
51 # Make predictions using the test data
52 predicted_values <- predict(mod, newdata = validation_data)
53
54 # Compare the predicted values with the actual values
55 comparison <- data.frame(Actual = test_data$price, Predicted = predicted_values)
56
57 # Calculate the mean squared error (RMSE)
58 rmse <- sqrt(mean((comparison$Actual - comparison$Predicted)^2))
59
60 # Display the comparison and RMSE
61 head(comparison)
62 cat("Root Mean Squared Error (RMSE):", rmse, "\n")
63
64 ###
```

# BREAKOUT

- Definiert eine lineare Modellgleichung und führt mit Eurem Datensatz eine lineare Regression durch.
- Versucht das adjustierte  $R^2$  des linearen Modells zu maximieren.

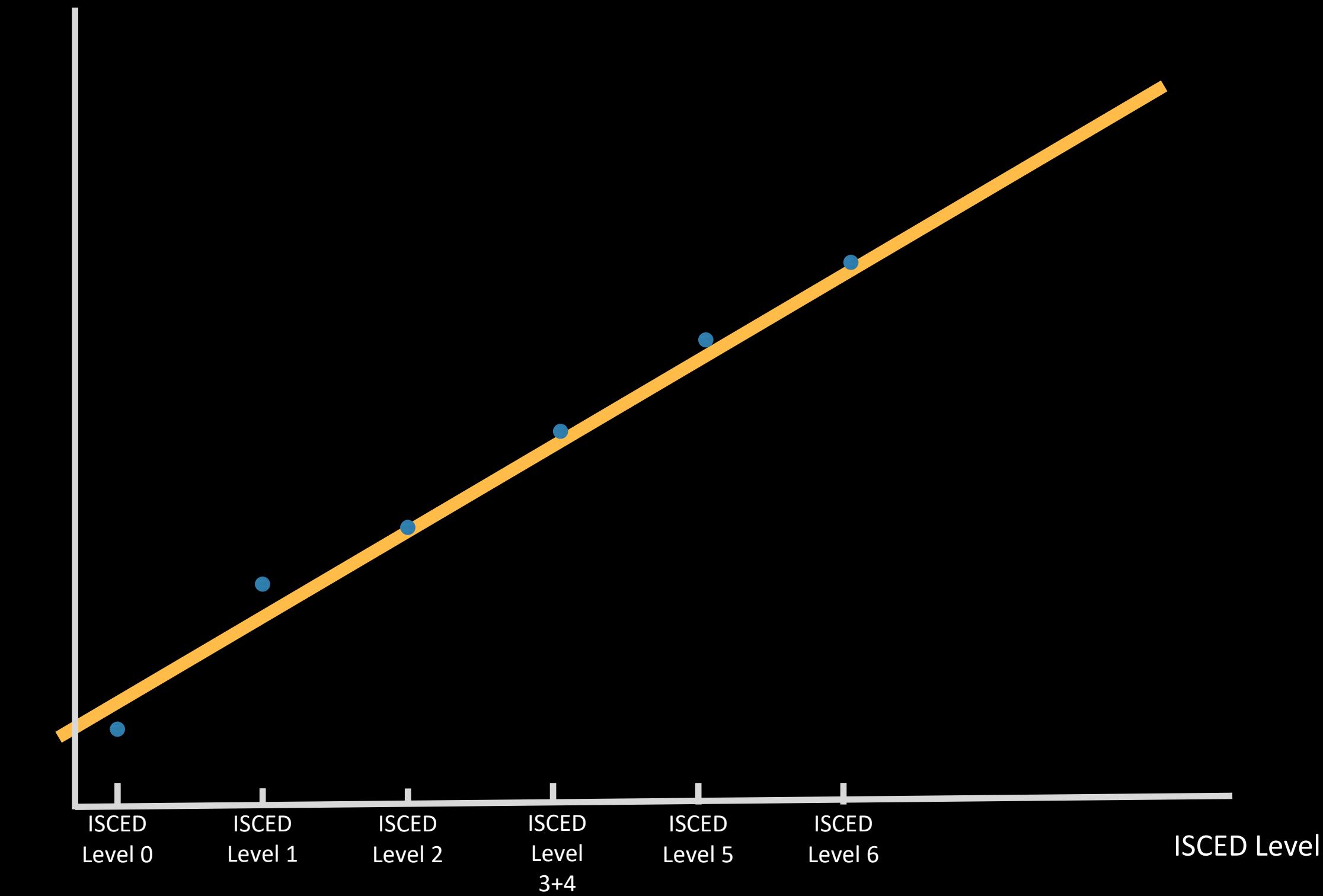
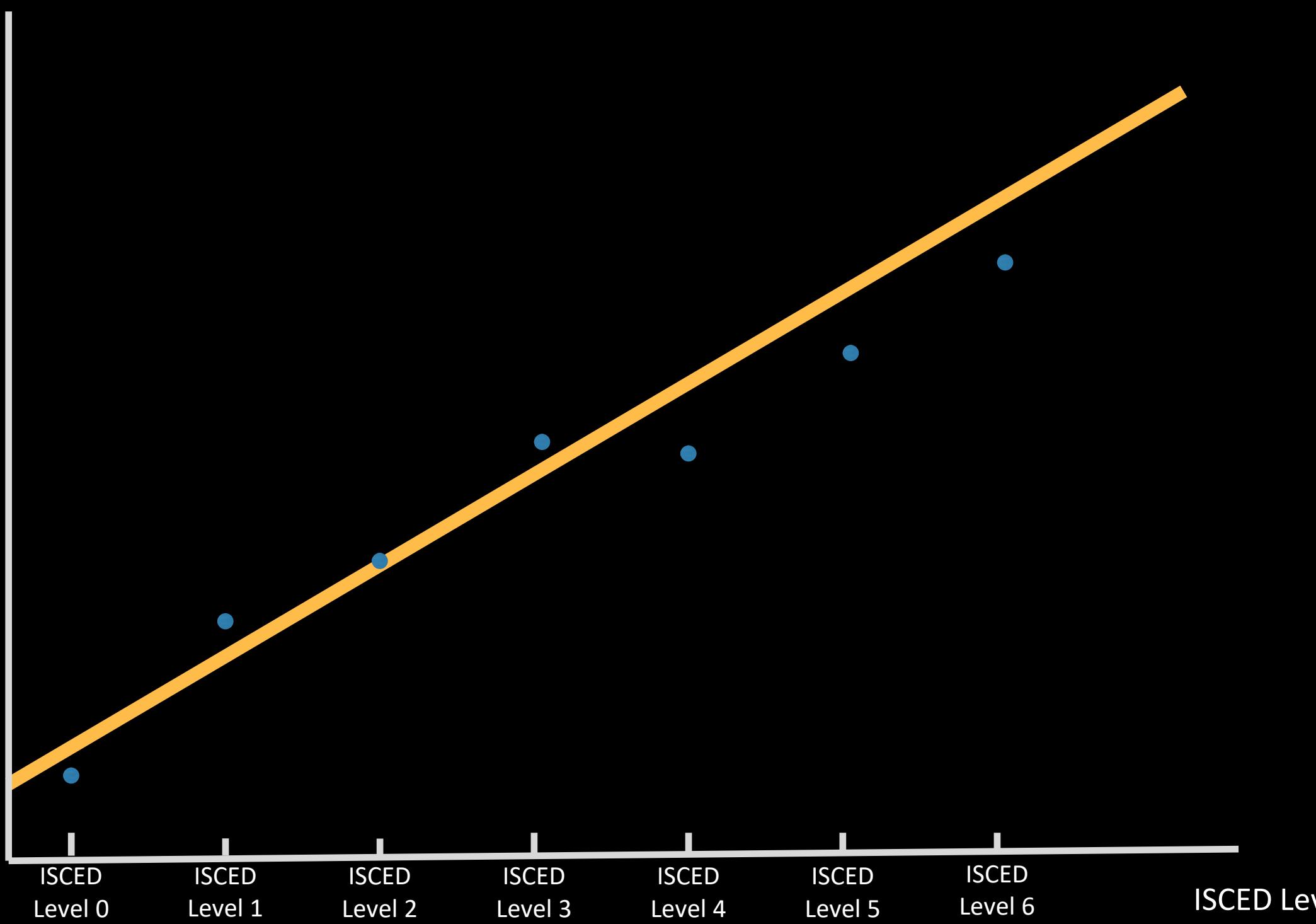
# BASELINE MODELL

- **Sollte einfach zu implementieren sein, mit einer vernünftigen Chance, anständige Ergebnisse zu liefern und einer sehr geringen Wahrscheinlichkeit des Overfitting.**
- **Sollte interpretierbar sein, um das Verständnis für die Daten zu verbessern und ein besseres „Feature Engineering“ zu ermöglichen.**

Ameisen, E. (2018, March 6). *Always start with a stupid model, no exceptions*. Medium.

<https://blog.insightdatascience.com/always-start-with-a-stupid-model-no-exceptions-3a22314b9aaa>

# RELEVANZ VON KATEGORISIERUNGEN



# AUFGABEN

- **Datensatz weiter um zusätzliche Variablen ergänzen, die für die Schätzung des Umsatzes relevant sein könnten.**
- **Euren Datensatz teilen in einen Trainingsdatensatz vom 01.07.2013 bis 31.07.2017 und einen Validierungsdatensatz vom 01.08.2017 bis 31.07.2018.**
- **Eine lineare Modellgleichung aufstellen, die das adjustierte R<sup>2</sup> für Euren Trainingsdatensatz maximiert.**
- **Im Verzeichnis „Baseline Model“ Eures Team Repositories die Berechnung für die lineare Regression dokumentieren.**
- **Zum Thema Overfitting [dieses Video \(9 Minuten\)](#) anschauen.**
- **Einen Account bei [Kaggle](#) erstellen.**

# Level up with the largest AI & ML community

Join over 15M+ machine learners to share, stress test, and stay up-to-date on all the latest ML techniques and technologies. Discover a huge repository of community-published models, data & code for your next project.

 [Register with Google](#)[Register with Email](#)

## Who's on Kaggle?

### Learners

Dive into Kaggle courses, competitions & forums.



### Developers

Leverage Kaggle's models, notebooks & datasets.



### Researchers

Advance ML with our pre-trained model hub & competitions.



≡ kaggle

+ Create

Home

Competitions

Datasets

Models

Code

Discussions

Learn

More

# Competitions

Grow your data science skills by competing in our exciting competitions. Find help in the [documentation](#) or learn about [Community Competitions](#).

Host a Competition

Search competitions

Filters

All Competitions

Everything, past & present

Featured

Premier challenges with prizes

Getting Started

Approachable ML fundamentals

Research

Scientific and scholarly challenges

Community

Created by fellow Kagglers

Playgr

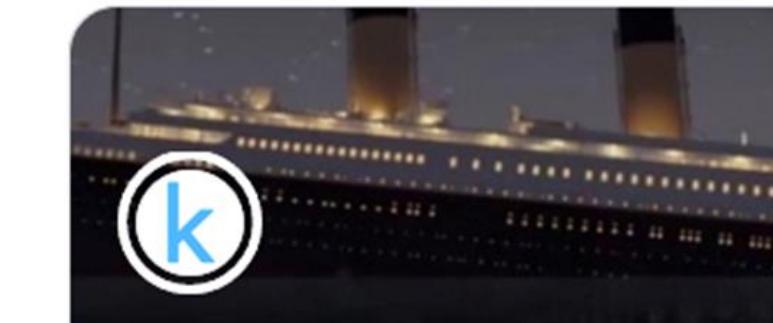
Fun practice problems

Get Started

See all

## New to Kaggle?

These competitions are perfect for newcomers.



Titanic - Machine Learning from Disaster

Start here! Predict survival on the Ti...  
Getting Started  
15573 Teams

Knowledge

Ongoing



House Prices - Advanced Regression...

Predict sales prices and practice fea...  
Getting Started  
4911 Teams

Knowledge

Ongoing



Spaceship Titanic

Predict which passengers are transp...  
Getting Started  
2555 Teams

Knowledge

Ongoing

View Active Events



Search competitions

Filters

+ Create

Home

Competitions

Datasets

Models

Code

Discussions

Learn

More



### LLM - Detect AI Generated Text

⋮

Identify which essay was written by ...

Featured · Code Competition

1740 Teams

\$110,000

2 months to go



### Open Problems - Single-Cell...

⋮

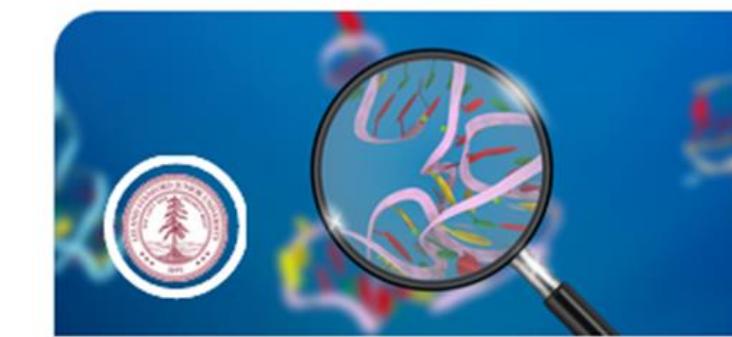
Predict how small molecules change...

Featured

1120 Teams

\$100,000

3 days to go



### Stanford Ribonanza RNA Folding

⋮

Create a model that predicts the str...

Research

676 Teams

\$100,000

10 days to go



### Optiver - Trading at the Close

⋮

Predict US stocks closing movements

Featured · Code Competition

3524 Teams

\$100,000

23 days to go



### NFL Big Data Bowl 2024

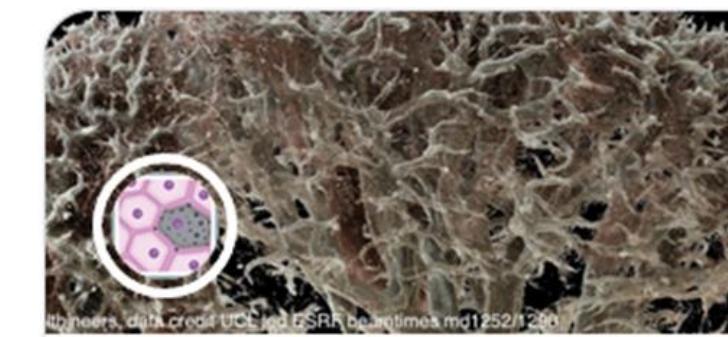
⋮

Help evaluate tackling tactics and st...

Analytics

\$100,000

a month to go



### SenNet + HOA - Hacking the Human...

⋮

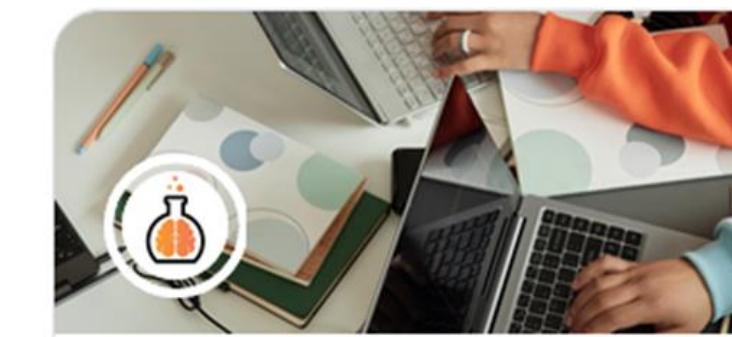
Segment vasculature in 3D scans of ...

Research · Code Competition

149 Teams

\$80,000

2 months to go



### Linking Writing Processes to Writing...

⋮

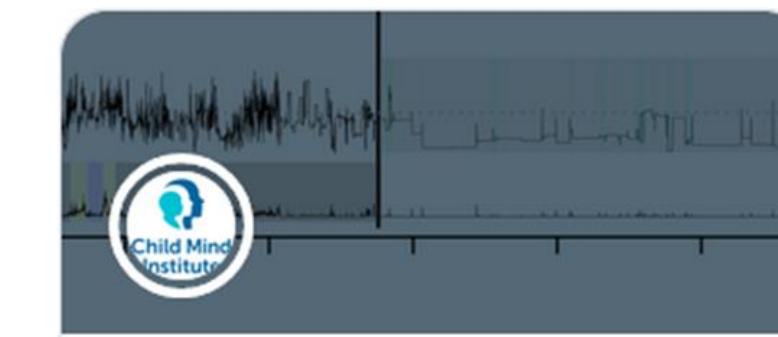
Use typing behavior to predict essa...

Featured · Code Competition

1139 Teams

\$55,000

a month to go



### Child Mind Institute - Detect Sleep States

⋮

Detect sleep onset and wake from w...

Featured · Code Competition

1762 Teams

\$50,000

8 days to go

View Active Events

[Create](#)[Home](#)[Competitions](#)[Datasets](#)[Models](#)[Code](#)[Discussions](#)[Learn](#)[More](#)

Research Prediction Competition

## Stanford Ribonanza RNA Folding

Create a model that predicts the structures of any RNA molecule



Stanford University · 676 teams · 10 days to go (3 days to go until merger deadline)

\$100,000  
Prize Money[Overview](#) [Data](#) [Code](#) [Models](#) [Discussion](#) [Leaderboard](#) [Rules](#)[Join Competition](#)

...

## Leaderboard

[Raw Data](#)[Refresh](#)

Search leaderboard

[Public](#)[Private](#)

This leaderboard is calculated with approximately 20% of the test data. The final results will be based on the other 80%, so the final standings may be different.

[Prize Contenders](#)

#	Team	Members	Score	Entries	Last	Join
1	HandFold		0.13733	85	40m	<a href="#">Join</a>
2	DI		0.13773	108	3h	<a href="#">Join</a>

[View Active Events](#)