

# DATA SCIENCE

KNN - LERNEN MIT ÄHNLICHKEITEN

PROF. DR. CHRISTIAN BOCKERMANN

HOCHSCHULE Bochum

WINTERSEMESTER 2025/2026

- 1 Lernen nach Ähnlichkeit
- 2 Distanz-Maße und nächste Nachbarn
- 3 Vergleich: Entscheidungsbäume und  $k$ -NN
- 4 Distanzen und Metriken
- 5 Overfitting: Wie genau müssen wir sein?

## Menschliches Lernen nutzt Ähnlichkeiten aus

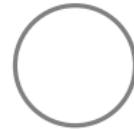
Zum Beispiel über Formen:



Quadrat



Rechteck



Kreis

## Menschliches Lernen nutzt Ähnlichkeiten aus

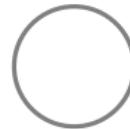
Zum Beispiel über Formen:



Quadrat



Rechteck



Kreis

Oder andere Eigenschaften (Größe, Gewicht):



Fussball



Handball



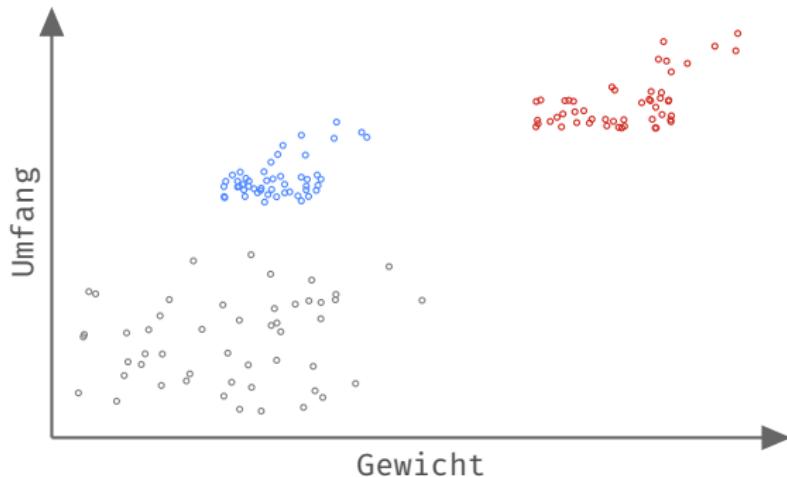
Basketball

## Beispiel: Klassifikation von Bällen

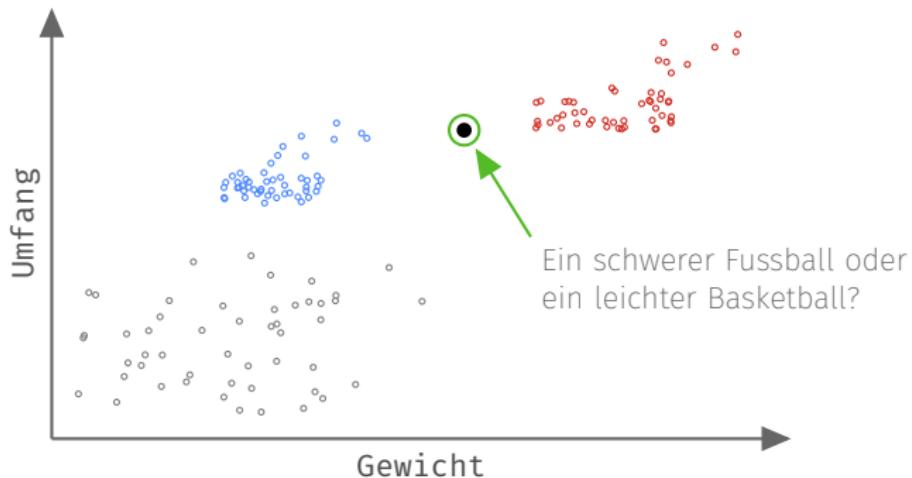
Wir wollen Bälle ihrer Sportart zuordnen ([Klassifikationsaufgabe](#))

Umfang (cm)	Gewicht (g)	Sportart
70.29	444.30	Fussball
77.73	647.53	Basketball
53.34	427.07	Handball
57.09	406.12	Handball
68.28	440.96	Fussball
80.38	648.94	Basketball

## Beispiel: Maße unterschiedlicher Bälle

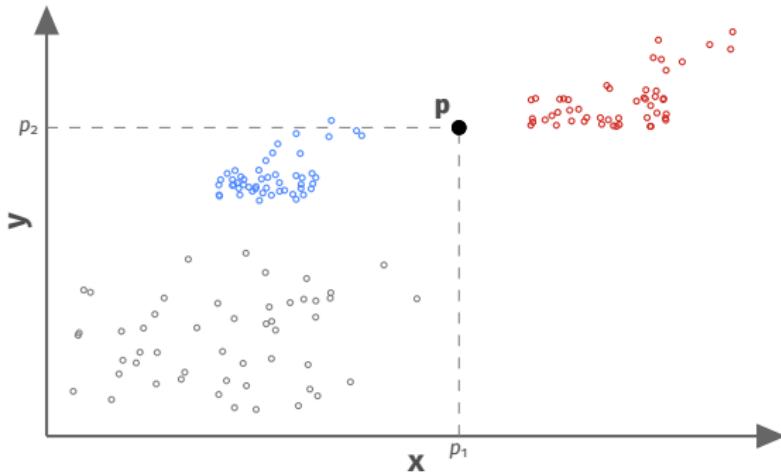


## Beispiel: Maße unterschiedlicher Bälle



Verschiedene Bälle nachgemessen: Fussball, Handball und Basketball

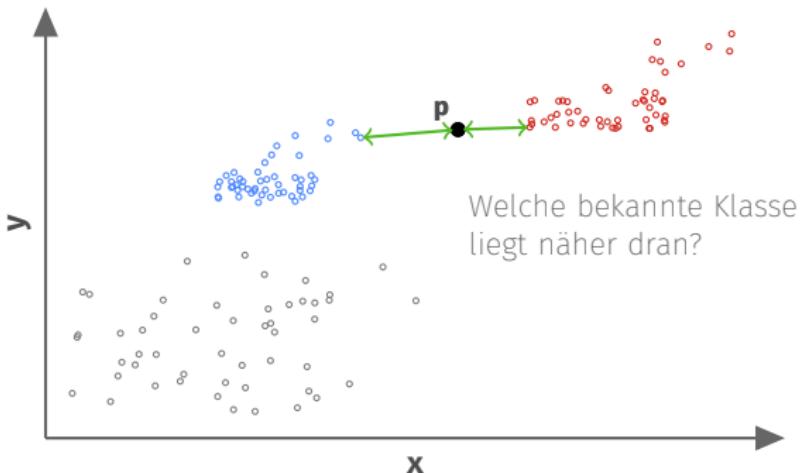
Betrachte 2-dimensionalen Raum:  $\mathbb{R}^2$



2-dimensionaler Raum: Jeder Punkt  $\mathbf{p}$  besteht aus 2 Koordinaten:

$$\mathbf{p} = (p_1, p_2)$$

Betrachte 2-dimensionalen Raum:  $\mathbb{R}^2$



Idee: Wir nutzen den Abstand als Ähnlichkeit und sagen die Klasse vorher, die am nächsten ist!

## Distanzmaße

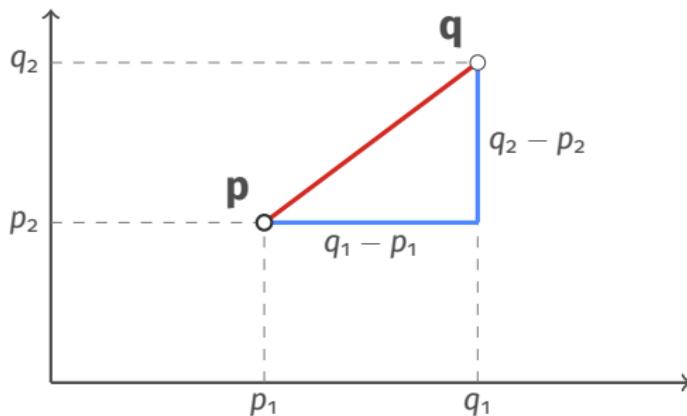
Euklidische Distanz zwischen Punkten  $\mathbf{p}$  und  $\mathbf{q}$ :

$$dist(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^d (p_i - q_i)^2}$$

- Gilt für  $d$  dimensionalen Raum
- Im Bälle-Beispiel ist der Raum 2-dimensional, d.h.  $d = 2$
- Euklidische Distanz entspricht der  $L_2$  Norm eines Vektorraumes

## Euklidische Distanz

Im 2-dimensionalen Raum ergibt sich die euklidische Distanz aus dem Satz des Pythagoras:



$$dist(\mathbf{p}, \mathbf{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$$

## Lernen mit Ähnlichkeiten

Wie bauen wir nun ein Modell auf Ähnlichkeiten auf?

1. Wir brauchen ein Ähnlichkeitsmaß (z.B. euklidische Distanz)
2. Wir merken uns den gesamten Trainingsdatensatz
3. Zur Vorhersage eines Beispiels vergleichen wir es mit allen Trainingsbeispielen und sagen die Klasse des ähnlichsten Beispiels vorher!

## Lernen mit Ähnlichkeiten: Training des Modells

$X_{train}$		$y_{train}$
Umfang	Gewicht	Art
75,93	638,26	BB
70,39	409,84	FB
52,95	351,01	HB

**Modell m**

## Lernen mit Ähnlichkeiten: Training des Modells

$X_{train}$		$y_{train}$
Umfang	Gewicht	Art
75,93	638,26	BB
70,39	409,84	FB
52,95	351,01	HB

**Modell m**

Das Modell merkt sich den kompletten Trainingsdatensatz!

```
m = datascience.AuswendigLernen()  
m.fit(X_train, y_train)
```

## Lernen mit Ähnlichkeiten: Vorhersage mit Modell m

```
y_hat = m.predict(X_test)
```

Modell m

$X_{train}$	$y_{train}$
Umfang	Gewicht
75,93	638,26
70,39	409,84
52,95	351,01

	Art
BB	
FB	
HB	

Neue Daten

X\_test

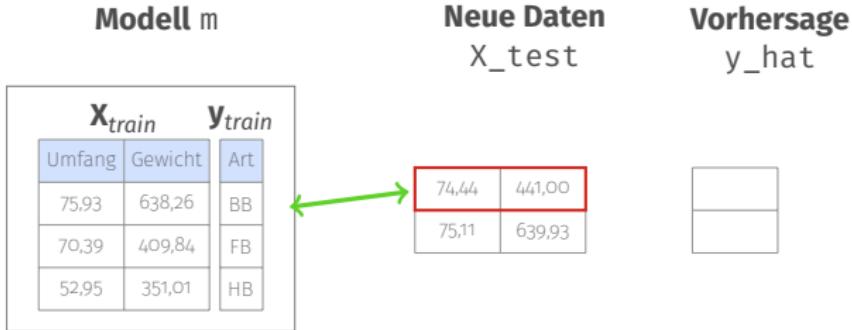
74,44	441,00
75,11	639,93

Vorhersage

y\_hat


## Lernen mit Ähnlichkeiten: Vorhersage mit Modell m

```
y_hat = m.predict(X_test)
```



Distanz:

$$\sqrt{(74,44 - 75,93)^2 + (441,00 - 638,26)^2} \approx 198,931$$

## Lernen mit Ähnlichkeiten: Vorhersage mit Modell m

```
y_hat = m.predict(X_test)
```

Modell m		Neue Daten X_test		Vorhersage y_hat
X <sub>train</sub>	y <sub>train</sub>			
Umfang	Gewicht	Art		
75,93	638,26	BB	74,44	441,00
70,39	409,84	FB	75,11	639,93
52,95	351,01	HB		

Distanz:

$$\sqrt{(74,44 - 70,39)^2 + (441,00 - 409,84)^2} \approx 31.422$$

## Lernen mit Ähnlichkeiten: Vorhersage mit Modell m

```
y_hat = m.predict(X_test)
```

Modell m

X <sub>train</sub>	y <sub>train</sub>
Umfang	Gewicht
75,93	638,26
70,39	409,84
52,95	351,01

Art
BB
FB
HB

Neue Daten  
X<sub>test</sub>

74,44	441,00
75,11	639,93

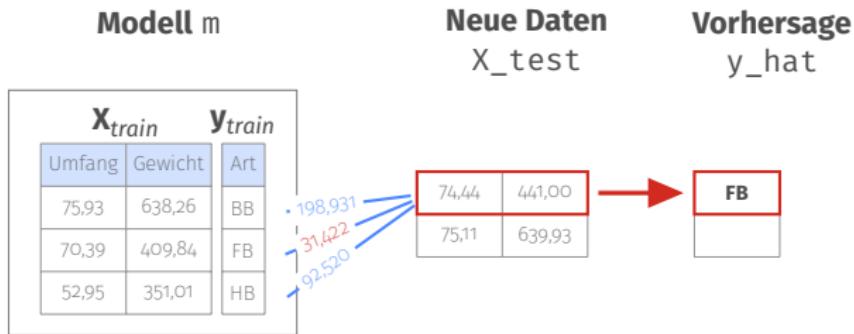
Vorhersage  
y\_hat


Distanz:

$$\sqrt{(74,44 - 52,95)^2 + (441,00 - 351,01)^2} \approx 92.520$$

## Lernen mit Ähnlichkeiten: Vorhersage mit Modell m

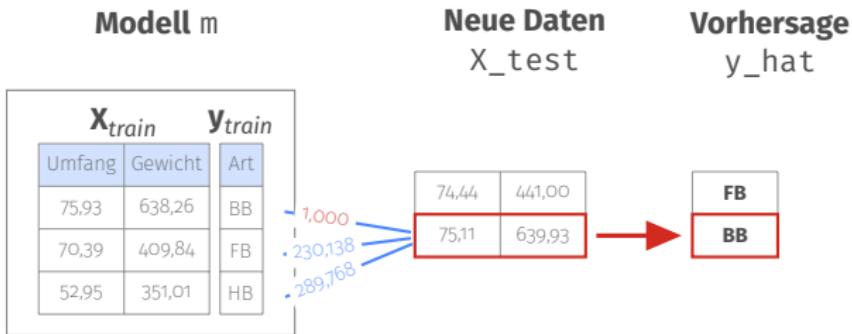
```
y_hat = m.predict(X_test)
```



Die Klasse **FB** ist  
am nächsten dran!

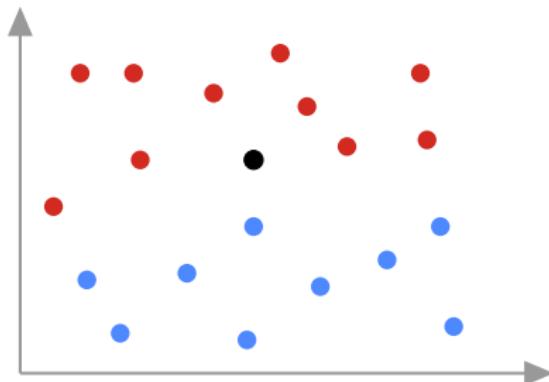
## Lernen mit Ähnlichkeiten: Vorhersage mit Modell m

```
y_hat = m.predict(X_test)
```

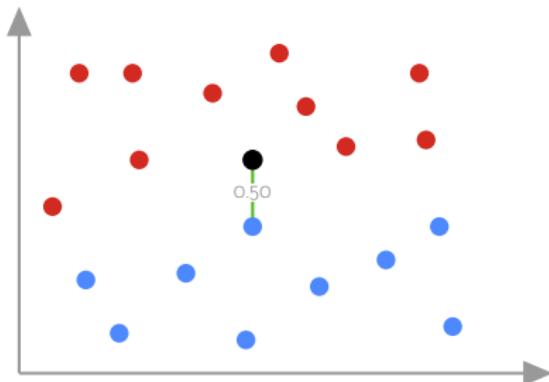


Die Klasse **BB** ist  
am nächsten dran!

Frage: Ist das nächstgelegene Beispiel eine gute Vorhersage?



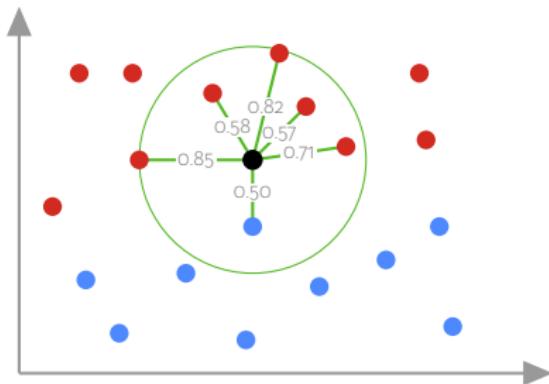
Frage: Ist das nächstgelegene Beispiel eine gute Vorhersage?



Welcher Klasse würden Sie den scharzen Punkt zuordnen?

- Der nächstgelegene Punkt ist blau

Frage: Ist das nächstgelegene Beispiel eine gute Vorhersage?



Welcher Klasse würden Sie den scharzen Punkt zuordnen?

- Der nächstgelegene Punkt ist blau
- Von den 5 nächstengelegenen Punkten sind 4 rot und 1 blau

## Verfahren: $k$ -Nearest Neighbors ( $k$ -NN)

**Training:** Speichern der Trainingsdaten  $\mathbf{X}_{train}, \mathbf{y}_{train}$

**Vorhersage:** Beispiel  $\mathbf{x}$

- Bestimme die  $k$  nächsten Nachbarn für  $\mathbf{x}$  in  $\mathbf{X}_{train}, \mathbf{y}_{train}$
- Vorhersage für  $\mathbf{x}$  ist häufigste Klasse der Nachbarn von  $\mathbf{x}$

## Verfahren: **$k$ -Nearest Neighbors ( $k$ -NN)**

**Training:** Speichern der Trainingsdaten  $\mathbf{X}_{train}, \mathbf{y}_{train}$

**Vorhersage:** Beispiel  $\mathbf{x}$

- Bestimme die  $k$  nächsten Nachbarn für  $\mathbf{x}$  in  $\mathbf{X}_{train}, \mathbf{y}_{train}$
- Vorhersage für  $\mathbf{x}$  ist häufigste Klasse der Nachbarn von  $\mathbf{x}$

## $k$ -NN in Python (SciKit-Learn):

```
from sklearn.neighbors import KNeighborsClassifier  
  
m = KNeighborsClassifier(5) # hier: 'k' = 5
```

## Was haben Sie jetzt gelernt?

- Daten als 2-dimensionaler Raum, euklidische Distanz
- Neues Lernverfahren ( $k$ NN) für Klassifikation
- Direkt in Python/Notebook nutzen:

```
m = KNeighborsClassifier(5)  
m.fit(X_train, y_train)
```

## Was haben Sie jetzt gelernt?

- Daten als 2-dimensionaler Raum, euklidische Distanz
- Neues Lernverfahren ( $k$ NN) für Klassifikation
- Direkt in Python/Notebook nutzen:

```
m = KNeighborsClassifier(5)  
m.fit(X_train, y_train)
```

## Worüber sollten wir uns weiter Gedanken machen?

- Welches  $k$  nehmen wir für unser  $k$ -NN Modell?
- Was müssen wir bei Distanzen beachten? Was ist mit  $d > 2$ ?
- Wie unterscheidet sich  $k$ -NN von Entscheidungsbäumen?

## Welches $k$ nehmen wir?

Ein Ansatz: Auswahl nach Trainingsfehler?

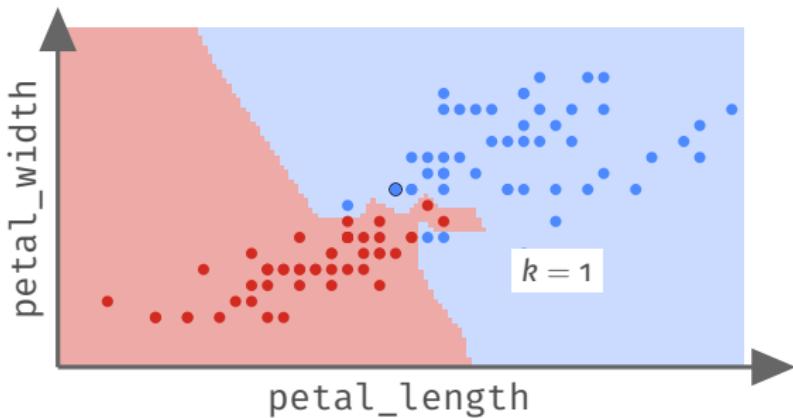
### Überlegen Sie selbst:

- Was passiert bei  $k = 1$  im Training/der Vorhersage?
- Welchen Trainingsfehler bekommen wir für  $k = 1$ ?



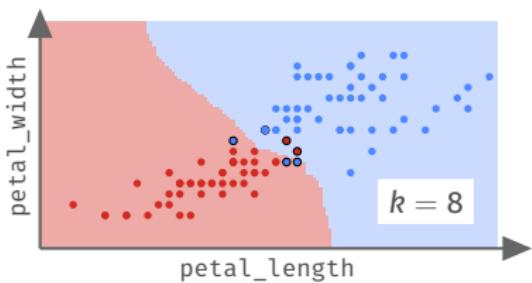
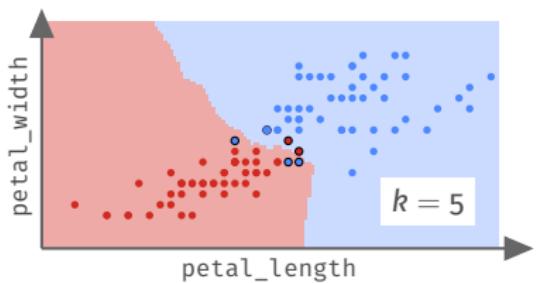
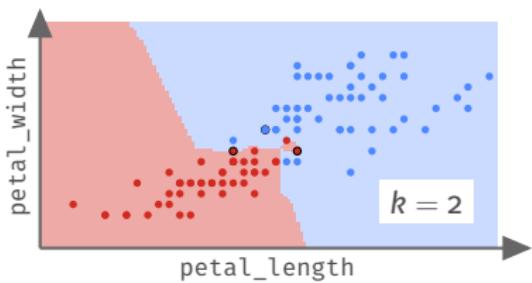
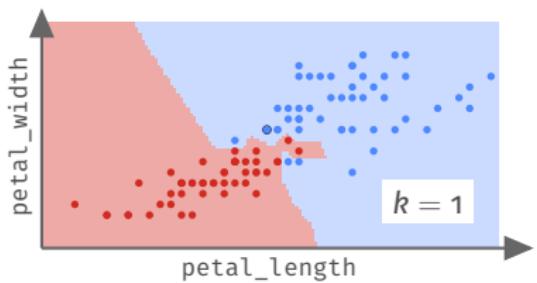
Probieren Sie es im Notebook aus!

Notebook: [Kurse/DataScience1/V6-knn-Welches-k.ipynb](#)

Iris Daten (2 Klassen),  $k$ -NN mit  $k = 1$ 

- Plot zeigt den Iris Datensatz ohne die Klasse *setosa*
- Hintergrund zeigt alle Punkte (Regionen) die der Klasse *versicolor*, bzw. *virginica* zugeordnet werden würden

## Iris Daten (2 Klassen) mit unterschiedlichen $k$ Werten



## Bedeutung von $k$

- Vorhersage ist quasi *Mehrheitsentscheid* (Majority Vote)
- Je größer  $k$ , desto kleiner der Einfluss *einzelner* Datenpunkte
- Größeres  $k$  führt zu weniger Inseln, geglätteten Entscheidungslinien

## Trade-Off: Generalisierbarkeit und Spezialisierung

- Wie geht  $k$ -NN für verschiedene  $k$  mit “Ausnahmen” um?

## Bedeutung von $k$

- Vorhersage ist quasi *Mehrheitsentscheid* (Majority Vote)
- Je größer  $k$ , desto kleiner der Einfluss *einzelner* Datenpunkte
- Größeres  $k$  führt zu weniger Inseln, geglätteten Entscheidungslinien

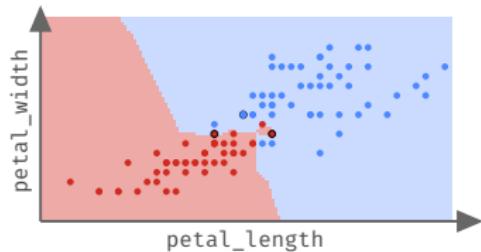
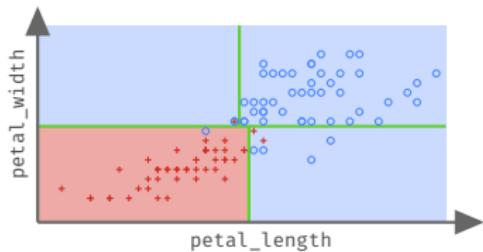
## Trade-Off: Generalisierbarkeit und Spezialisierung

- Wie geht  $k$ -NN für verschiedene  $k$  mit “Ausnahmen” um?

Schauen Sie sich vor dem Hintergrund dieser Punkte nochmal die Folien 16 an!

## Wo besteht der Unterschied zu Entscheidungsbäumen?

- Entscheidungslinien parallel zu den Achsen
- Schrittweise Aufteilung nach jeweils nur *einem* Attribut
- Nachbarschaftsansatz führt zu *Bereichen* für jede Klasse
- Entscheidung nach Distanz über alle Attribute (Mehrheit)



# Distanzen und Metriken

## Was ist eine Distanz? Was ist eine Metrik?

Sei  $V$  eine Menge.

Ein Distanzmaß ***dist*** ist eine Funktion, die zwei Punkten  $p, q \in V$  einen Abstand zuordnet

$$dist : V \times V \rightarrow \mathbb{R}$$

und dabei die folgenden Eigenschaften hat:

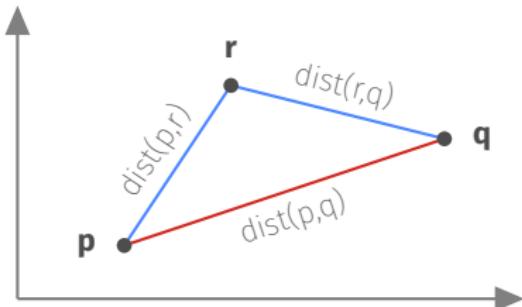
1.  $dist(p, q) = dist(q, p)$
2.  $dist(p, q) \geq 0$  und  $dist(p, q) = 0 \Leftrightarrow p = q$

## Was ist eine Distanz? Was ist eine Metrik?

Ein Distanzmaß ist eine **Metrik**, wenn zusätzlich die sogenannte Dreiecks-Ungleichung gilt:

$$3. \ dist(p, q) \leq dist(p, r) + dist(r, q)$$

**Anschaulich:** “Der direkte Weg ist immer der kürzeste!”



## Welche Distanzen/Metriken gibt es?

Manhattan Distanz  
(**1**-Norm)

$$dist_{L_1}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^d |x_i - y_i|$$

Euklidische Distanz  
(**2**-Norm)

$$dist_{L_2}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^d (x_i - y_i)^2}$$

Minkowski Distanz  
(**p**-Norm)

$$dist_{L_p}(\mathbf{x}, \mathbf{y}) = \left[ \sum_{i=1}^d |x_i - y_i|^p \right]^{1/p}$$

## Distanzen funktionieren nur auf metrischen, skalierten Variablen

Datensätze bisher:

- Iris-Daten, Attribute waren Maße in Zentimetern
- Ball-Daten, Attribute in Gramm und Zentimetern

**Frage:** Was bedeutet  $dist(p, q) = 10$  bei den Ball-Daten?

$$\sqrt{\underbrace{(p_1 - q_1)^2}_{\text{Umfang}} + \underbrace{(p_2 - q_2)^2}_{\text{Gewicht}}} = 10$$

0 cm	10 g	Gleicher Umfang, 10g schwerer
10 cm	0 g	10cm größer, gleiches Gewicht

## Distanzen funktionieren nur auf metrischen, skalierten Variablen

Datensätze bisher:

- Iris-Daten, Attribute waren Maße in Zentimetern
- Ball-Daten, Attribute in Gramm und Zentimetern

**Frage:** Was bedeutet  $dist(p, q) = 10$  bei den Ball-Daten?

$$\sqrt{\underbrace{(p_1 - q_1)^2}_{\text{Umfang}} + \underbrace{(p_2 - q_2)^2}_{\text{Gewicht}}} = 10$$

0 cm	10 g	Gleicher Umfang, 10g schwerer
10 cm	0 g	10cm größer, gleiches Gewicht

Wertebereich *Umfang*: 48,57 cm bis 83,97 cm

Wertebereich *Gewicht*: 315,64 g bis 686,33 g

**Frage:** Was bedeutet  $dist(p, q) = 10$  bei den Ball-Daten?

$$\sqrt{\underbrace{(p_1 - q_1)^2}_{Umfang} + \underbrace{(p_2 - q_2)^2}_{Gewicht}} = 10$$

0 cm	10 g	Gleicher Umfang, 10g schwerer
10 cm	0 g	10cm größer, gleiches Gewicht

**Frage:** Was bedeutet  $dist(p, q) = 10$  bei den Ball-Daten?

$$\sqrt{\underbrace{(p_1 - q_1)^2}_{Umfang} + \underbrace{(p_2 - q_2)^2}_{Gewicht}} = 10$$

0 cm	10 g	Gleicher Umfang, 10g schwerer
10 cm	0 g	10cm größer, gleiches Gewicht

Wertebereich *Umfang*: 48,57 cm bis 83,97 cm

Wertebereich *Gewicht*: **315,64** g bis **686,33** g

**Frage:** Was bedeutet  $dist(p, q) = 10$  bei den Ball-Daten?

$$\sqrt{\underbrace{(p_1 - q_1)^2}_{\text{Umfang}} + \underbrace{(p_2 - q_2)^2}_{\text{Gewicht}}} = 10$$

0 cm	10 g	Gleicher Umfang, 10g schwerer
10 cm	0 g	10cm größer, gleiches Gewicht

Wertebereich *Umfang*: 48,57 cm bis 83,97 cm

Wertebereich *Gewicht*: **315,64** g bis **686,33** g

**Die Metrik behandelt beide Variablen gleich.  
Das macht eventuell nicht immer so viel Sinn!**

## Bezug z.B. zur **Wirtschaftsstatistik**

Charakterisierung von Attributen/Merkmalen/Variablen durch

- Minimum, Maximum
- Mittelwert, Standardabweichung

## In welcher Relation steht $\text{dist}(p,q) = 10$ zu Umfang/Gewicht?



Probieren Sie es im Notebook aus!

Notebook: [Kurse/DataScience1/V6-Wertebereiche.ipynb](#)

## Idee: Normalisierung der Attribute/Variablen

- Anpassung der Werte auf gleichen Wertebereich
- z.B. Skalierung jeder Spalte auf [0,1]

### Min-Max-Normalisierung einer Variablen X

$$X' = \frac{X - \min(X)}{\max(X) - \min(X)}$$

```
zaehler = df['Umfang'] - min(df['Umfang'])
nenner = max(df['Umfang']) - min(df['Umfang'])

df['Umfang'] = zaehler / nenner
```

## Frage: Was ist mit der Verteilung der Attribute?

- Wertebereiche mit Min/Max auf [0,1] normalisiert
- Variablen haben aber ggf. unterschiedliche Mittelwerte/Std-Abweichung?

## Z-Normalisierung einer Variablen X

$$X'' = \frac{X - \mu(X)}{\sigma(X)}$$

ergibt eine Variable  $X''$  mit Mittelwert 0 und Standardabweichung etwa bei 1.

# Overfitting: Wie genau müssen wir sein?

## Diskussion: Wie genau müssen wir sein?



## Diskussion: Wie genau müssen wir sein?



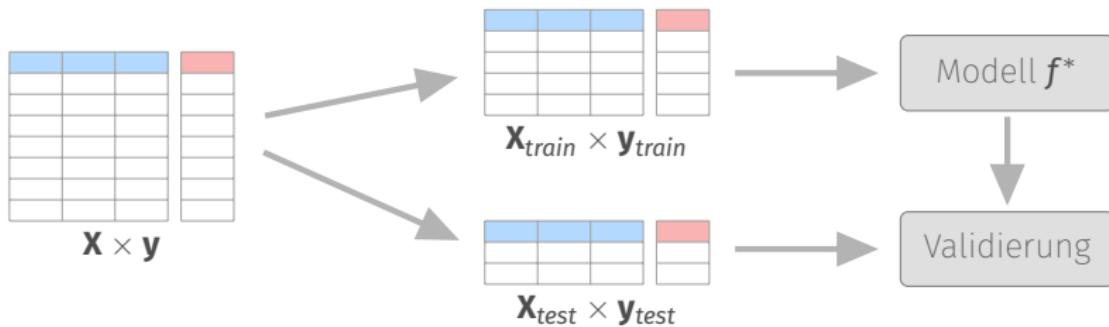
## Diskussion: Wie genau müssen wir sein?



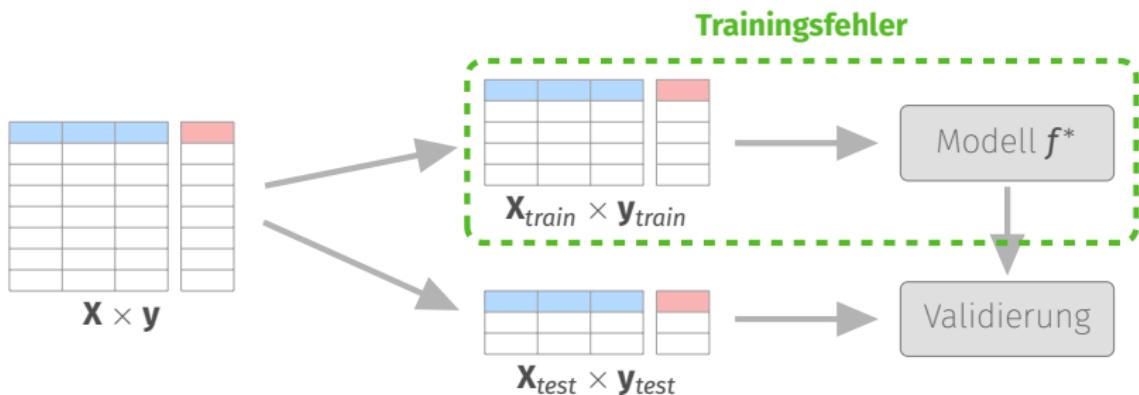
## Diskussion: Wie genau müssen wir sein?



## Diskussion: Wie genau müssen wir sein?

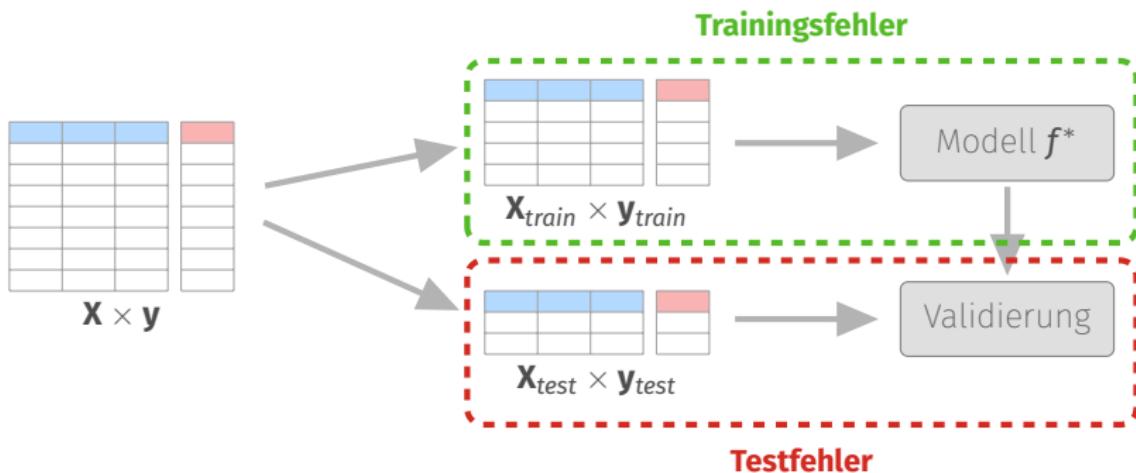


## Diskussion: Wie genau müssen wir sein?



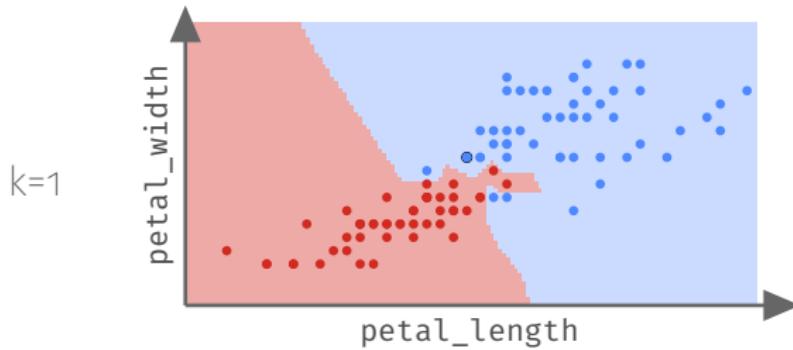
- Trainingsfehler: Modell an Trainingsdaten anpassen

## Diskussion: Wie genau müssen wir sein?

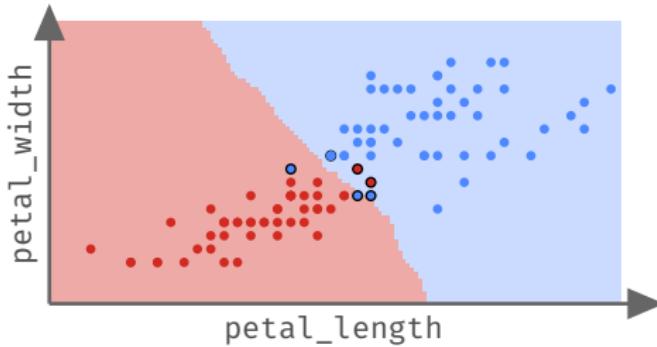


- Trainingsfehler: Modell an Trainingsdaten anpassen
- Testfehler: Modell auf unbekannten Daten  
(auch: Generalisierungsfehler)

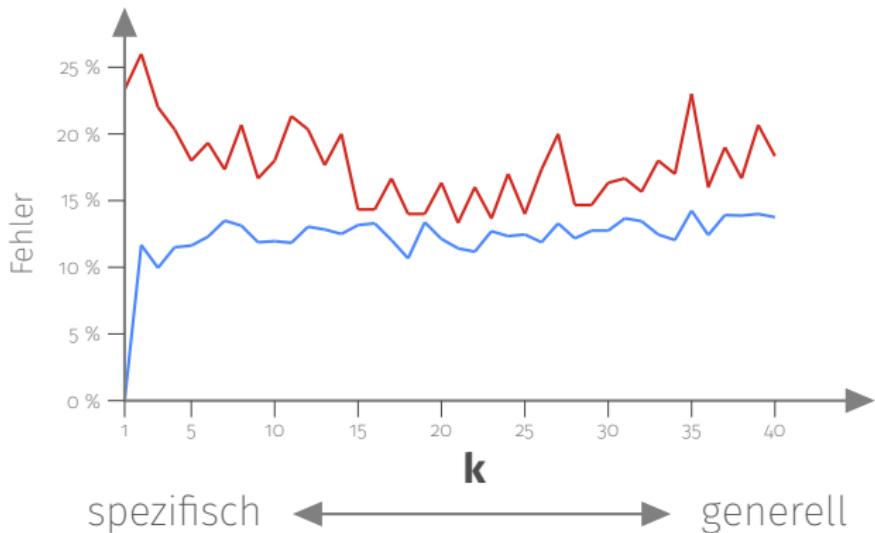
# DISKUSSION - MODELLANPASSUNG



$k=16$



## Training und Test-Fehler auf generiertem Datensatz (k-NN)



## Overfitting

“Das Modell passt nur zu den Trainingsdaten.”

## Overfitting

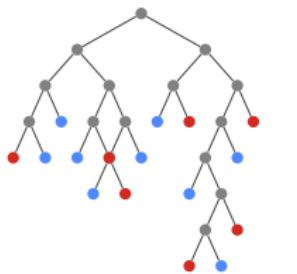
“Das Modell passt nur zu den Trainingsdaten.”

	Trainingsfehler <b>klein</b>	Trainingsfehler <b>groß</b>
Testfehler <b>klein</b>	Das sieht gut aus!	
Testfehler <b>groß</b>	<b>Overfitting!</b>	Das Modell lernt nicht!?

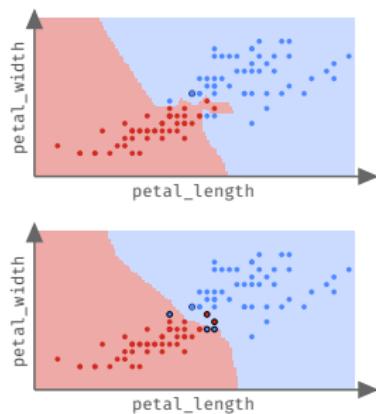
## Overfitting - zu spezifisches Modell

- Modell zu sehr an die Trainingsdaten angepasst
- Vorhersage auf unbekannten Daten schlechter
- Modellkomplexität begrenzen (generelleres Modell)

Tiefe bei Bäumen beschränken



k bei k-NN erhöhen



## A capella Song zum Thema Overfitting



<https://youtu.be/DQWI1kvmwRg>