

Statistik für Data Scientists

Vorlesung 2: Deskriptiven Statistik - Lage, Streuung und Verteilungsform

Prof. Dr. Siegfried Handschuh

Universität St. Gallen

Agenda

1. Lagekennzahlen
2. Streuungskennzahlen
3. Ausreisser erkennen und behandeln
4. Histogramm und **KDE** (Kernel Density Estimation)

Letze Woche

- Statistik hängt von Datentypen ab
- Bias = systematische Verzerrung erkennen
- Missing Values unterscheiden (MCAR, MAR, MNAR)
- Erste EDA (Exploratory Data Analysis) = Daten anschauen & einfache Plots

Kahoot!

1. Lagekennzahlen

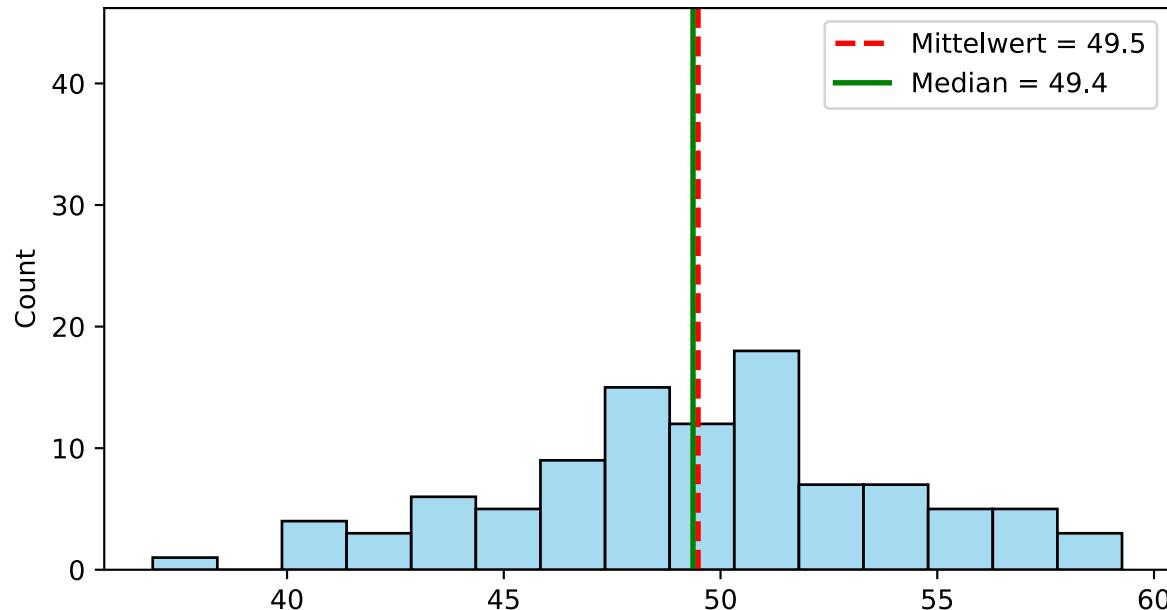
Warum Lagekennzahlen wichtig sind

Die Wahl der Lagekennzahl bestimmt das Zentrum.

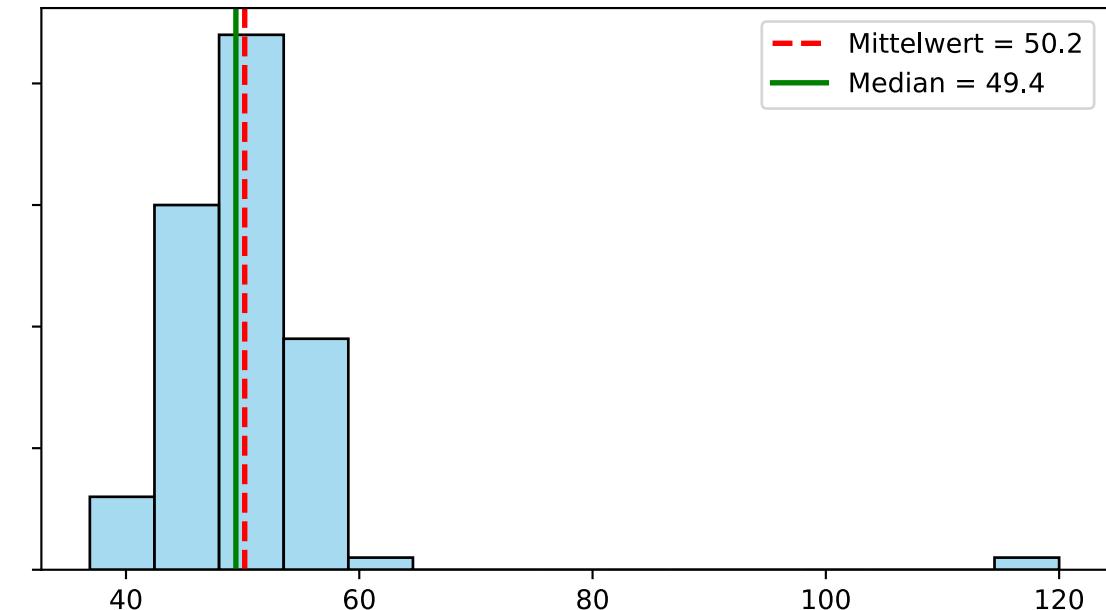
- Beispiel Einkommen: [30, 32, 35, 40, 45, 50, 500]
- Mittelwert =104.6 vs. Median =40
- Ausreißer kippen den Mittelwert
- Median oder getrimmter Mittelwert sind robuster
- **Mini-Check:** Welche Kennzahl würdest du für Einkommen einer Stadt berichten und warum?

Median bleibt stabil – Mittelwert verschiebt sich bei Ausreisern

Ohne Ausreisser



Mit Ausreisser

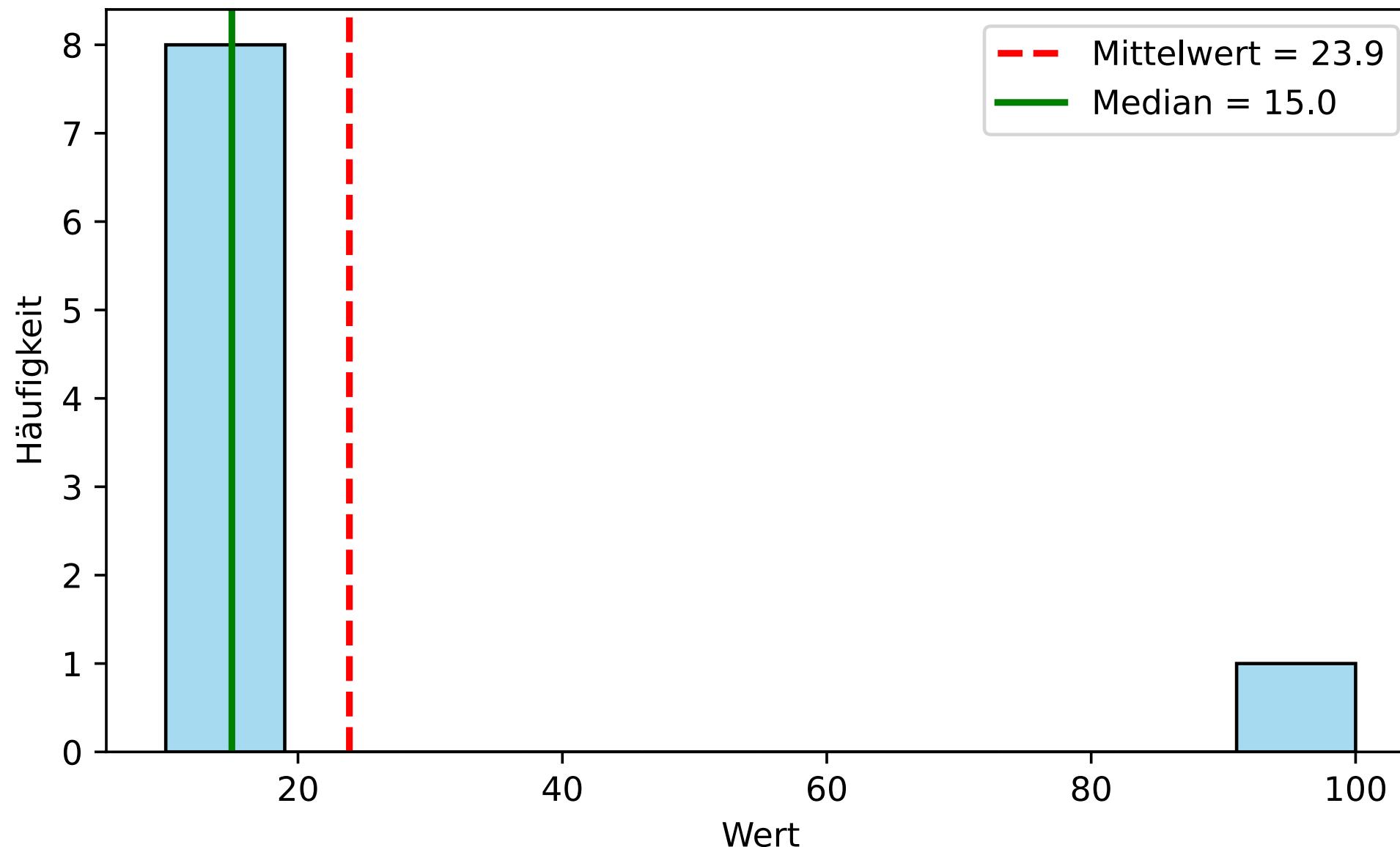


Mittelwert – Definition & Eigenschaften

Mittelwert ist linear und ausreisserempfindlich.

- $\bar{x} = \frac{1}{n} \sum x_i$
- Linearität: $E[aX + b] = aE[X] + b$
- Sensitiv für extreme Werte
- Python: `s.mean()`
- **Mini-Check:** Wann ist Sensitivität ein Vorteil?

Histogramm mit Ausreißer, Mittelwert & Median

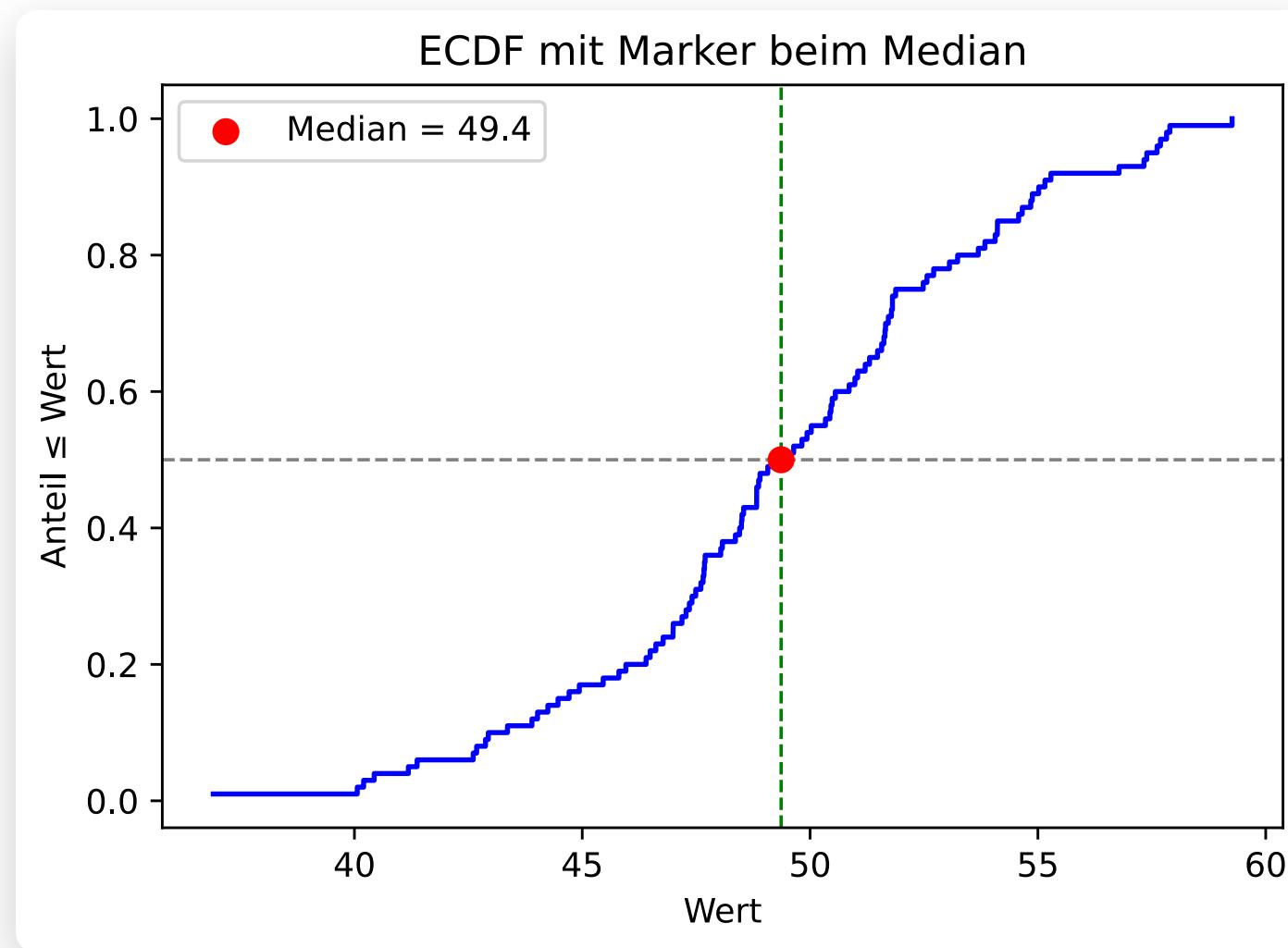


Median – robustes Zentrum

Median ist das 50%-Quantil und robust.

- $Q_{0.5}$ teilt Daten in zwei Hälften
- Minimiert Summe absoluter Abweichungen
- Stabil bei wenigen extremen Werten
- Python: `s.median()`
- **Mini-Check:** Warum ist Median bei Wartezeiten oft besser?

Empirische Kumulative Verteilungsfunktion



Getrimmter Mittelwert

*Trimmen dämpft Ausreisser, behält Mittelwertlogik.

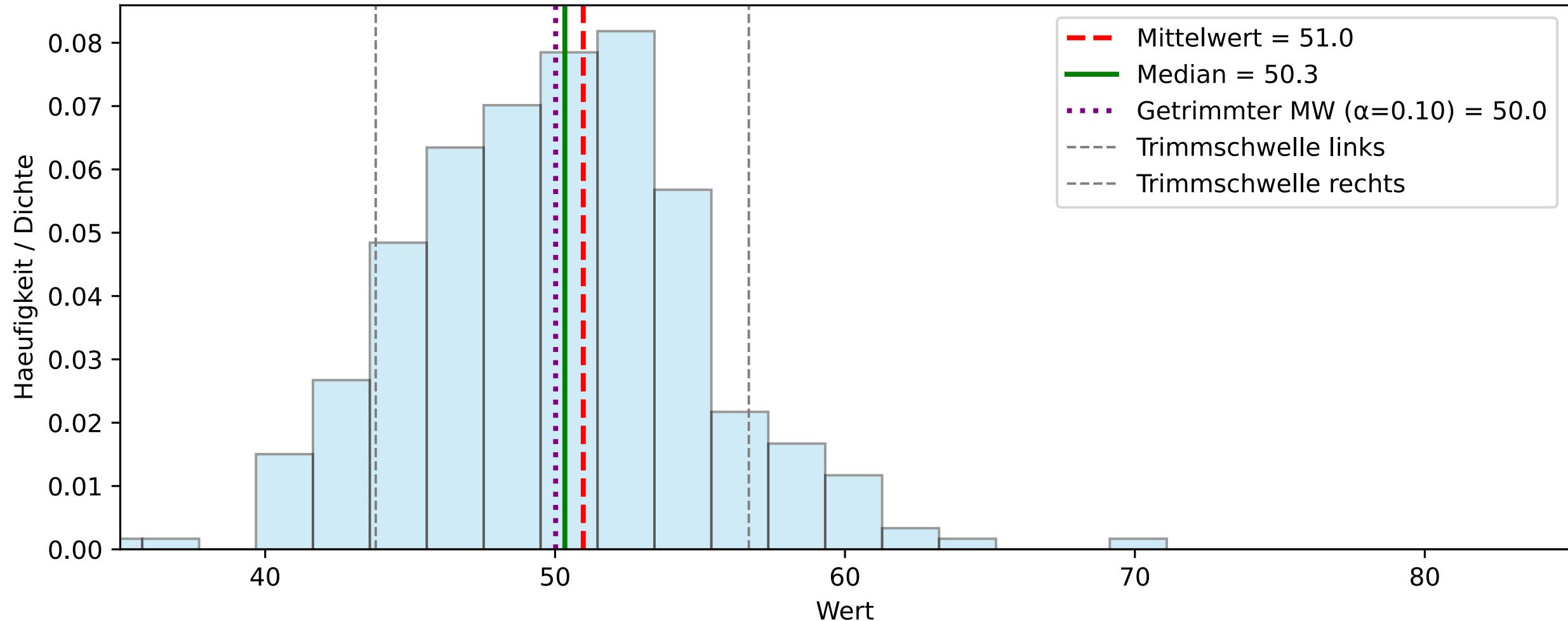
- Idee: je 10 Prozent an Rändern entfernen
- Reduziert Varianz bei Heavy Tails
- Abgrenzung zu Winsorisierung

Python:

```
from scipy.stats import trim_mean  
trim_mean(s, proportiontocut=0.1)
```

Mini-Check: Wann wären 20 Prozent Trimmen zu viel?

Getrimmter Mittelwert ist robuster gegen Ausreißer

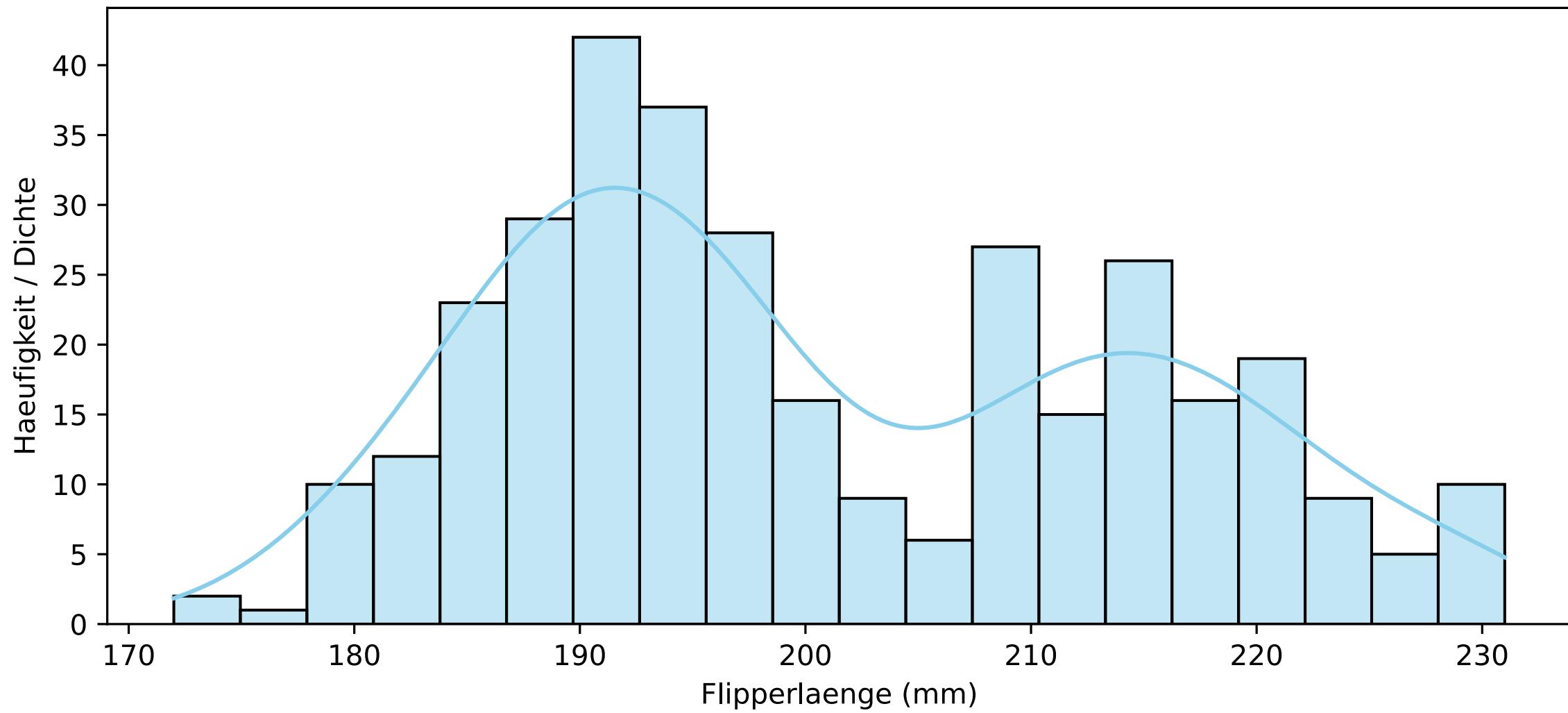


Modus & Multimodalität

Modus signalisiert Häufigkeit und Mischungen.

- Modus: häufigster Wert
- Multimodalität deutet Gruppen oder Mischungen an
- Bei stetigen Daten Modus binabhängig
- Python: `s.mode()`; Dichte anschauen
- **Mini-Check:** Woran erkennst du zweigipflige Verteilungen?

Penguins: Flipperlaenge zeigt Multimodalitaet



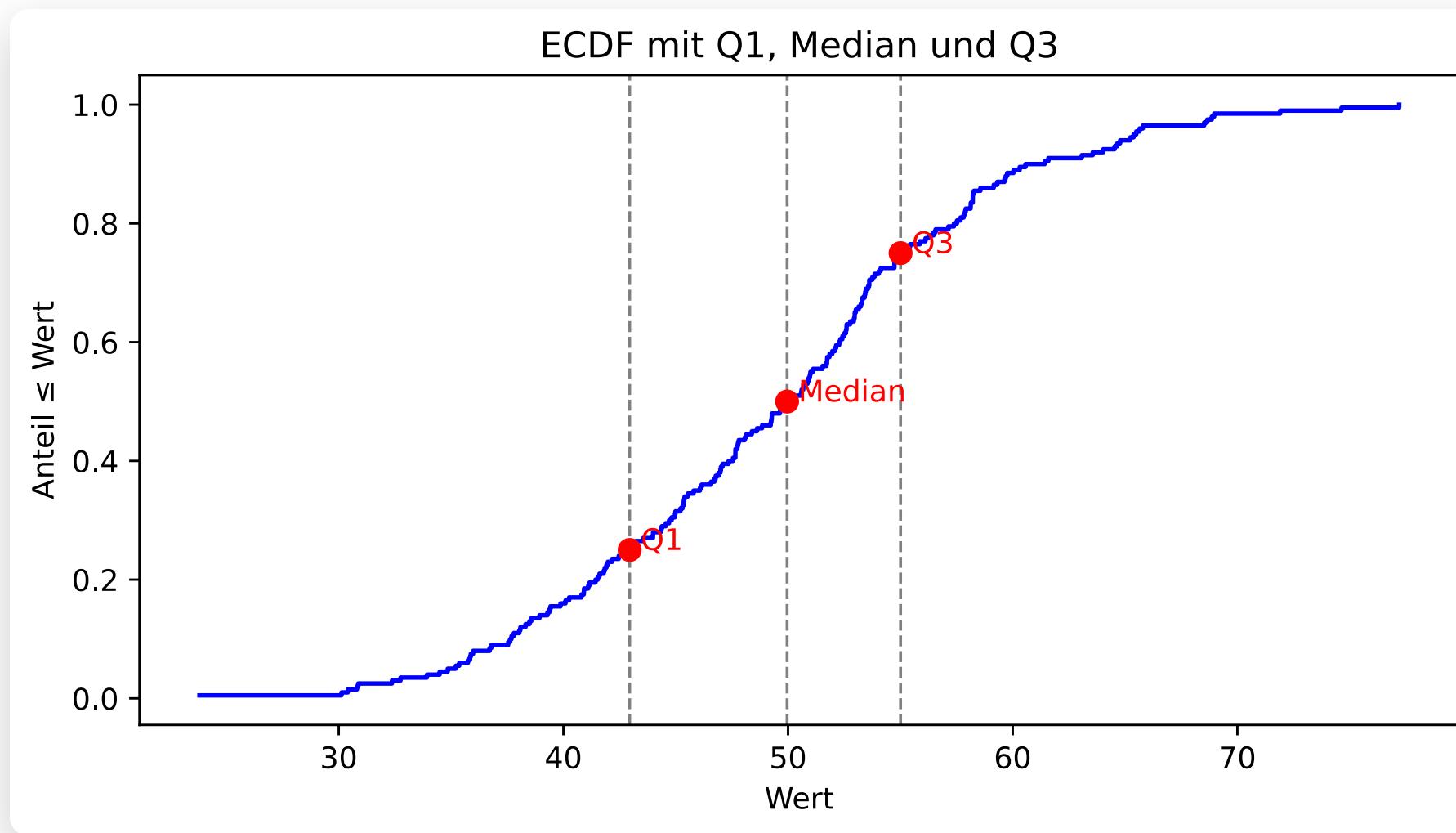
Quantile & Perzentile

Quantile teilen die Verteilung und bilden Basis für IQR (Interquartilsabstand).

- $Q_p = \inf\{x : F(x) \geq p\}$
- Quartile Q_1, Q_2, Q_3 , Perzentile
- Software-Varianten beachten
- Python: `s.quantile([.25, .5, .75])`

Mini-Check: Wann nutzt du $Q_{0.9}$ statt Median?

Empirische Kumulative Verteilungsfunktion (ECDF)



ECDF inverse zu Quantile

ECDF (Zahl → Prozent)

Zahl (x)	Anteil $\leq x$ (y)
3	20 %
4	60 %
5	80 %
6	100 %

Quantile (Prozent → Zahl)

Prozent (p)	Zahl (Q_p)
0.25	4
0.50 (Median)	4
0.75	5

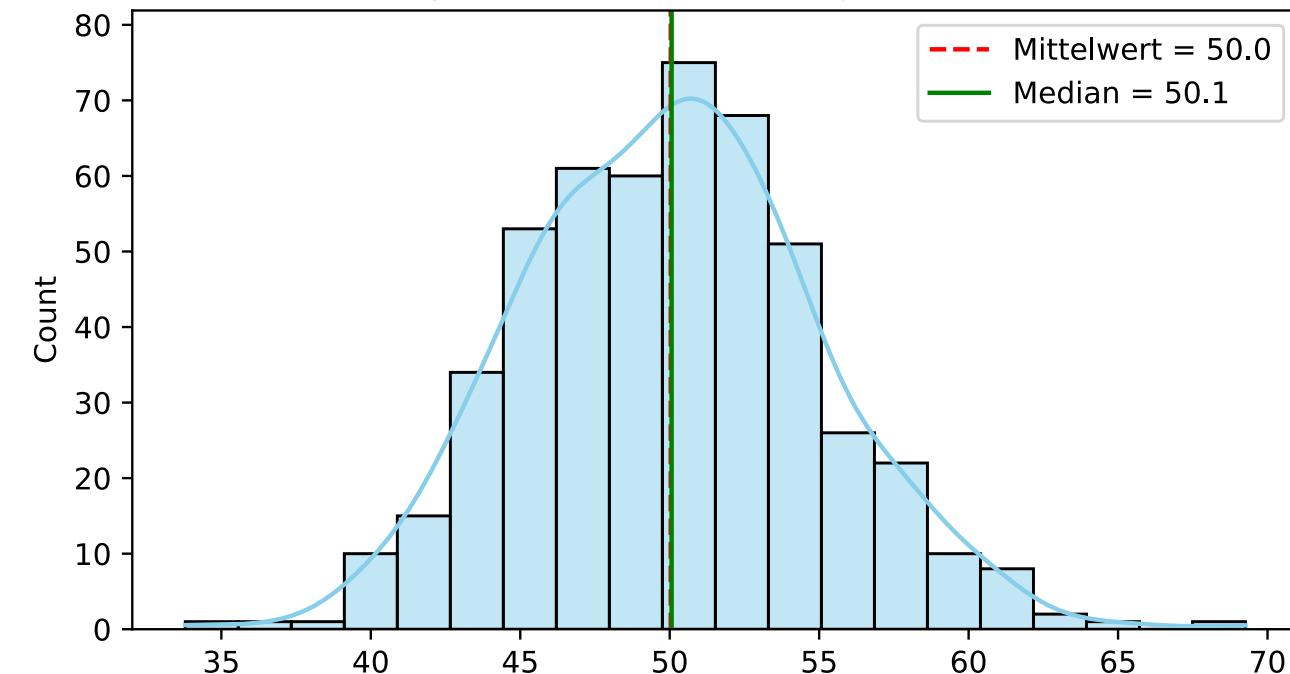
Mittelwert vs. Median bei Rechts-Schiefe

Bei Rechts-Schiefe liegt Mittelwert oft über Median.

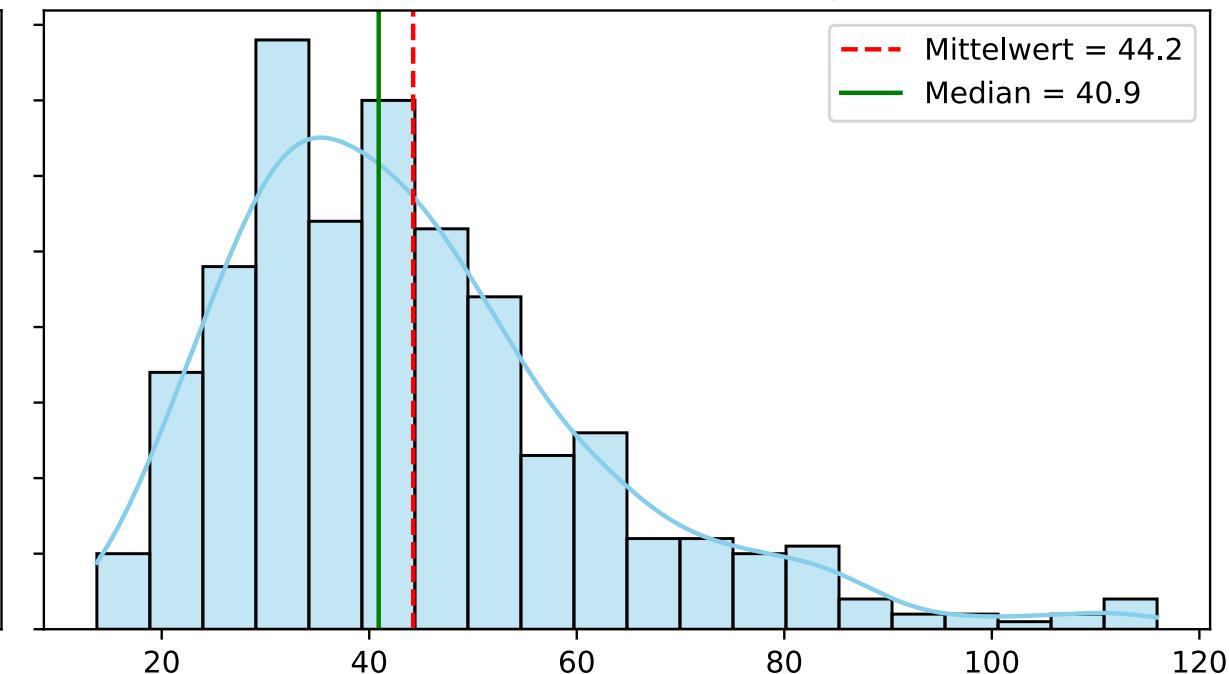
- Beispiel `[30, ..., 500]` → \bar{x} 104.6, Median 40
- Ein Ausreisser reicht für grosse Differenz
- Empfehlung: immer Median und IQR berichten
- Python: `s.mean()` vs. `s.median()`
- **Mini-Check:** Welche Kennzahlen für Notaufnahme-Wartezeiten?

Mittelwert vs. Median bei Symmetrie und Rechts-Schiefe

Symmetrische Verteilung (Normal)



Rechtsschiefe Verteilung



Python Basics Lage

Lagekennzahlen sind Einzeiler.

```
import pandas as pd  
s = pd.Series([30,32,35,40,45,50,500])  
s.mean(), s.median(), s.quantile([.25,.5,.75])
```

- NaN = Not a Number = fehlender Wert
- Vor Berechnung prüfen: entfernen (dropna) oder ersetzen (Imputation)
- Ergebnis kurz interpretieren,
- **Mini-Check:** Wie Strategien kennst du für Nan-Umgang?

Code

```
import pandas as pd
import numpy as np
s = pd.Series([30, 32, np.nan, 35, 40, 45, 50, 500, np.nan])
s = s.dropna()
print("Mittelwert:", s.mean())
print("Median:", s.median())
print("Quantile:")
print(s.quantile([0.25, 0.5, 0.75]))
```

Ausgabe

```
Mittelwert: 104.57
Median: 40.0
Quantile:
0.25      33.5
0.50      40.0
0.75      47.5
dtype: float64
```

Die typischen Einkommen liegen im Bereich von 33.5 bis 47.5 (IQR). Der Median von 40 zeigt die Mitte. Der Mittelwert von 104.6 ist viel höher, weil ein einziger Ausreißer (500) das Ergebnis stark verzerrt.

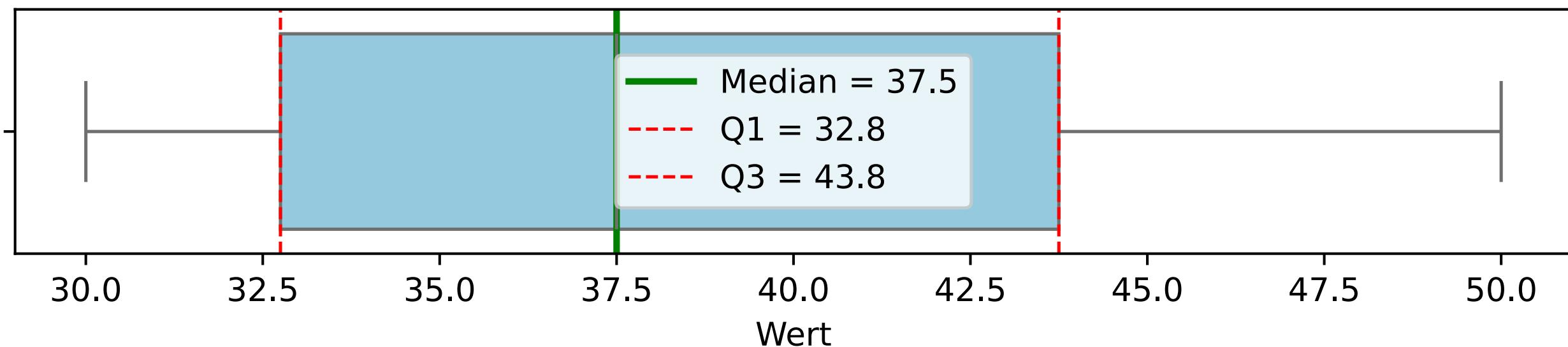
Mini-Datensatz interpretieren

Zahlen brauchen Deutung.

```
import seaborn as sns  
df = sns.load_dataset("penguins").dropna(subset=["flipper_length_mm"])  
s = df["flipper_length_mm"]  
s.median(), s.quantile([.25,.75])
```

- Typische Länge, Streuung, mögliche Gruppen
- Gruppenvergleich später
- **Mini-Check:** Was sagt ein grosser IQR?

Boxplot mit Median und Quartilen (ohne Ausreisser)



Typische Fehler bei Lagemassen

Standardfallen vermeiden.

- Mittelwert allein bei Schiefe
- Ausreisser nicht geprüft
- NaN ignoriert
- Einheiten gemischt
- **Mini-Check:** Welche zwei Checks vor Mittelwert?

Key Takeaways – Lagekennzahlen

- **Mittelwert:** effizient, aber ausreisserempfindlich
- **Median:** robust, besser bei Schiefe
- **Getrimmtes Mittel:** Kompromiss zwischen Median und Mean
- **Modus:** nützlich für kategoriale Daten, Achtung bei stetigen Werten
- **Quantile:** Grundlage für robuste Masse (z. B. IQR)
- **Praxis:** Bei schiefen Daten immer Median + IQR berichten

 Robustheitsmasse geben realistischere Bilder als nur der Mittelwert.

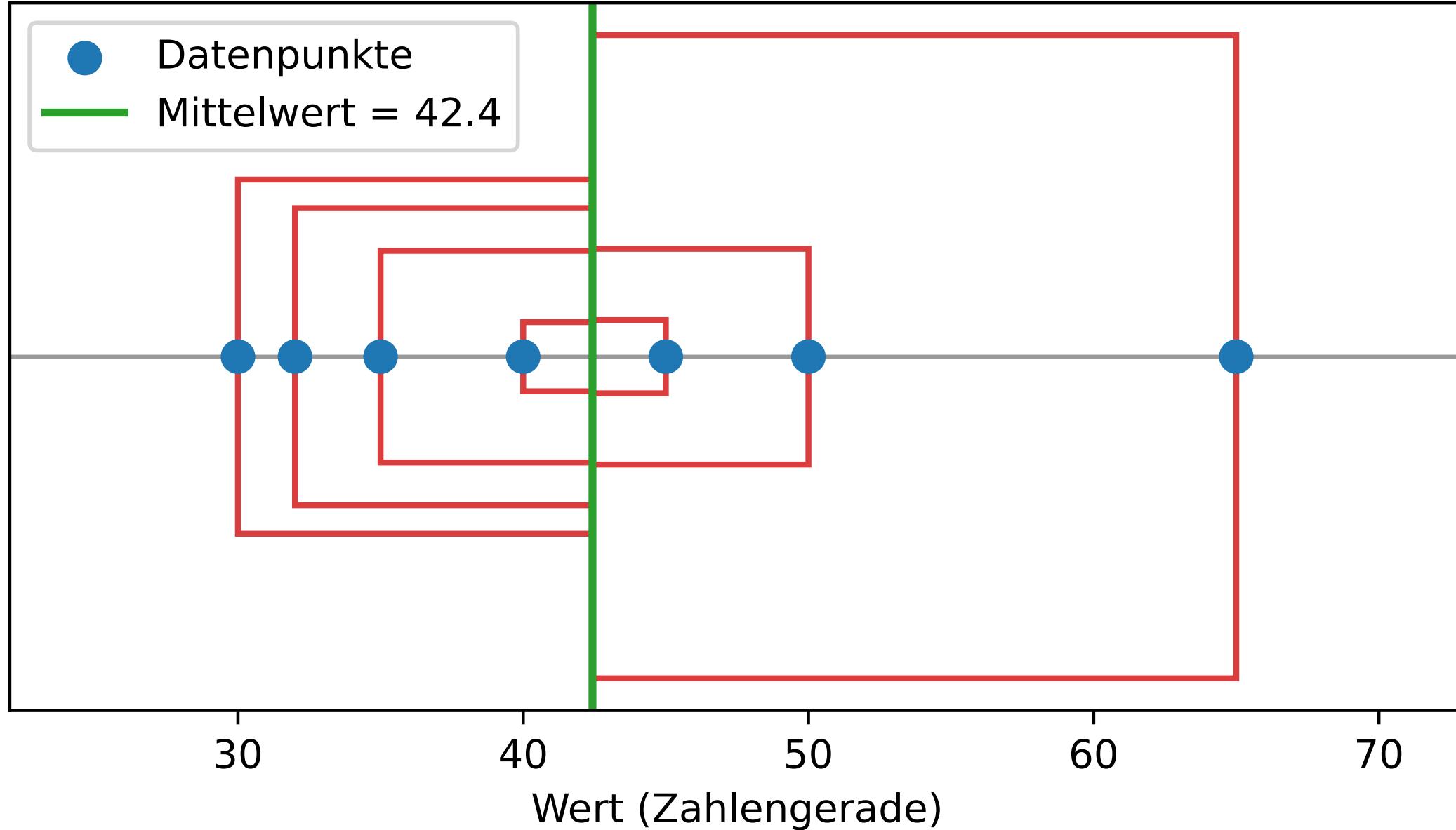
2. Streuungskennzahlen

Varianz und Standardabweichung

Messen mittlere quadratische Abweichung.

- Population: $\text{Var}(X) = E[(X - \mu)^2]$, $\sigma = \sqrt{\text{Var}}$
- Stichprobe: $s^2 = \frac{1}{n-1} \sum(x_i - \bar{x})^2$
- Einheit: Standard Deviation (SD) in Originaleinheit
- Ausreisserempfindlich
- **Mini-Check:** Warum $n - 1$ im Nenner?

Quadrate der Abstände zum Mittelwert (beitragen zur Varianz)

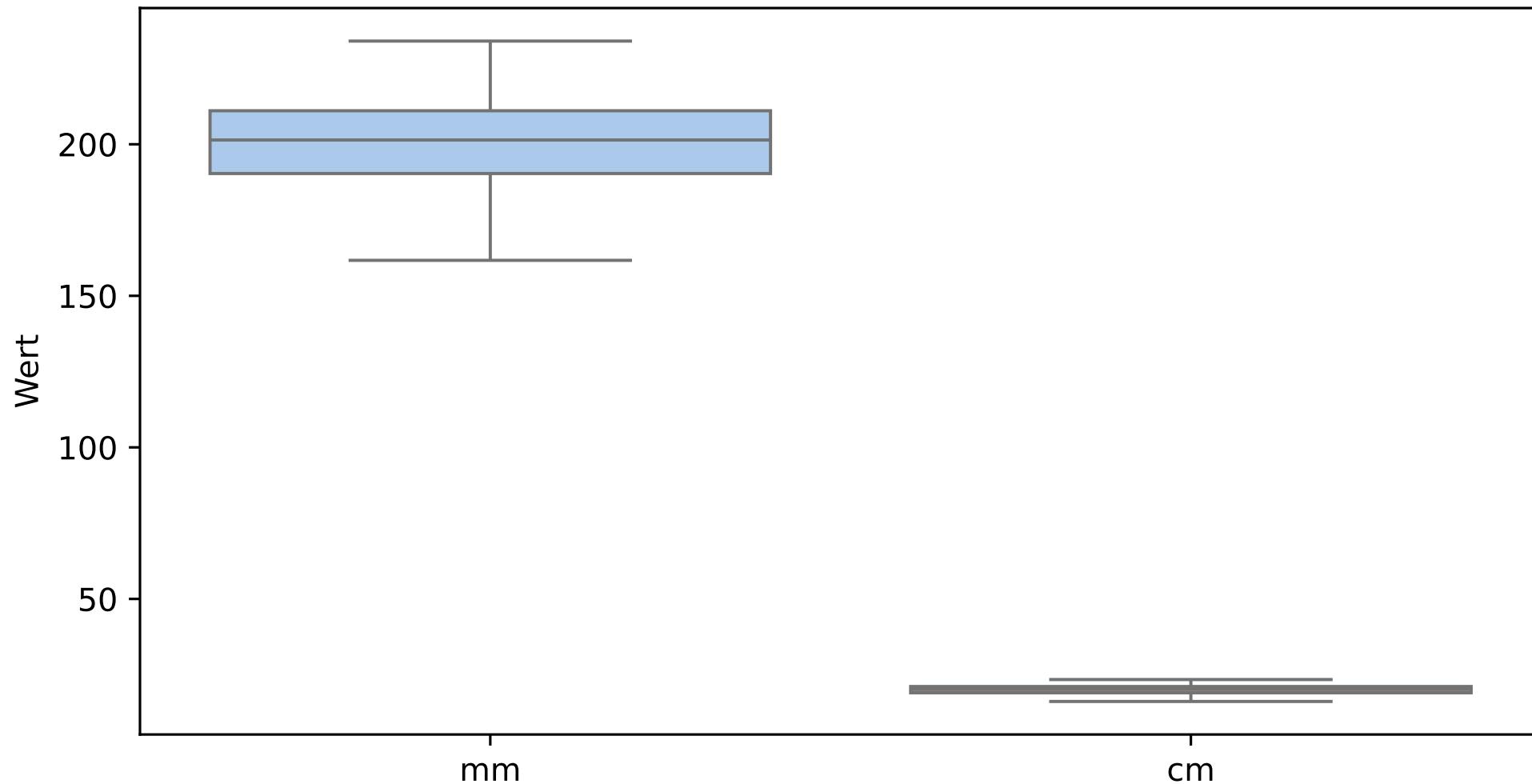


Eigenschaften von Varianz und SD

Skalierungsregeln gezielt nutzen.

- Skalierungsregeln: $\text{Var}(aX + b) = a^2\text{Var}(X)$
- Skalierungsregeln: $\text{SD}(aX + b) = |a|\text{SD}(X)$
- Unabhängig: $\text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y)$
- Alternative Darstellung: $\text{Var}(X) = E[X^2] - \mu^2$
- **Mini-Check:** Was passiert bei mm zu cm?

Boxplot: Flossenlaenge in mm vs. cm



$SD \approx 15.2 \text{ mm}$
 $Var \approx 230.9 \text{ mm}^2$

$SD \approx 1.5 \text{ cm}$
 $Var \approx 2.31 \text{ cm}^2$

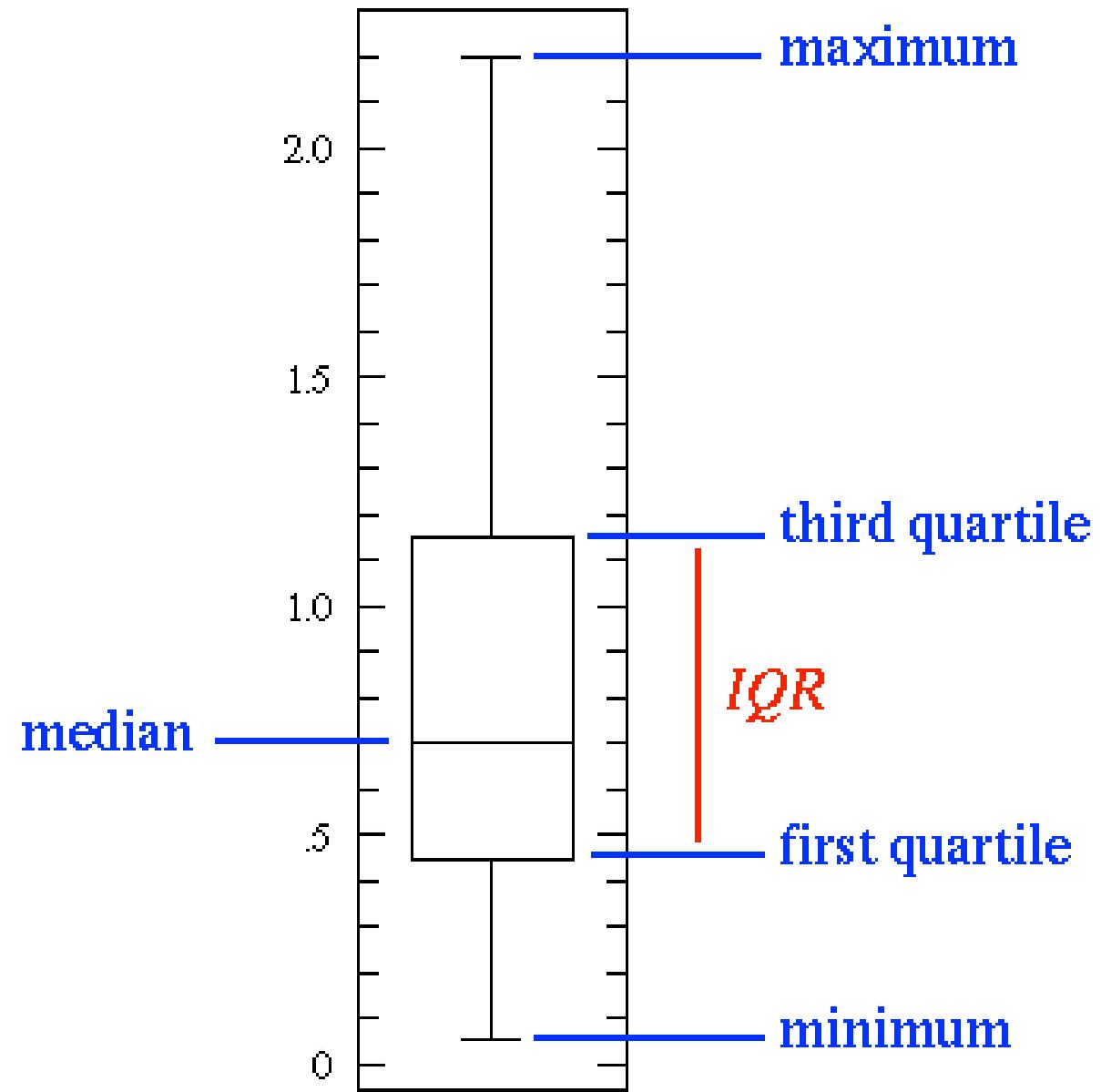


IQR als robuste Streuung

IQR (Interquartile Range – Interquartilsabstand)

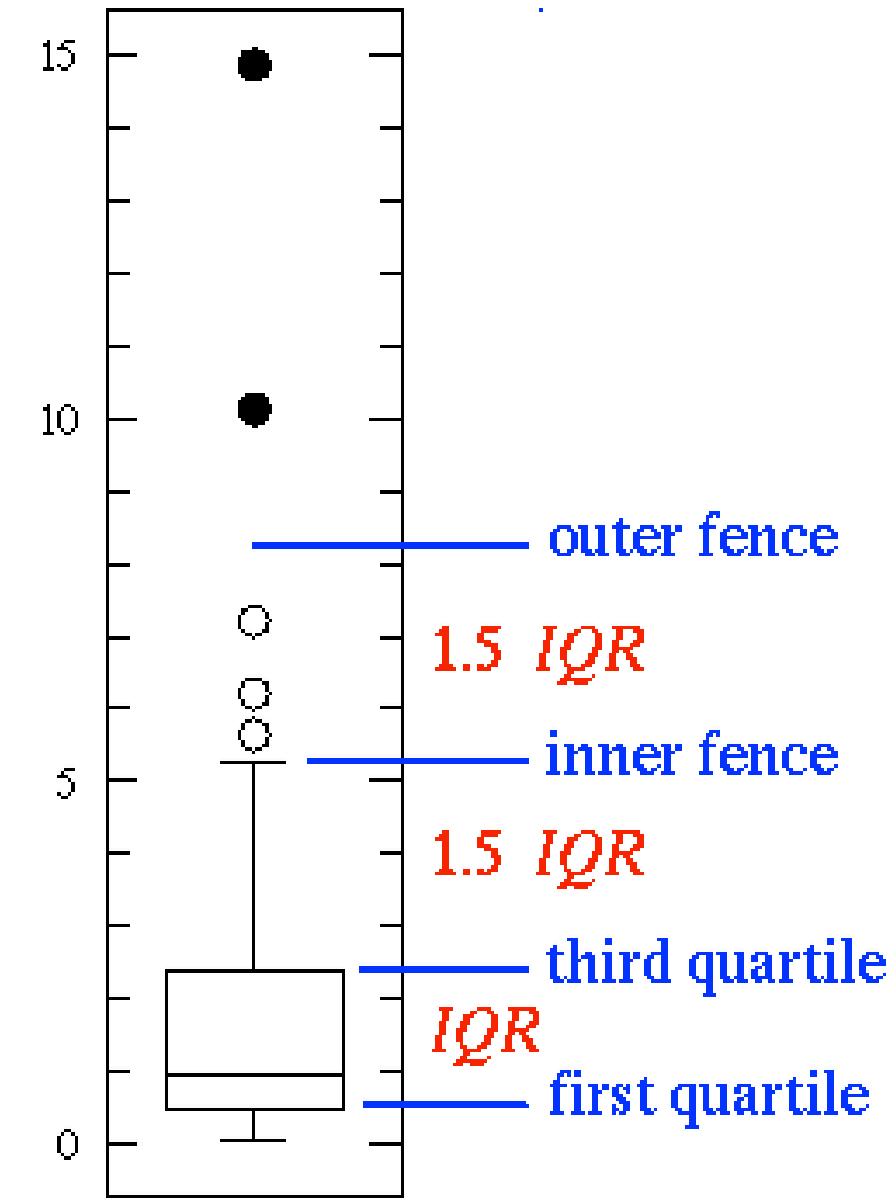
IQR misst mittlere 50 Prozent robust.

- $IQR = Q_3 - Q_1$
- Unempfindlich gegen wenige Extreme
- Immer mit Median melden
- Typische Werte im IQR
- **Mini-Check:** Warum bei Wartezeiten sinnvoll?



outliers

suspected outliers



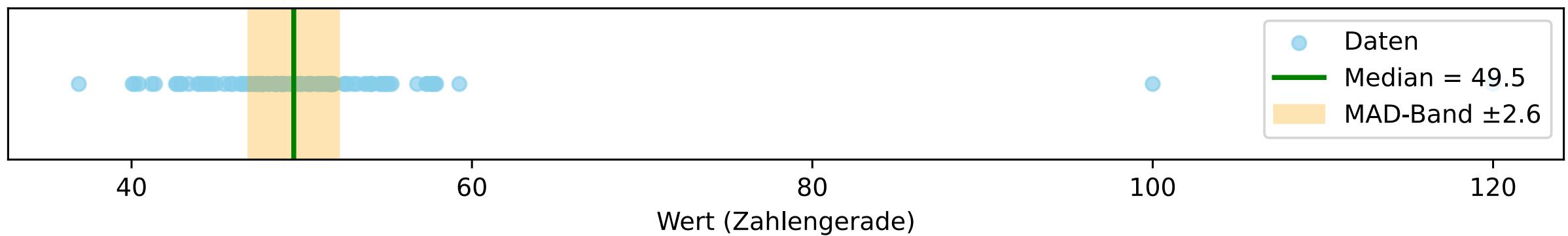
MAD robuste Alternative

MAD (Median Absolute Deviation – Mittlere absolute Abweichung)

MAD nutzt Abstände zum Median.

- $\text{MAD} = \text{median}(|x - \tilde{x}|)$
- Skaliert: $1.4826 \cdot MAD \approx \hat{\sigma}$ (für Normalverteilung)
- Ergänzt IQR und Median
- Gut bei Schiefe und Heavy Tails
- **Mini-Check:** Wann MAD dem IQR vorziehen?

Median Absolute Deviation (MAD)

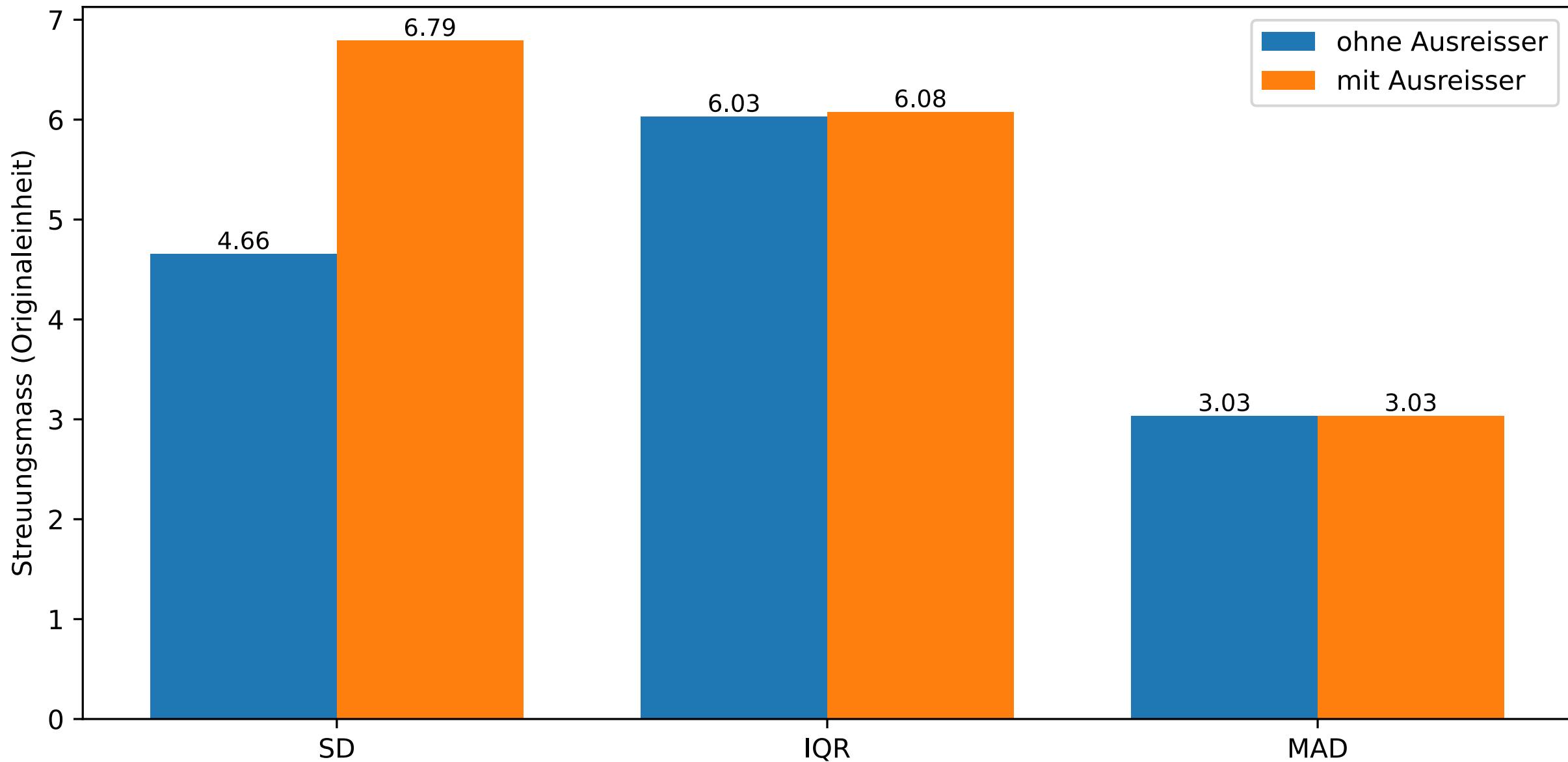


Klassisch vs. robust im Vergleich

SD reagiert stark, IQR und MAD bleiben stabil.

- Beispiel ohne Ausreisser: ähnlich
- Beispiel mit Ausreisser: SD hoch, IQR/MAD stabil
- Bericht: beides, Entscheidung begründen
- Sprache: „typische Streuung über IQR, MAD“
- **Mini-Check:** Domäne für robuste Masse?

Klassisch vs. robust: SD reagiert stark, IQR/MAD bleiben stabil



Python Basics Streuung

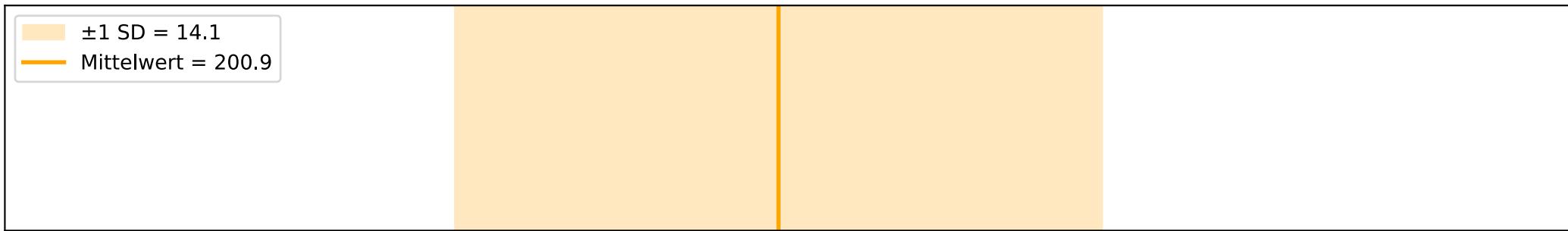
SD, IQR, MAD schnell berechnen.

```
s.std(ddof=1), s.var(ddof=1)  
q1,q3 = s.quantile(.25), s.quantile(.75); iqr = q3 - q1  
med = s.median(); mad = (s - med).abs().median(); mad_sigma = 1.4826*mad
```

- ddof (delta degrees of freedom) für Stichprobe → so aus wie „doof“, aber gemeint ist ddof
- **Mini-Check:** Wozu ddof=1?

Streuung im Vergleich: SD (um Mean), MAD (um Median), IQR

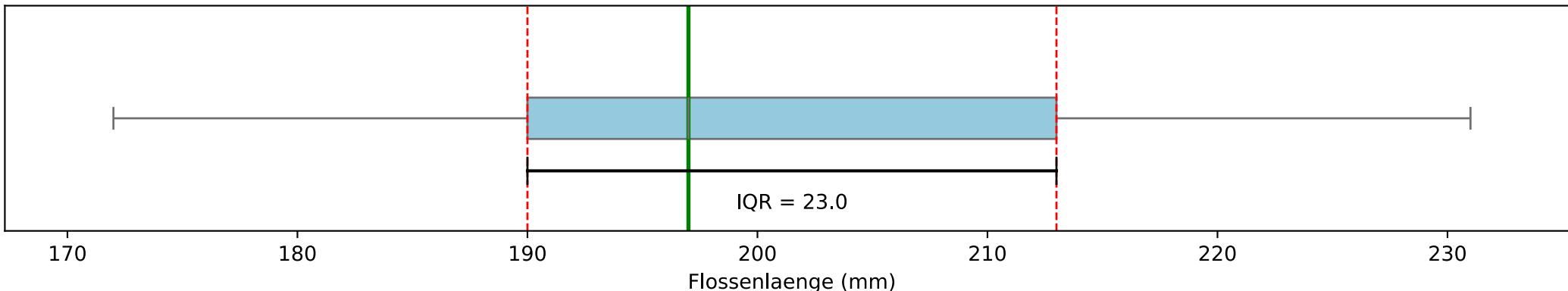
Mittelwert & \pm SD



Median & \pm MAD



Boxplot mit IQR



Exkurs: ddof (delta degrees of freedom)

Freiheitsgrade-Korrektur

Stichproben brauchen Korrektur.

- Beobachtung: Viele Stichproben → Varianz mit Nenner **n** liegt im Schnitt **unter** der wahren Varianz
- Konsequenz: Man teilt durch **n–1**, damit es im Mittel passt

Statistiker haben es überprüft:

- Mit n → systematisch zu klein
- Mit $n - 1$ → erwartungstreu, im Mittel korrekt

Merksatz:

- $\text{ddof}=0$ → **Population** (alle Daten)
- $\text{ddof}=1$ → **Stichprobe** (üblich in Statistik)

Python Basics Streuung

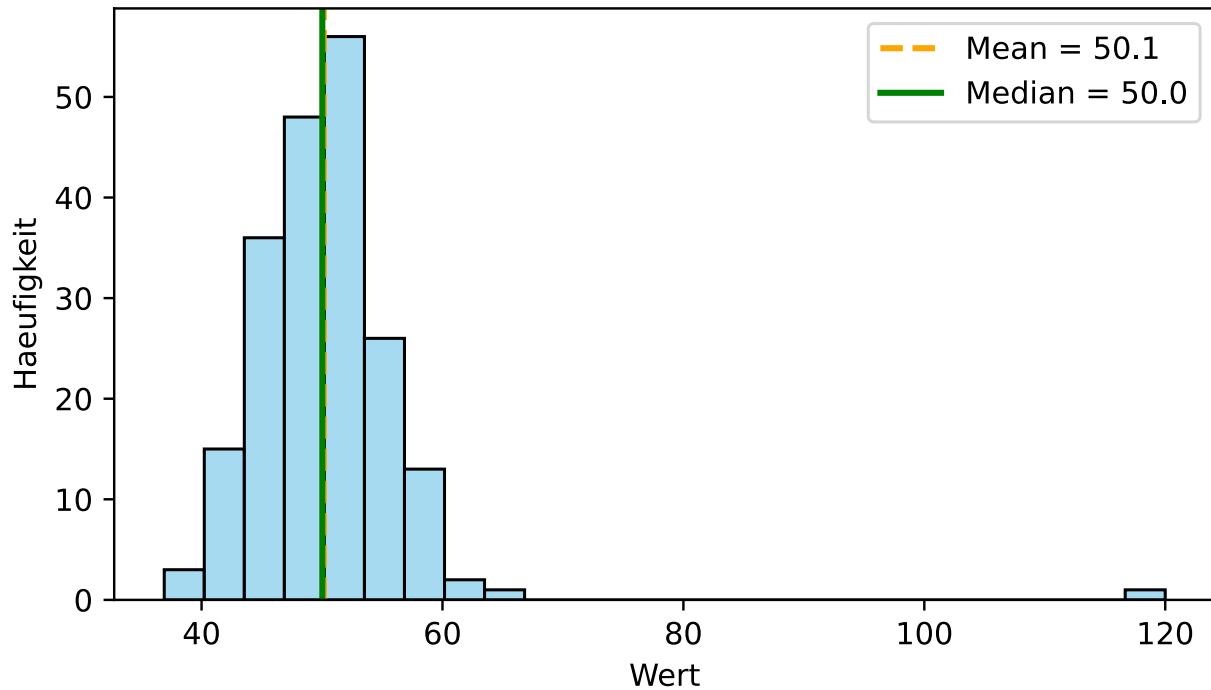
SD, IQR, MAD schnell berechnen.

```
s.std(ddof=1), s.var(ddof=1)  
q1,q3 = s.quantile(.25), s.quantile(.75); iqr = q3 - q1  
med = s.median(); mad = (s - med).abs().median(); mad_sigma = 1.4826*mad
```

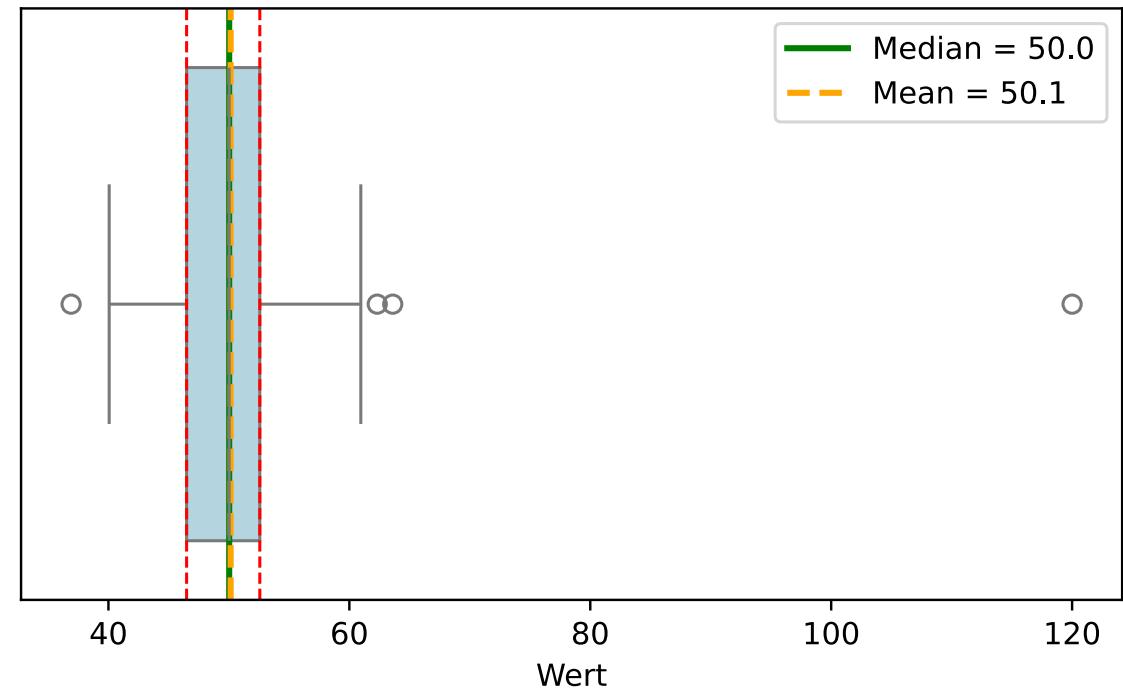
- ddof (delta degrees of freedom) für Stichprobe → so aus wie „doof“, aber gemeint ist ddof
- **Mini-Check:** Wozu ddof=1?

Interpretation derselben Daten: Robust vs. klassisch

Histogramm mit Ausreißer



Boxplot mit Ausreißer



Häufige Fehler bei Streuung

Formale Fehler erzeugen Fehlurteile.

- **ddof** falsch
- Range statt robuster Masse
- Keine SD ohne Mittelwert
- Einheit fehlt
- **Mini-Check:** Zwei Checks vor SD-Reporting

Gute Berichtssprache

Klare Sätze erhöhen Verständlichkeit.

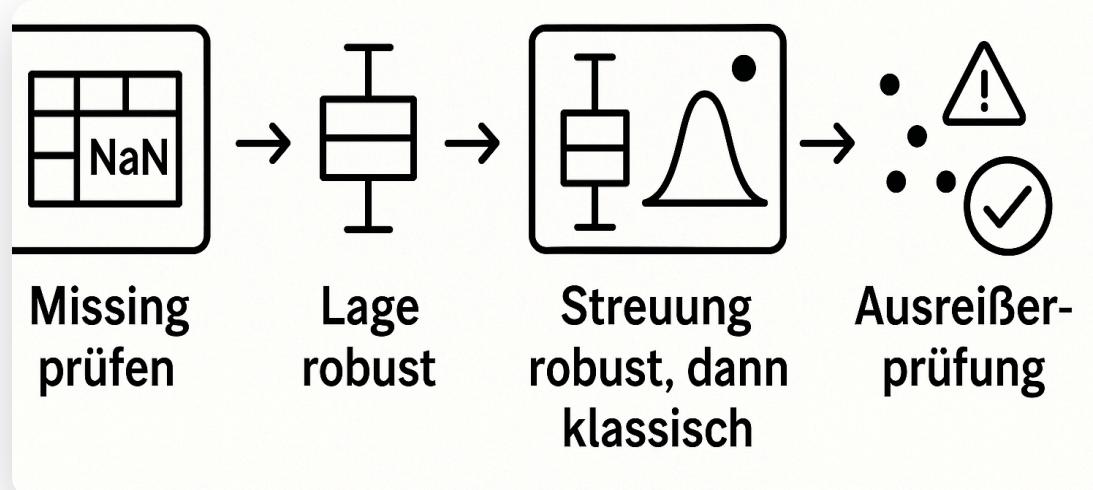
- Robust: „Median = ..., IQR = ..., MAD = ...“
- Klassisch: „Mittelwert = ..., SD = ...“
- Alltagssprache ohne Jargon
- NaN und Datengrundlage nennen
- **Mini-Check:** Satz mit SD und Einheit formulieren

 Sprache macht Statistik verständlich.

Streuungs-Profil als Routine

Fixe Reihenfolge sichert
Qualität.

1. Missing prüfen
2. Lage robust, dann klassisch
3. Streuung robust, dann klassisch
4. Ausreißerprüfung ankündigen



Key Takeaways – Streuungskennzahlen

- **Varianz & SD:** weit verbreitet, aber empfindlich für Ausreisser
 - **IQR:** robuste Spannweite der mittleren 50 %
 - **MAD:** medianbasierte Abweichung, sehr robust
 - **Vergleich:** robust zuerst berichten, klassisch ergänzen
 - **Routine:** Reihenfolge einhalten → Missing → Lage → Streuung → Ausreisser
-  Robuste Massen liefern das realistischere Bild bei echten Daten.

3. Ausreisser erkennen und behandeln

Drei Ansätze zur Ausreißer-Erkennung

Verfahren	Basis	Eignung
Klassischer Z-Score	Mittelwert & Standardabw.	sinnvoll nur bei (fast) normalverteilten Daten
Tukey-Fences (Boxplot)	Quartile & IQR	robust, keine Verteilungsannahme
Modifizierter Z-Score	Median & MAD	sehr robust, geeignet bei Schiefe & Heavy Tails

Klassischer Z-Score

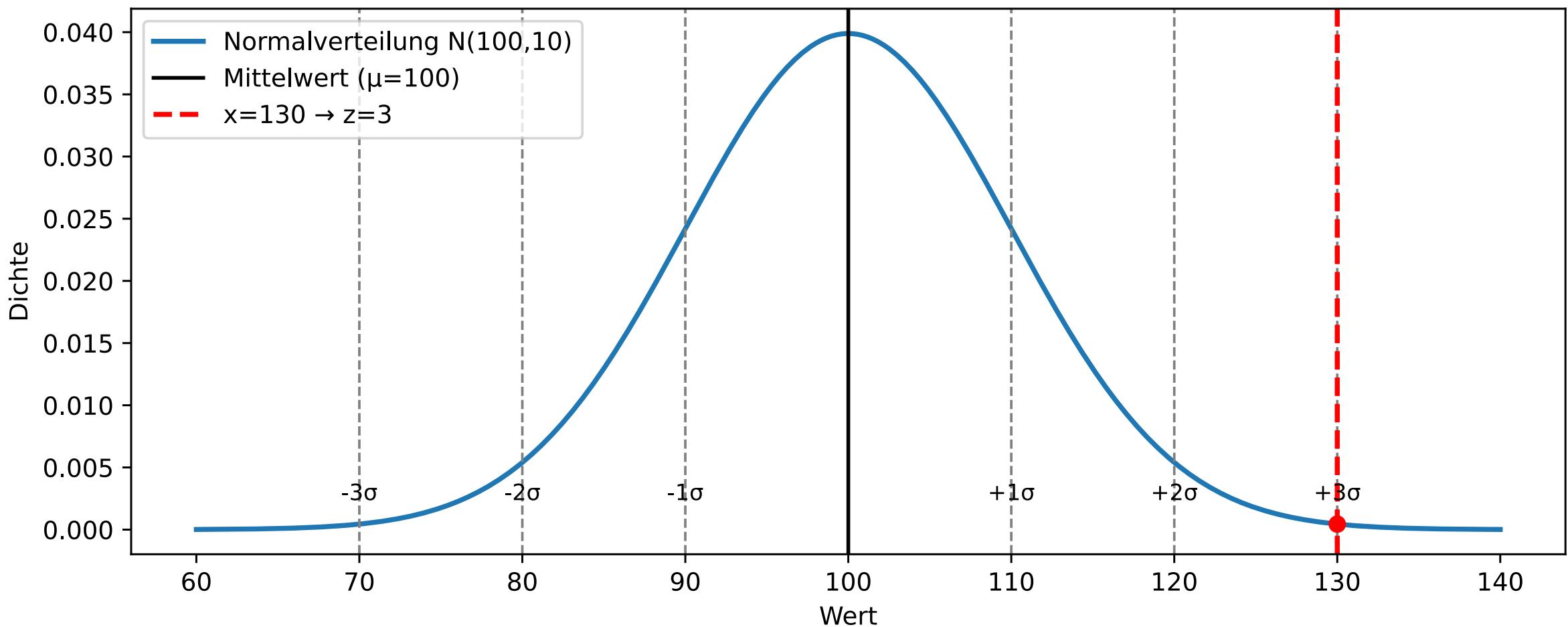
- Misst, wie viele Standardabweichungen ein Wert vom Mittelwert entfernt ist
- Definition:

$$z_i = \frac{x_i - \bar{x}}{s}$$

- Annahme: Daten sind (nahezu) normalverteilt
- Typische Schwelle: $|z| > 3 \rightarrow$ Ausreißerkandidat

👉 Nützlich bei Normalverteilung, empfindlich bei Schiefe und Ausreissern.

Z-Score Beispiel: $x=130, \mu=100, s=10 \rightarrow z=3$

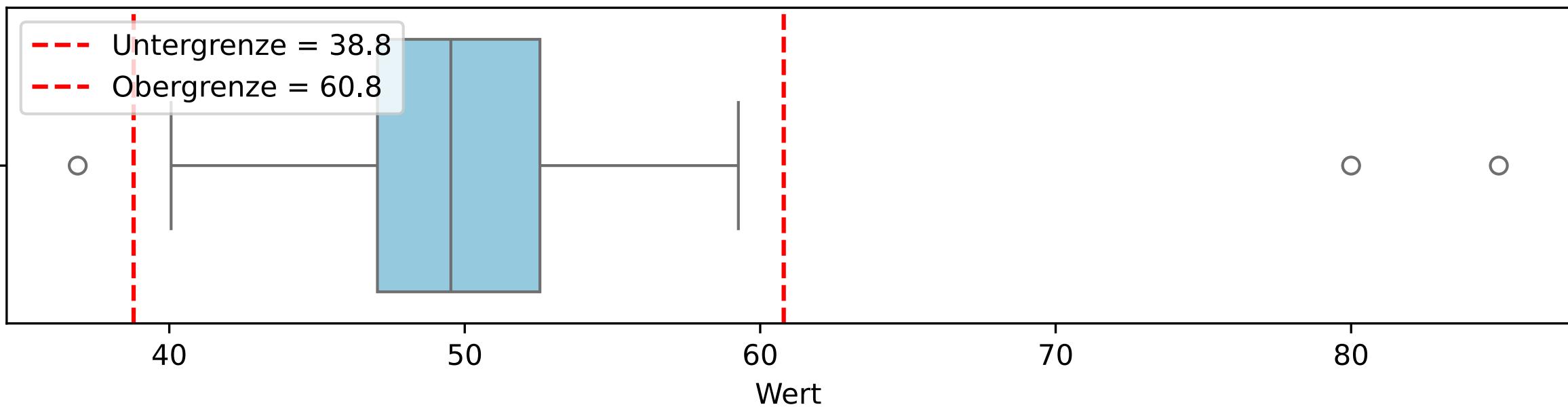


Tukey Fences mit Quartilen

IQR-Regel liefert robuste Kandidaten.

- $IQR = Q_3 - Q_1$
- Untere Grenze $Q_1 - 1.5 \cdot IQR$
- Obere Grenze $Q_3 + 1.5 \cdot IQR$
- Identifiziert Kandidaten, ohne Verteilungsannahme
- **Mini-Check:** Was passiert bei breiterer Verteilung?

Boxplot mit Tukey-Fences (1.5·IQR)

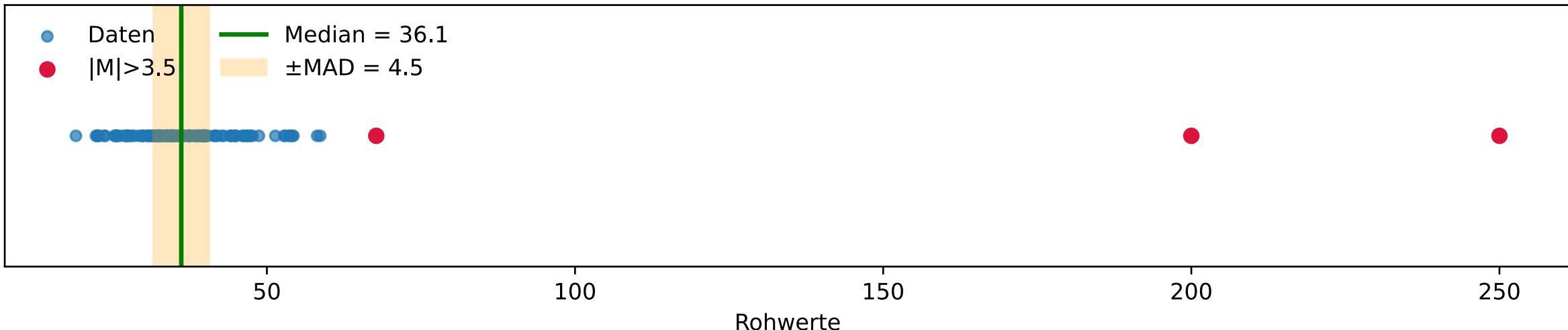


Modifizierter Z-Score mit MAD

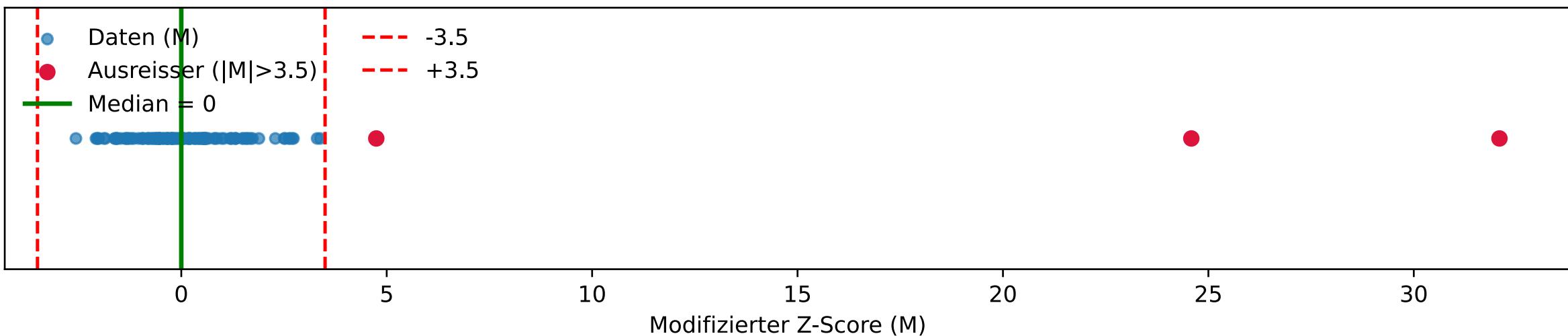
Robuste Lage und Streuung kombiniert.

- $\text{MAD} = \text{median}(|x - \tilde{x}|)$
- $M_i = 0.6745 \cdot \frac{x_i - \tilde{x}}{\text{MAD}}$ → lineare Transformation
- Schwelle: $|M_i| > 3.5 \rightarrow \text{Ausreisser}$
- Gut bei Schiefe
- **Mini-Check:** Warum MAD stabiler als SD?

Median + MAD-Band (Rohwert-Skala)



Schwellen ± 3.5 (M-Skala)



Einfluss von Ausreisern

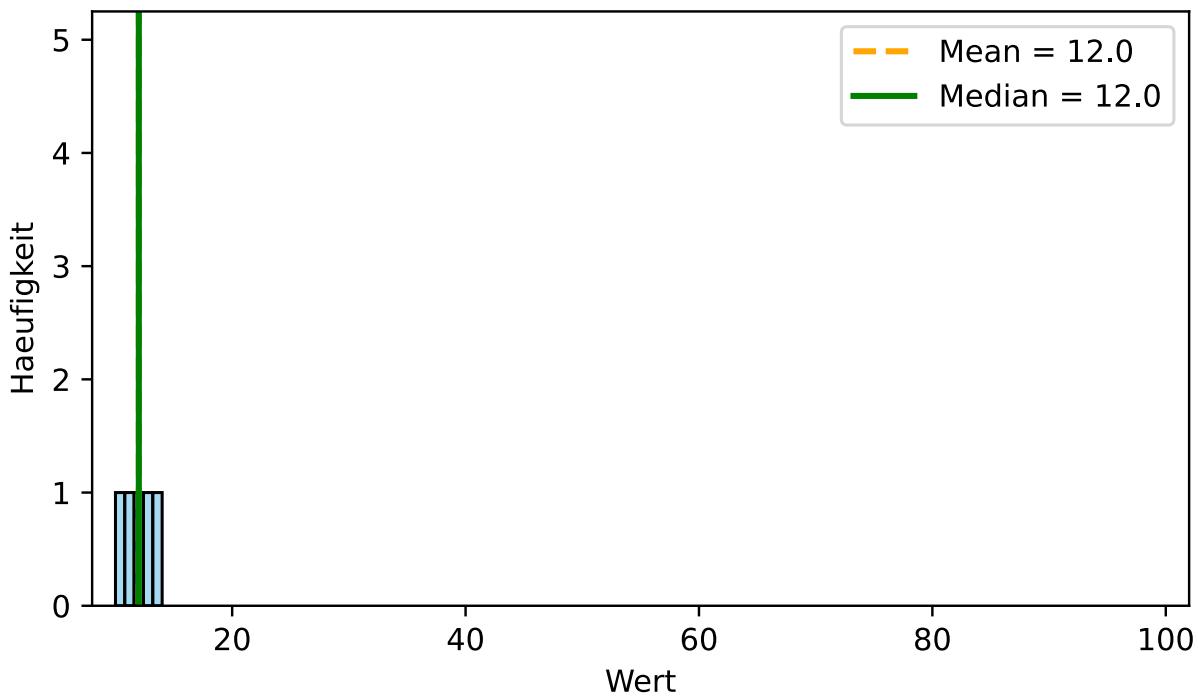
Einzelne Extremwerte ziehen Mean & SD stark nach oben

- Beispiel [10, 11, 12, 13, 14, 100]
- \bar{x} steigt stark durch den Ausreißer
- s (Standardabweichung) bläht sich auf
- Median & IQR verändern sich nur moderat
- Vor Modellierung prüfen

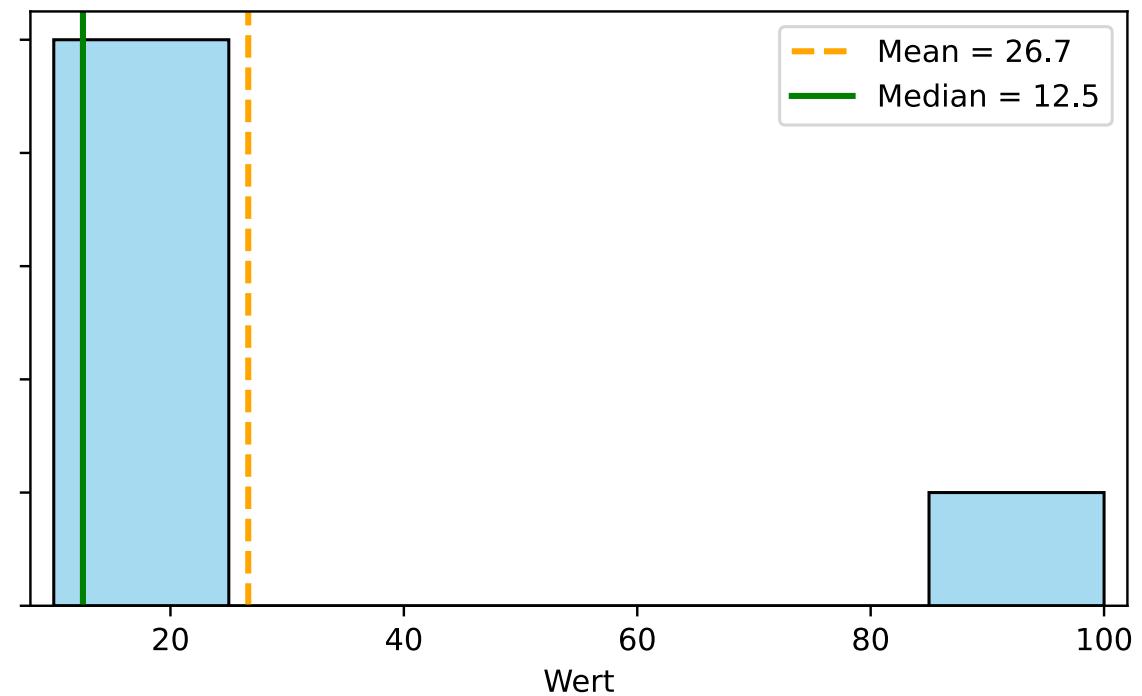
Mini-Check: Welche Kennzahl kippt zuerst und warum?

Ein Ausreißer zieht Mean & SD stark - Median & IQR bleiben stabil

Ohne Ausreißer



Mit Ausreißer (100)

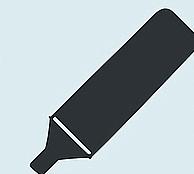


Strategien im Umgang

Markieren, winsorisieren, trimmen mit Begründung.

- Markieren und separat berichten
- Winsor: auf Grenze setzen (deckeln)
- Trimmen: Ränder entfernen
- Immer Entscheidung dokumentieren
- **Mini-Check:** Domäne für Markieren statt Entfernen?

Markieren



Ausreißer markieren,
separat berichten

Winsor



Werte auf
Grenze setzen

Trimmen



Ränder
entfernen

Dokumentieren



Entscheidung
festhalten

Python Tukey und mod. Z

Masken in wenigen Zeilen.

```
q1,q3 = s.quantile([.25,.75]); iqr = q3-q1  
lo,hi = q1-1.5*iqr, q3+1.5*iqr  
mask_tukey = (s<lo)|(s>hi)  
  
med = s.median(); mad = (s-med).abs().median()  
M = 0.6745*(s-med)/mad  
mask_modz = M.abs()>3.5
```

Beispiel-Output

```
Tukey: 2 Ausreisser markiert  
ModZ: 1 Ausreisser markiert
```

- **Mini-Check:** Warum sind Masken nützlich?

Fallnotiz und Reproduzierbarkeit

Entscheidungen schriftlich festhalten.

- Regel, Schwelle, Datum, Variable
- Maske und Datenversion speichern
- Varianten mit und ohne Ausreisser berichten
- Nutzen für Audit und Kollaboration
- **Mini-Check:** Drei Metadaten für die Notiz ?

Fallnotiz und Reproduzierbarkeit

Regel	Schwelle	Datum	Kommentar
mod. Z-Score	3,5	21.05.2025	Variable X markiert
IQR	$> Q3 + 1,5 * IQR$	15.06.2025	Nur in Variante B entfernt
Saubere Dokumentation erhöht Transparenz & Vertrauen.			

Dokumentation

Feld	Eintrag
Regel	Tukey 1.5 IQR + mod. Z 3.5
Schwelle & Grund	Robuste Erkennung, da Schiefe vermutet
Datum & Version	Datenstand 2025-09-20, Codeversion v1.3
Variable	flipper_length_mm
Ergebnis	5 Kandidaten, 3 real, 2 Messfehler
Entscheid	Markieren, Bericht mit und ohne

Heavy Tails vs echte Ausreisser

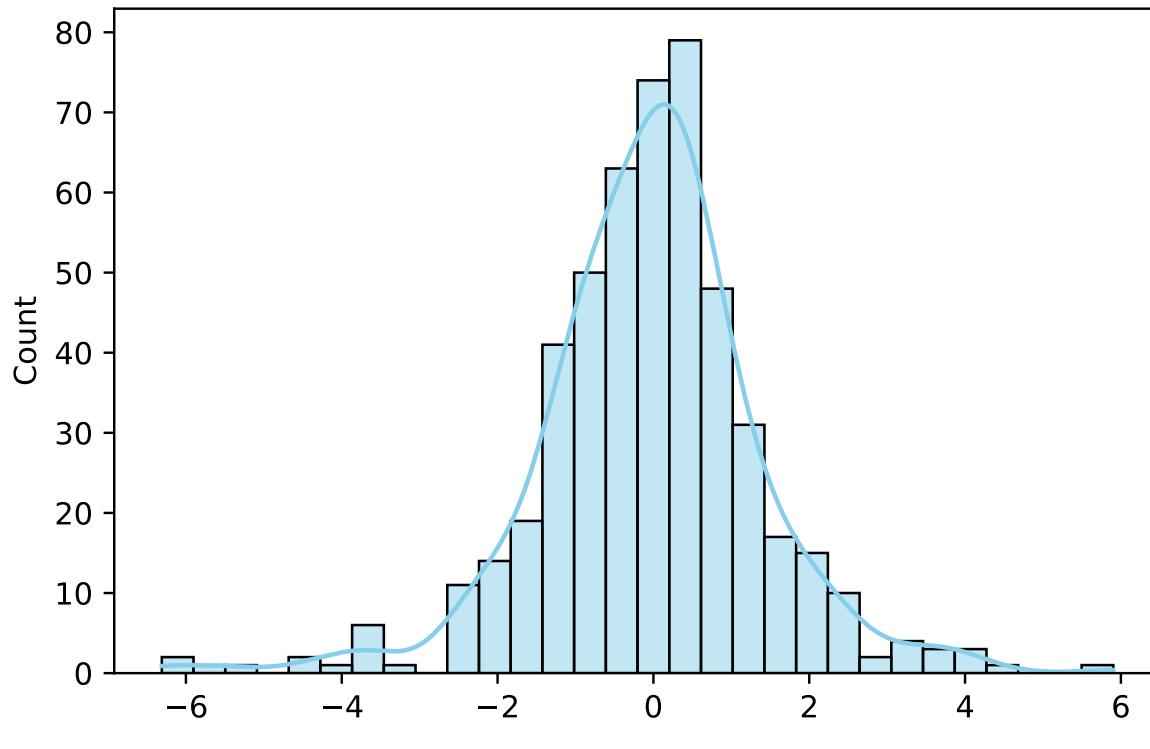
Heavy Tails (schwere Ausläufer) sind nicht automatisch Fehler.

- Heavy Tails = viele grosse Werte, nicht zwingend Fehler
- Echte Ausreisser = Messfehler, falsche Einheit, Datenproblem
- Diagnose mit mehreren Plots: Hist, KDE, ECDF, QQ
- Kontext immer berücksichtigen
- **Mini-Check:** Welcher Plot zeigt Tails klar?

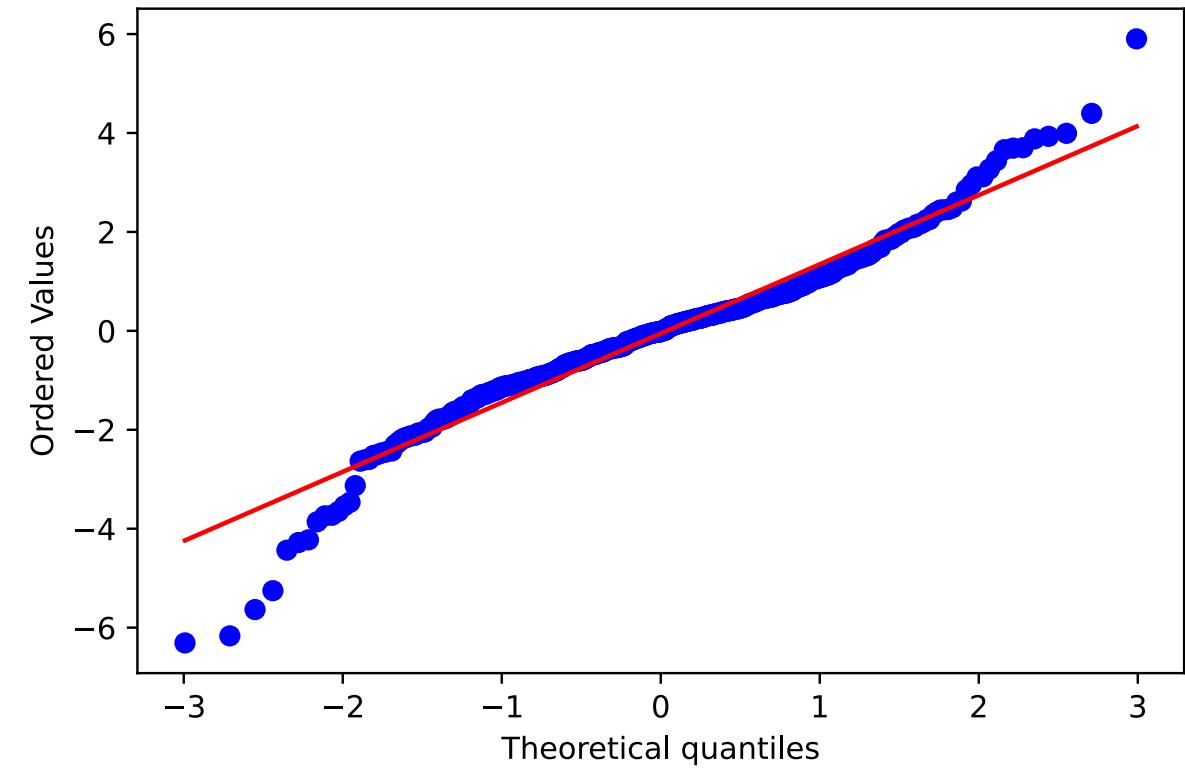


Nicht jeder Extremwert ist ein Fehler.

Histogramm + KDE (heavy tails)



QQ-Plot zeigt Abweichung in den Tails

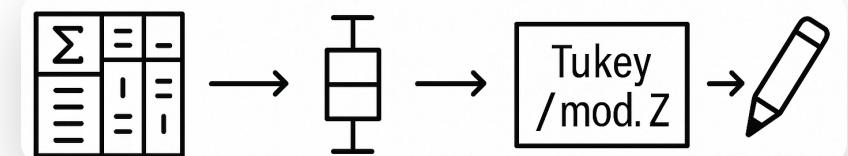


Pipeline: Kennzahl → Plot → Entscheid

Feste Reihenfolge verhindert Ad-hoc-Fehler.

1. Kennzahlen berechnen
2. Plots prüfen
3. Regel anwenden (z. B. Tukey, mod. Z)
4. Entscheidung dokumentieren

Zuerst **Zählen**, dann
Schauen, dann **Handeln**,
dann **Aufschreiben**.



👉 Standardisierte Pipeline verhindert Ad-hoc-Fehler.

Mini-Check: Warum erst robust dann Plots ?

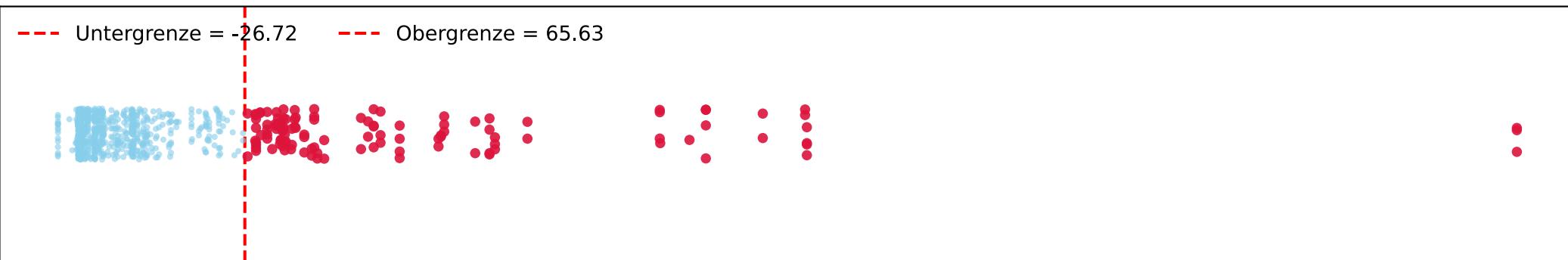
Mini-Aufgabe: Ausreißer finden

- Datensatz: Penguins `flipper_length_mm`
- Wende Tukey und mod. Z an
- Vergleiche Ergebnisse
- Formuliere eine kurze Begründung
- Optional: Winsorisieren

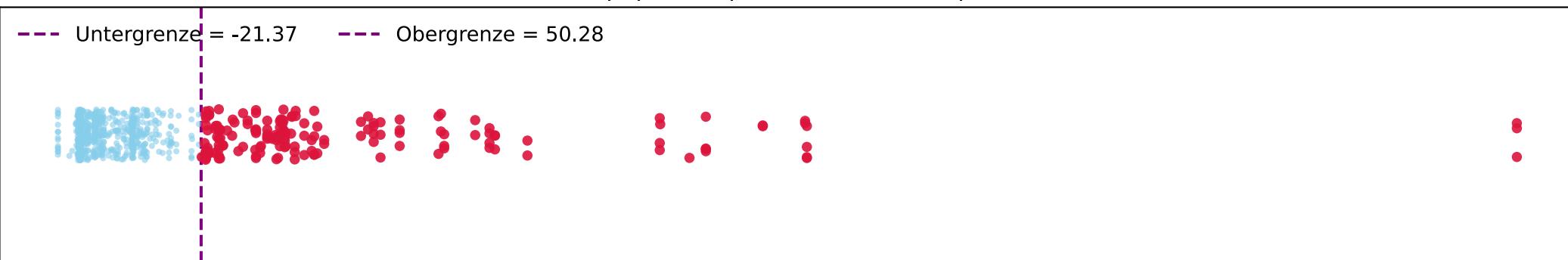


Methode wählen, begründen, dokumentieren.

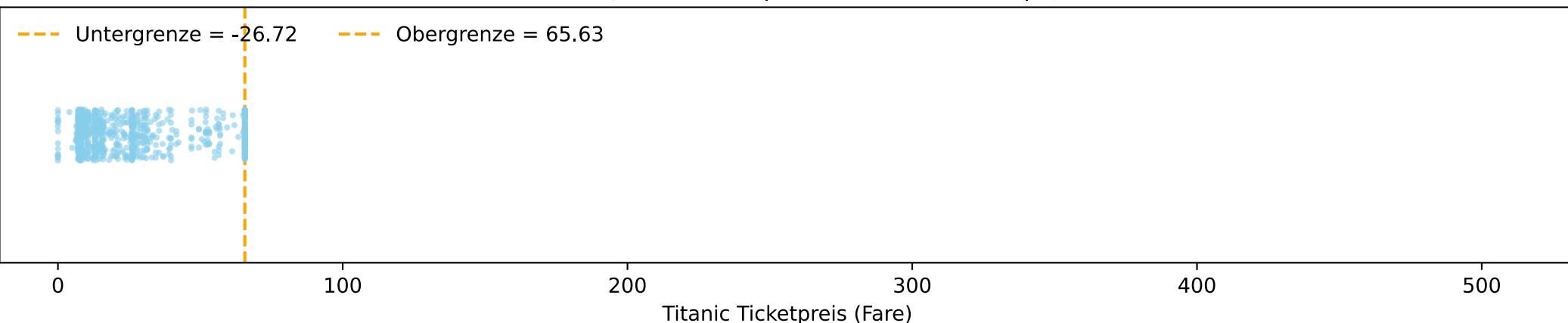
Tukey (1.5·IQR) | Ausreisser: 116 | n (Inlier): 775



Mod. Z-Score ($|M|>3.5$) | Ausreisser: 160 | n (Inlier): 731



Winsorisiert (auf Tukey-Grenzen) | n (innerhalb): 891 | gedeckelt: 116



Checkliste – Ausreißer-Handling

Einheitliche Checkliste für Konsistenz.

- Missing prüfen
- Tukey- und mod. Z-Test anwenden
- Heavy Tails vs. echte Fehler unterscheiden
- Strategie wählen & dokumentieren
- **Mini-Check:** Zwei Punkte, die oft fehlen



👉 Einheitliche Checkliste schützt vor Willkür.

Key Takeaways – Ausreißer erkennen & behandeln

- **Klassischer Z-Score:** basiert auf Mean & SD, nur bei Normalverteilung sinnvoll
- **Tukey-Fences:** IQR-basiert, robust, Standard im Boxplot
- **Mod. Z-Score:** Median & MAD, sehr robust bei Schiefe/Heavy Tails
- **Einfluss:** Mean & SD kippen schnell, Median & IQR bleiben stabil
- **Strategien:** Markieren, Winsorisieren, Trimen – Entscheidung dokumentieren
- **Praxis:** Heavy Tails \neq Fehler → immer Diagnose vor Eingriff

 Outlier-Erkennung = Methode wählen + Kontext verstehen + dokumentieren.

4. Histogramm und KDE

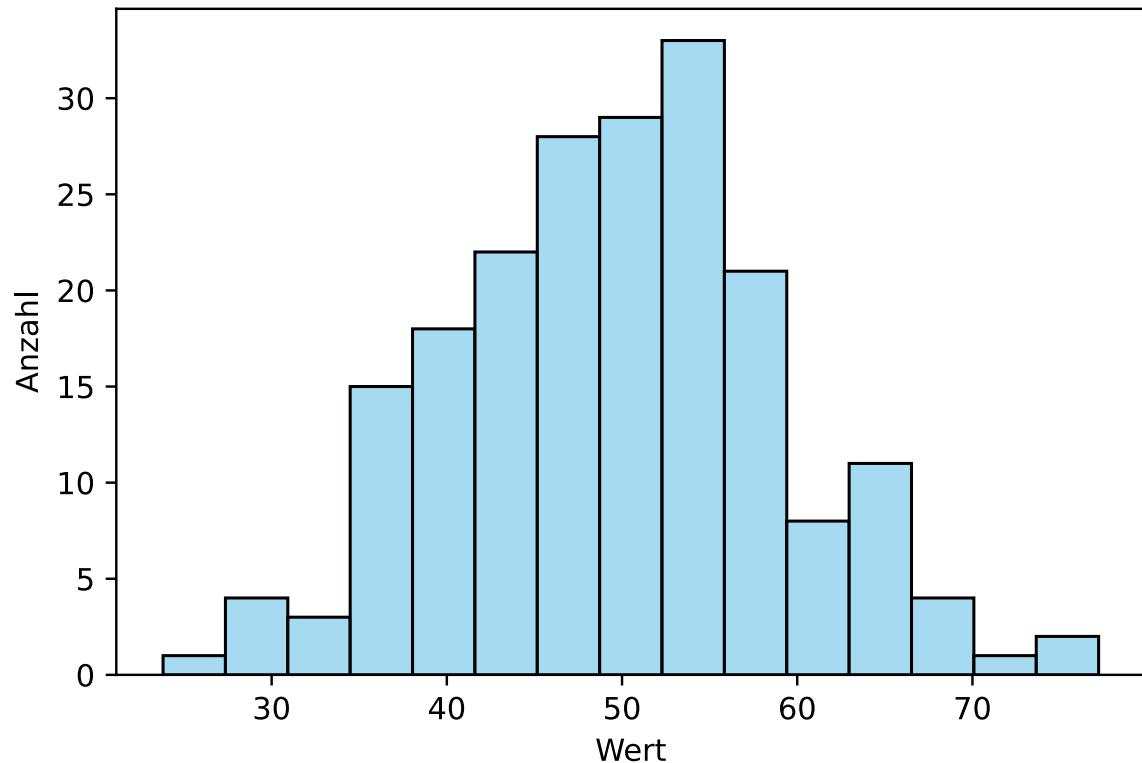
(Kernel Density Estimation – Kerndichteschätzung)

Histogramm – Grundidee

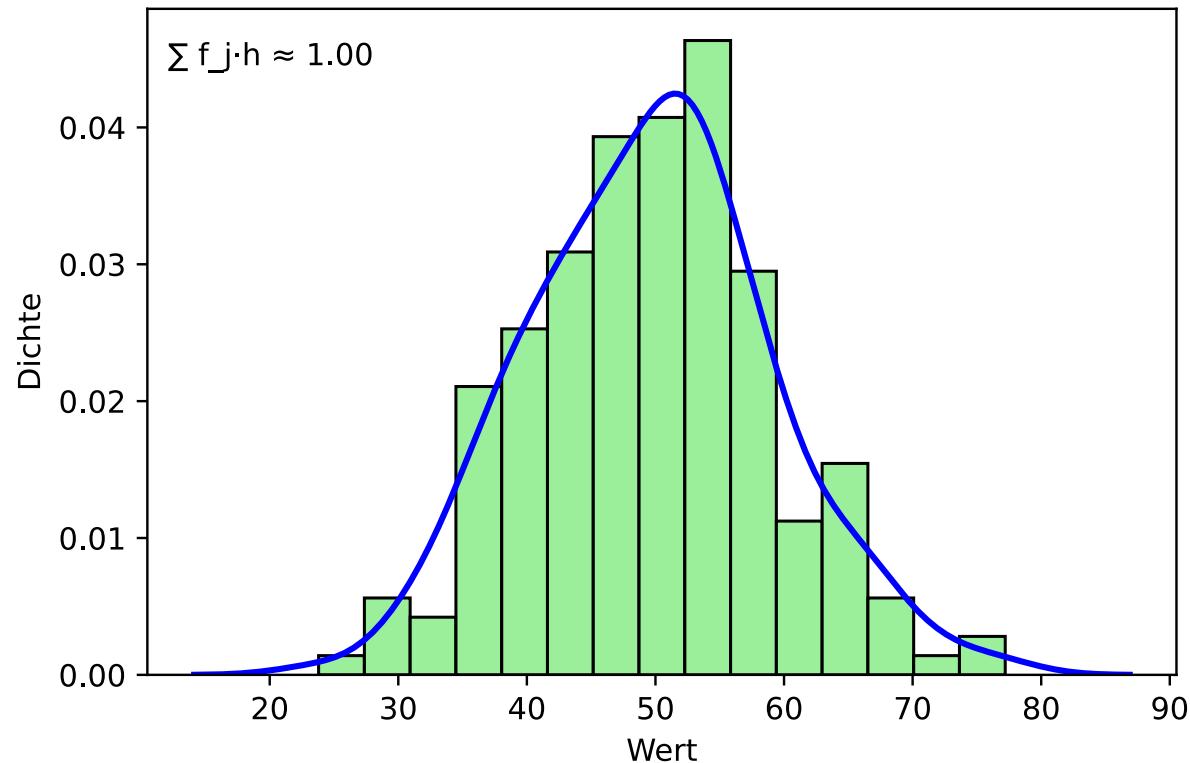
Klassen mit fester Breite zeigen Verteilung.

- Darstellung: Häufigkeit (Counts) oder Dichte
- Dichtehöhe: $f_j = \frac{c_j}{n \cdot h}$
- Fläche der Dichte = 1
- Erkennt Form, Lage und Moden
- **Mini-Check:** Warum ist Fläche 1?
- c_j = Anzahl Werte im Intervall
- n = Gesamtzahl der Werte
- h = Breite des Intervalls

Histogramm mit Counts



Histogramm mit Dichte (Fläche = 1)

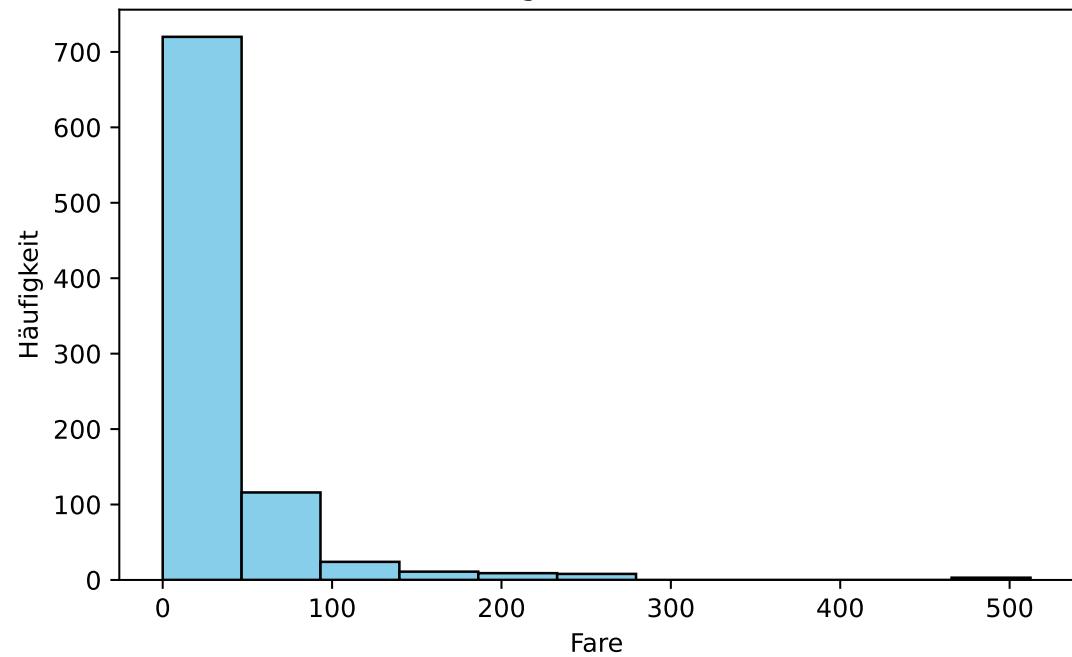


Binwahl – Regeln & Formeln

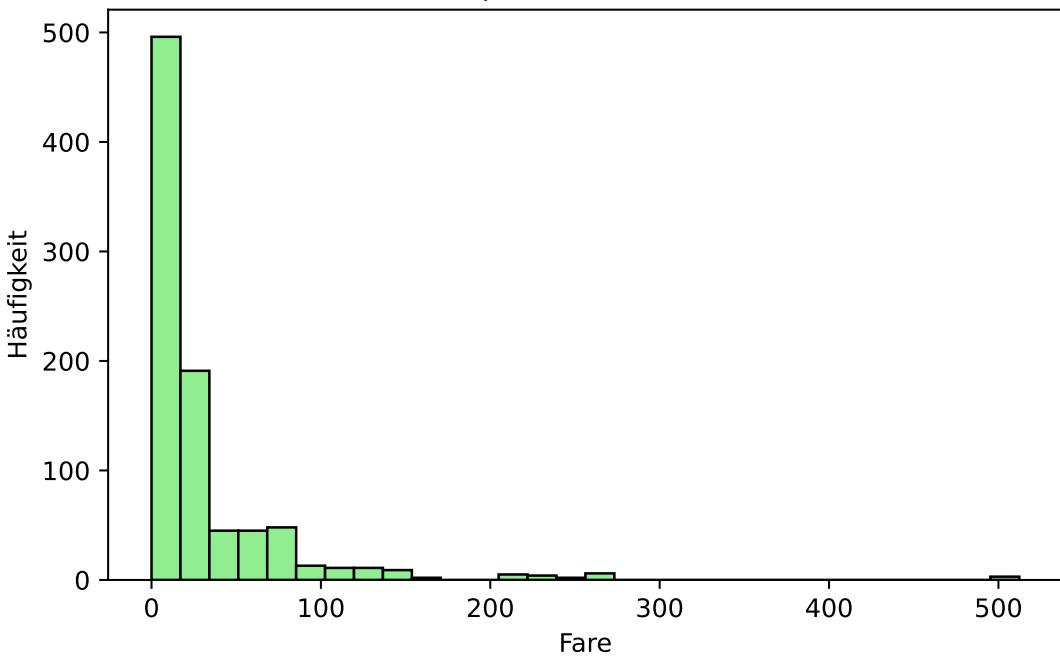
Bins steuern Aussagekraft.

- Faustregeln:
 - **Sturges:** $B = \lceil \log_2(n) + 1 \rceil \leftarrow$ normalverteilungsnah, eher konservativ.
 - $\sqrt{n} \leftarrow$ Daumenregel, schnell & simpel.
 - **Freedman–Diaconis (FD):** $h = 2 \cdot IQR \cdot n^{-1/3} \leftarrow$ theoretisch fundiert
- Praxis: FD am robustesten
- **Mini-Check:** Welche Regel bei Schiefe?

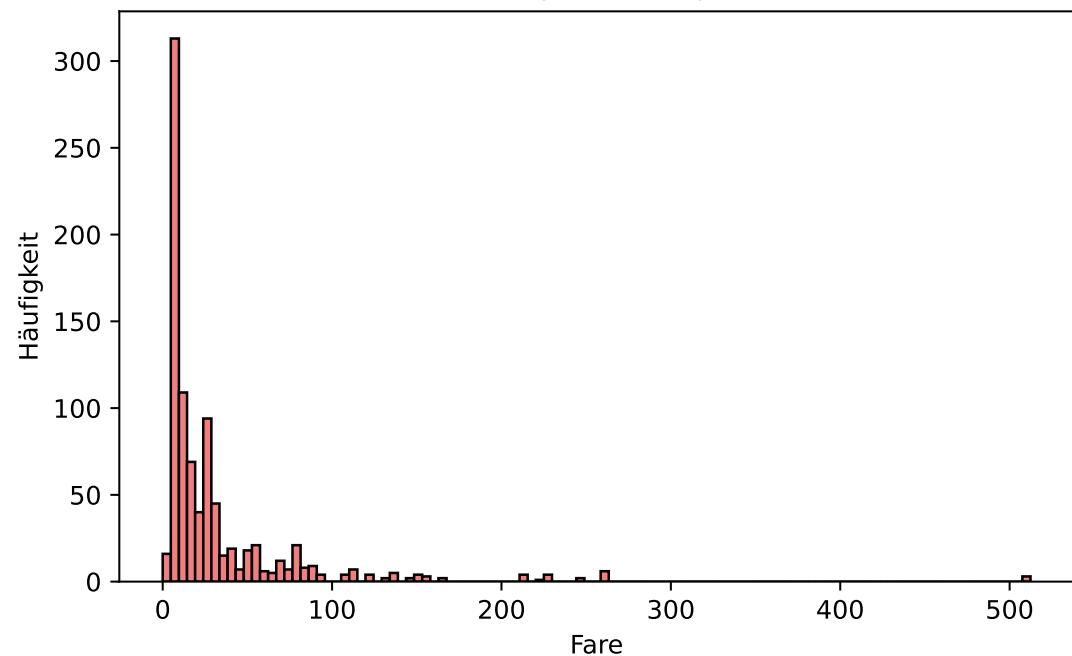
Sturges (Bins=11)



Sqrt(n) (Bins=30)



FD (Bins=107)



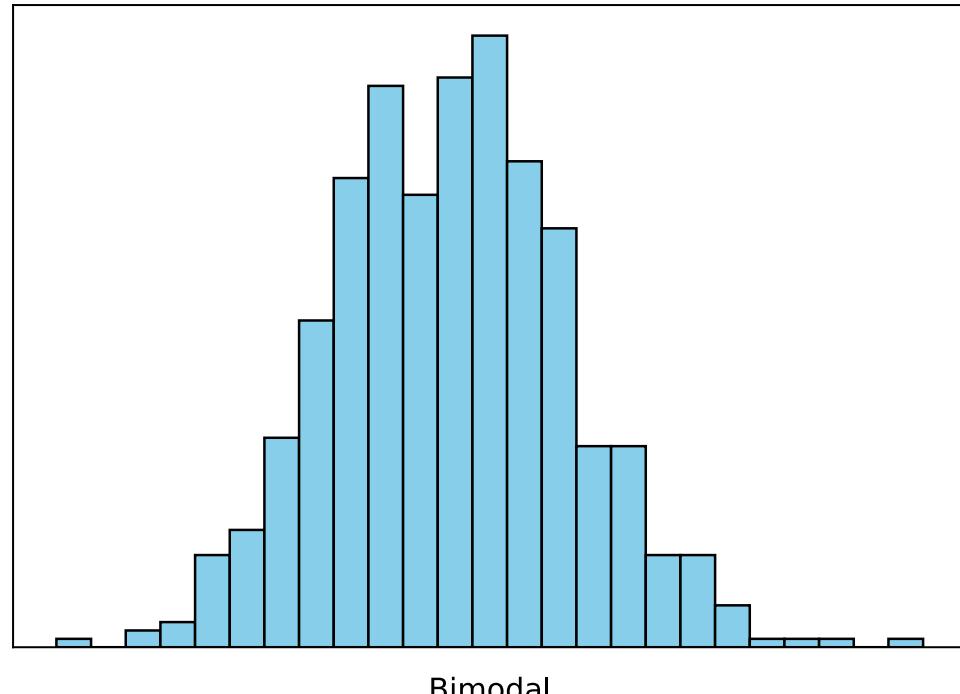
Histogramme lesen

Interpretation geht über Zählen hinaus.

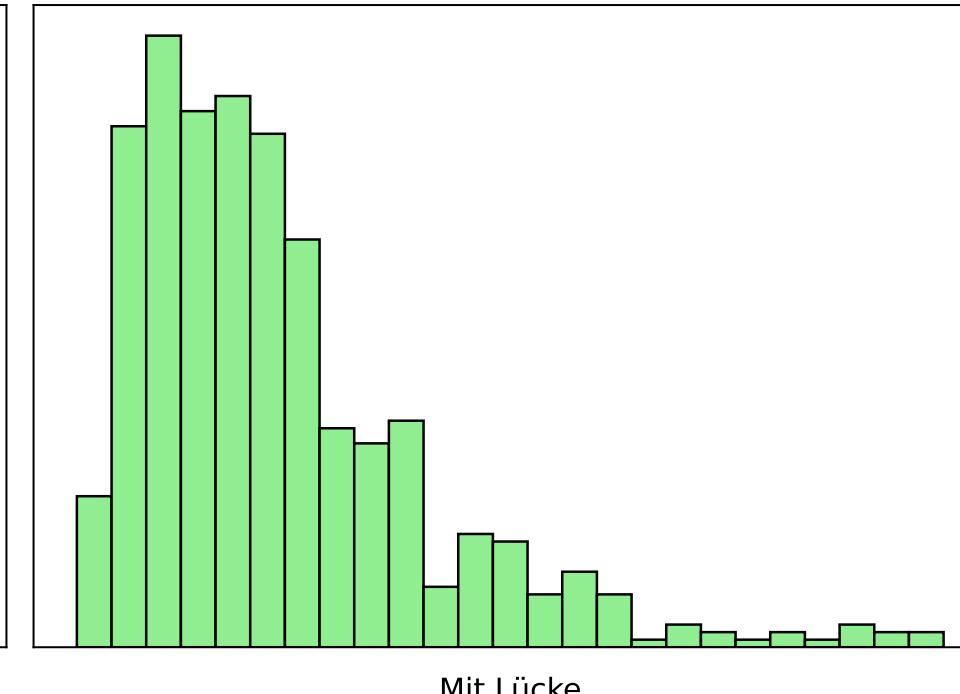
- **Schiefe erkennen**
- **Moden zählen**
- **Lücken deuten**
- **Tails prüfen**

Mini-Check: Was bedeutet Lücke in der Mitte?

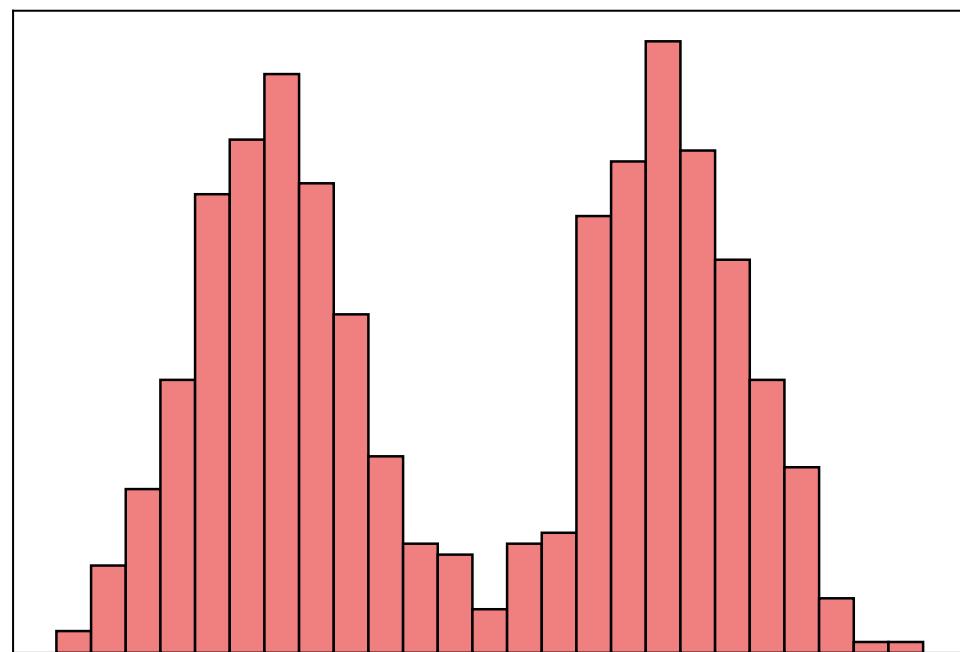
Symmetrisch



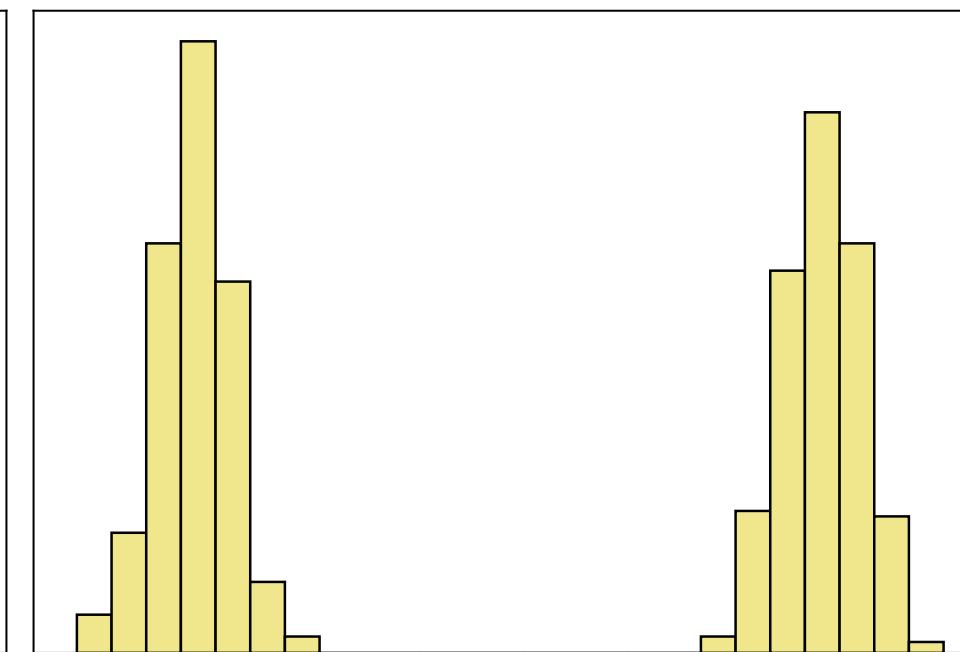
Rechts-schief



Bimodal



Mit Lücke



KDE (Kerndichteschätzung) – Idee der Glättung

Bandbreite ersetzt Bins

- Schätzung: $\hat{f}_h(x) = \frac{1}{nh} \sum K\left(\frac{x-x_i}{h}\right)$
- Üblich: Gauß-Kern
- Parameter h = Bandbreite steuert Glättung
- Bin-frei, aber parameterabhängig
- **Mini-Check:** Warum nicht parameterfrei?

n : Anzahl Daten

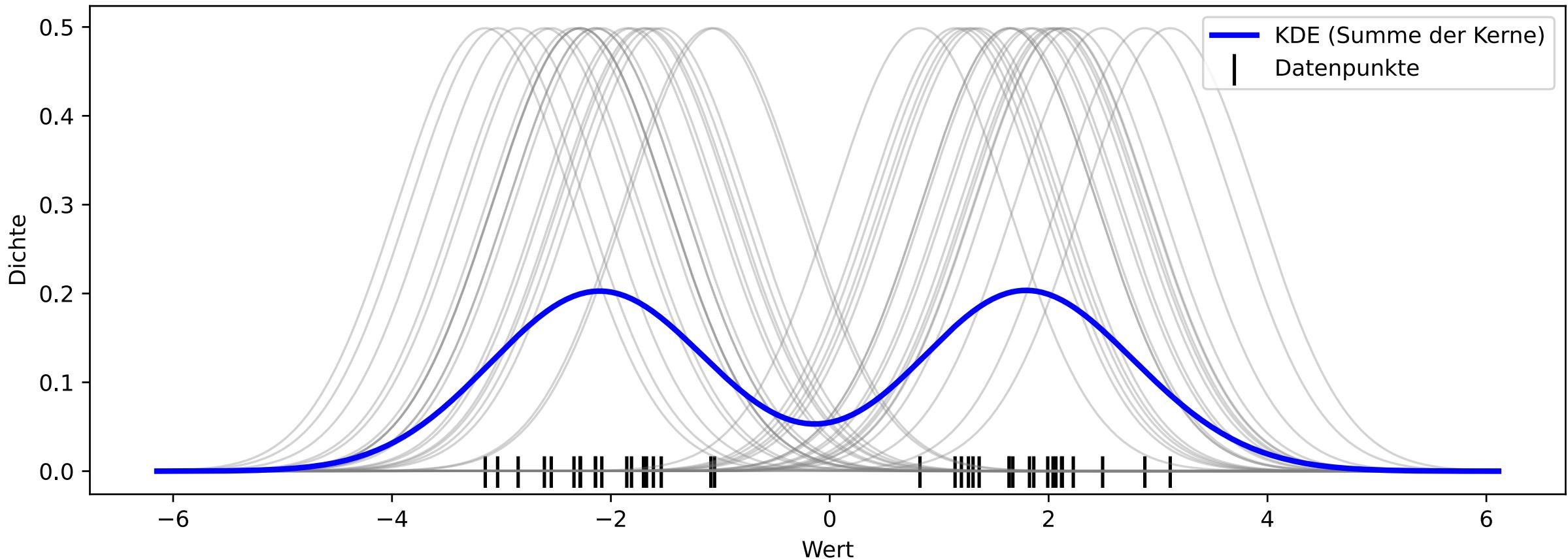
h : Bandbreite (Glättung)

x_i : i-ter Datenpunkt

K : Kern (z. B. Gauß-Glocke)

$\frac{x-x_i}{h}$: Distanz zu x_i in h -Einheiten

KDE als Summe von Gauss-Kernen (Bandbreite $h=0.8$)

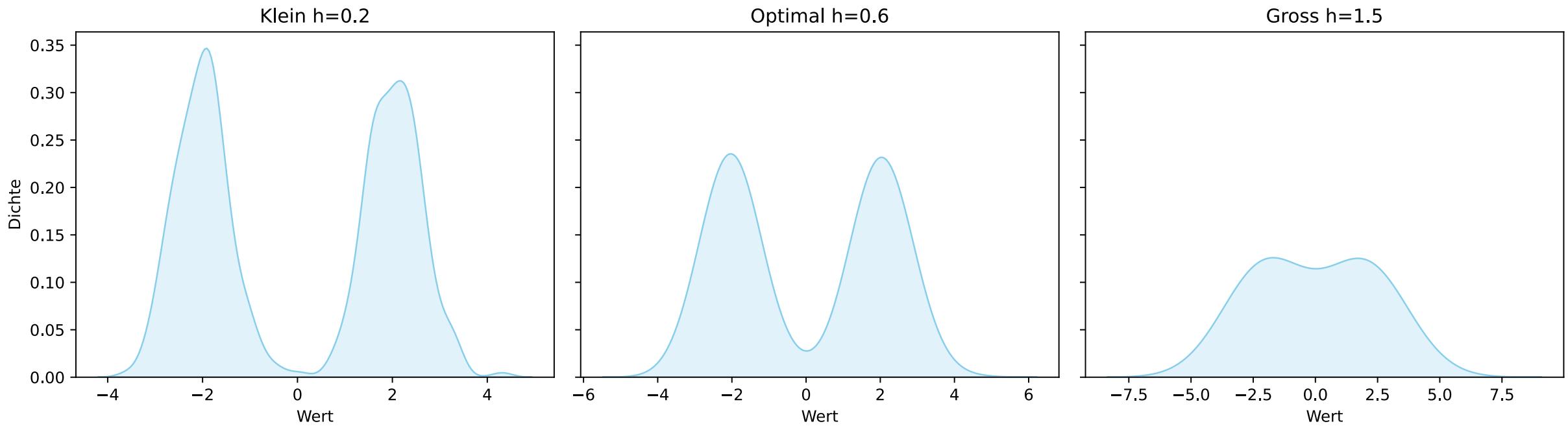


Bandbreite wählen

Unter vs Überglättung unterscheiden.

- Kleine $h \rightarrow$ viele Details, evtl. Rauschen
- Große $h \rightarrow$ geglättet, Details verschwinden
- Heuristiken:
 - **Scott:** $h = \sigma n^{-1/5} \sqrt{n} \leftarrow$ normalverteilten Daten
 - **Silverman:** $h = 0.9 \cdot \min(\sigma, IQR/1.34) n^{-1/5}$ robuster bei schießen Daten

Mini-Check: Wann Silverman statt Scott?-



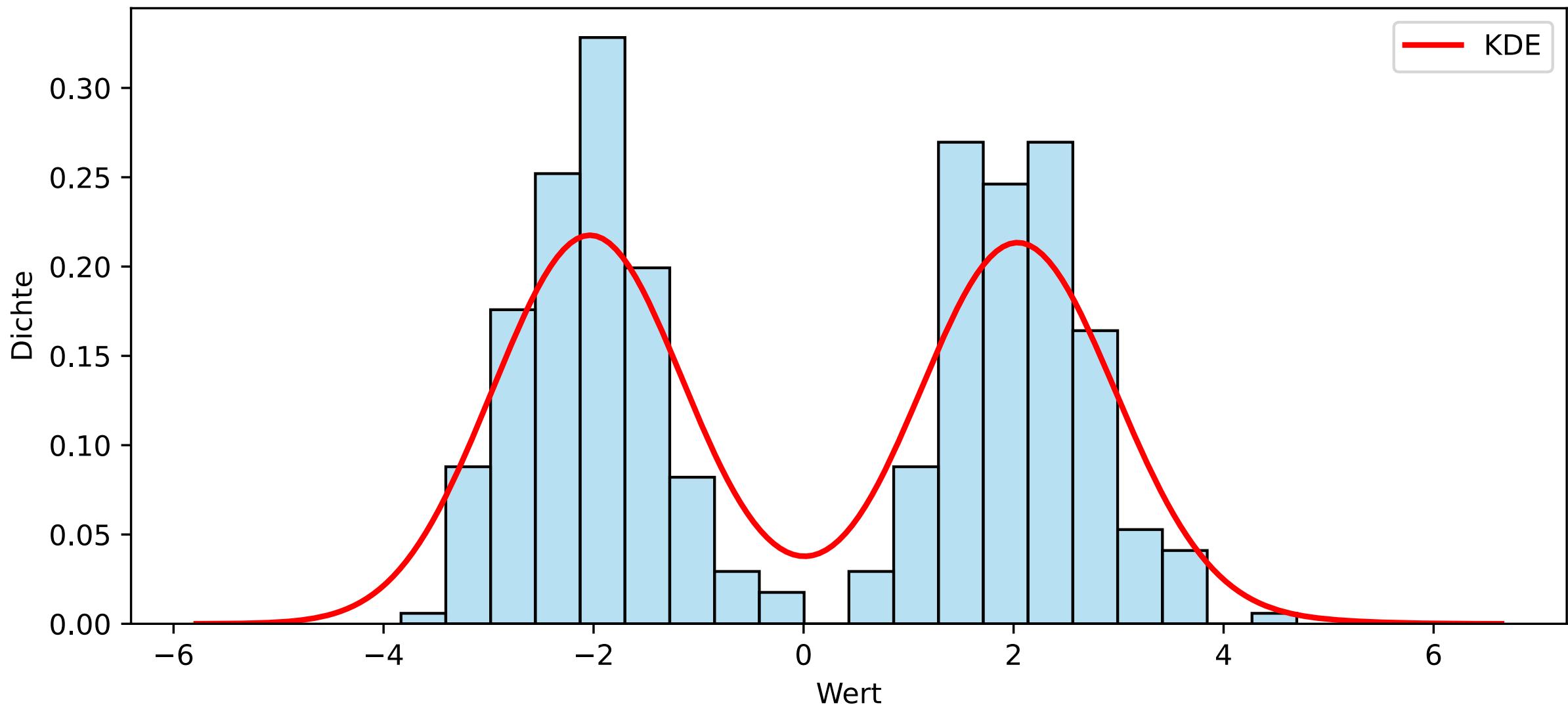
Histogramm oder KDE?

Kombination ist oft ideal

Methode	Eigenschaften
Histogramm	<ul style="list-style-type: none">• Intuitiv, zeigt Häufigkeiten• Stabil bei kleinem n
KDE	<ul style="list-style-type: none">• Glatte Form, Moden erkennbar• Parameterabhängig h

- Beste Praxis: Overlay von Histogramm + KDE
- **Mini-Check:** Wann ist nur KDE irreführend?

Histogramm + KDE Overlay



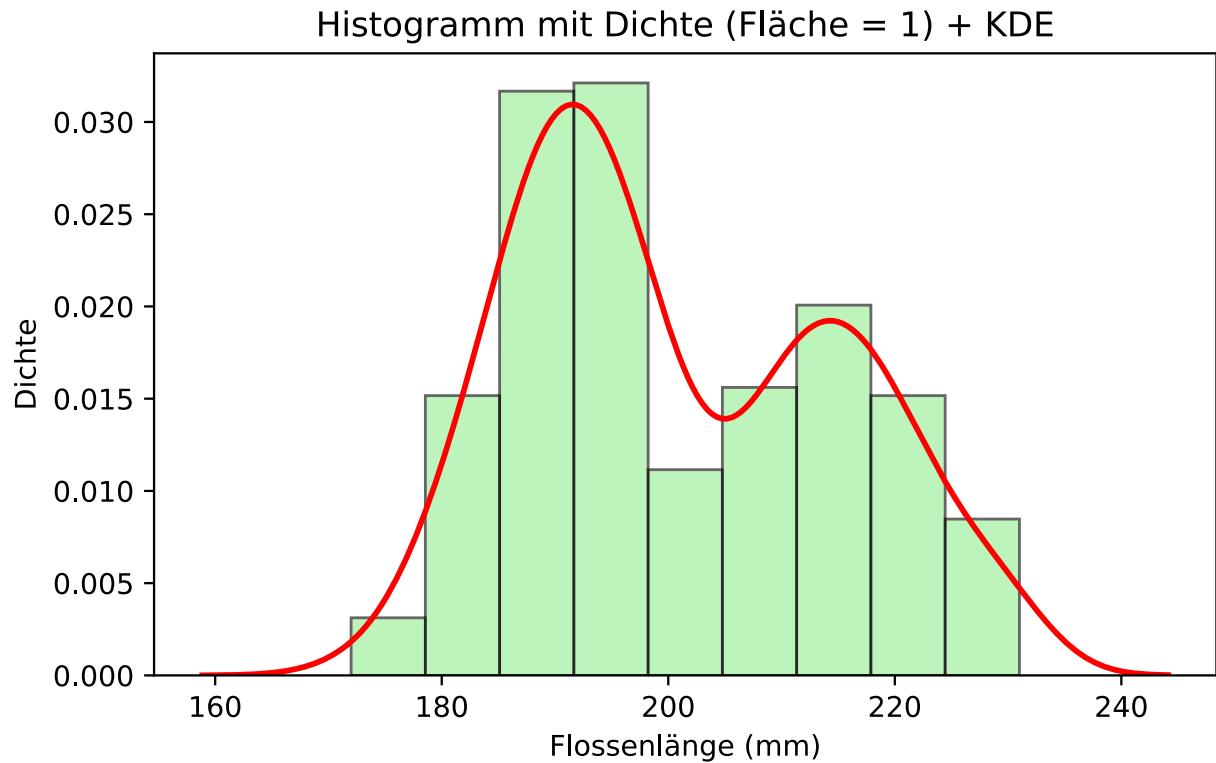
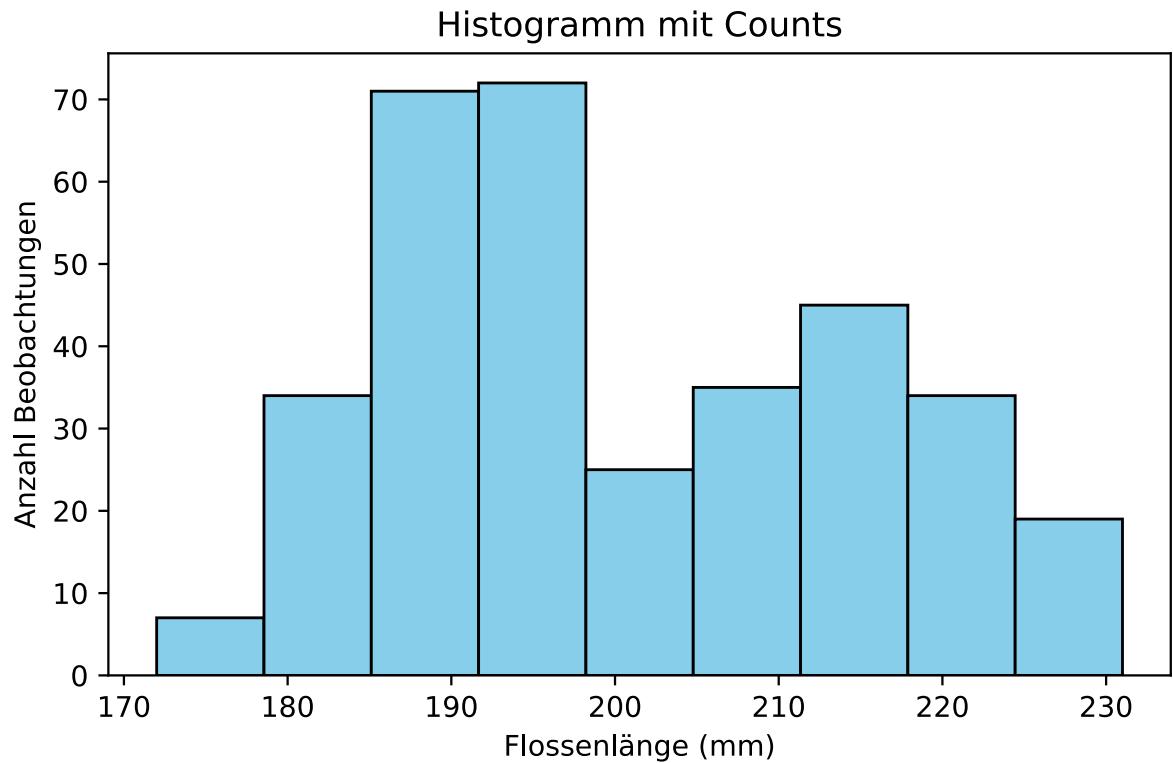
Python Histogramm und KDE

Saubere Defaults genügen.

```
plt.hist(x, bins="fd", density=True, edgecolor="black", alpha=0.4)  
sns.kdeplot(x=x, bw_adjust=1.0)
```

- `density=True` → normiert Fläche = 1
- `bw_adjust` variieren → Bandbreite anpassen

Mini-Check: Warum `density=True` ?



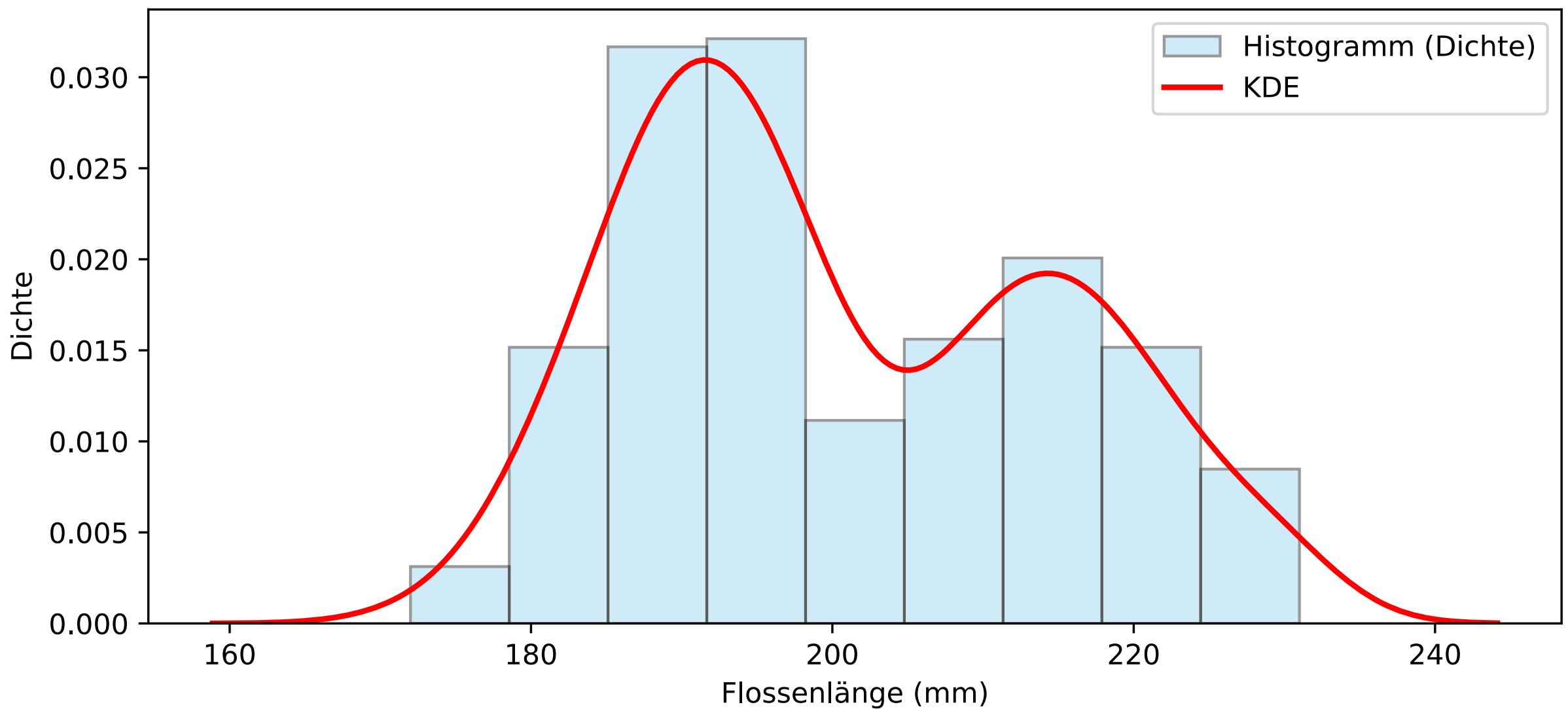
Beispiel Penguins flipper_length

Overlay zeigt Moden und Lage.

```
df = sns.load_dataset("penguins").dropna(subset=["flipper_length_mm"])
x = df["flipper_length_mm"].to_numpy()
plt.hist(x, bins="fd", density=True, alpha=0.4); sns.kdeplot(x=x)
```

- Lage, Streuung, Moden deuten
- **Mini-Check:** Erkennst du Multimodalität?

Penguins – Flossenlänge (Overlay Histogramm + KDE)



Häufige Fehler: Histogramm & KDE

Setup-Fehler vermeiden.

- Counts und Dichte verwechseln
- Zu wenige Bins → Muster verdeckt
- Falsche Achsenbeschriftung („Counts“ oder „Dichte“), Log ohne Hinweis...
- Falsche Bandbreite

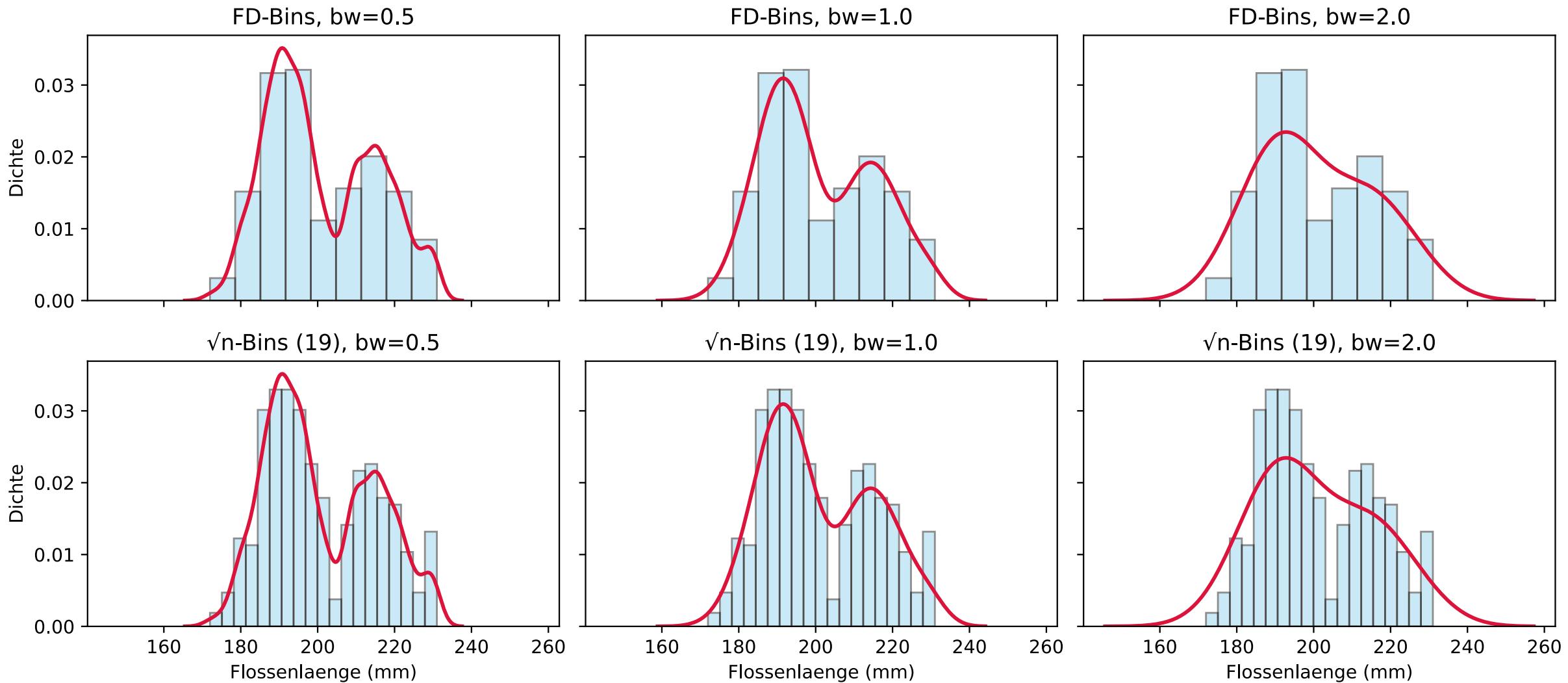
Mini-Check: Wie zeigst du Überglättung schnell?

Mini-Aufgabe: Binzahl & Bandbreite

Regel anwenden und verteidigen.

- Wähle Bins mit FD-Regel vs. \sqrt{n}
- Teste `bw_adjust` in {0.5, 1.0, 2.0}
- Formuliere Entscheidung in 1 Satz
- Kurz begründen
- **Mini-Check:** Warum ändert h die Schlussfolgerung

Penguins – Einfluss von Binzahl (FD vs. \sqrt{n}) und Bandbreite (bw_adjust)



Reporting

Schreiben Sie nach der Analyse **einen Satz** zur Entscheidung:

- Was zeigt die Struktur am besten?
- Ohne zu rauschen, und **warum?**

Musterbeispiel:

„FD-Bins plus KDE mit `bw_adjust=1.0` zeigen zwei stabile Gipfel und einen klaren Schwerpunkt um 195 mm.

Die Aussage bleibt robust gegenüber `bw_adjust` 0.5 und 2.0.“



Kurz, prägnant, reproduzierbar.

Key Takeaways – Histogramm & KDE

- **Histogramm:** zeigt Form, Lage & Moden; Fläche = 1 bei Dichte
- **Binwahl:** Regeln (Sturges, \sqrt{n} , FD); FD ist robustester Default
- **Interpretation:** Schiefe, Moden, Lücken & Tails beachten
- **KDE:** glatte Alternative zum Histogramm, abhängig von Bandbreite h
- **Bandbreite:** zu klein = Rauschen, zu gross = Details verschwinden
- **Praxis:** Overlay Histogramm + KDE liefert das beste Bild



Immer Parameter variieren und Ergebnisse kritisch prüfen.

5. Zusammenfassung & Ausblick

Bereich	Kernaussagen
Lage & Streuung	<p>Median + IQR/MAD = robust</p> <p>Mean + SD = ergänzend</p> <p>Immer zuerst robuste Masse prüfen</p>
Visualisierung	<p>Histogramm + KDE → Form sichtbar</p> <p>Bin-Wahl und Bandbreite kritisch prüfen</p>
Diagnostik	<p>Outlier: Tukey (IQR), mod. Z (Median+MAD)</p> <p>Klass. Z nur bei Normalität</p> <p>Heavy Tails sind nicht immer Fehler</p>
Reporting	<p>Klare Sätze mit Einheit</p> <p>Entscheidungen dokumentieren</p> <p>Robustheit betonen</p>

Nächster Schritt: Online-Vertiefung

Zweiten Teil als **Online-Video**.

Titel: **Verteilungsdiagnostik und Gruppenvergleiche: Fortgeschrittene Visualisierungen**

Im Video behandeln wir:

- **Boxplots und Violinplots** zum schnellen **Gruppenvergleich**
- **ECDF und QQ-Plots** zur präzisen **Verteilungsdiagnostik**

Ausblick: Nächste Woche

Untersuchung von **Zusammenhängen** zwischen zwei Variablen.

- **Masse:** Kovarianz, Pearsons r , Spearman's ρ und Kendalls τ
- **Wichtige Fallen:** Das **Simpson-Paradox** und die Unterscheidung: **Korrelation** \neq **Kausalität**

Vorbereitung:

- Practical Statistics for Data Scientists (*PSDS*) Kapitel 2
- Statistics for Data Scientists (*SDS*) Kapitel 3

Kahoot!