Angewandte Statistik

Julian Hinz — Universität Bielefeld

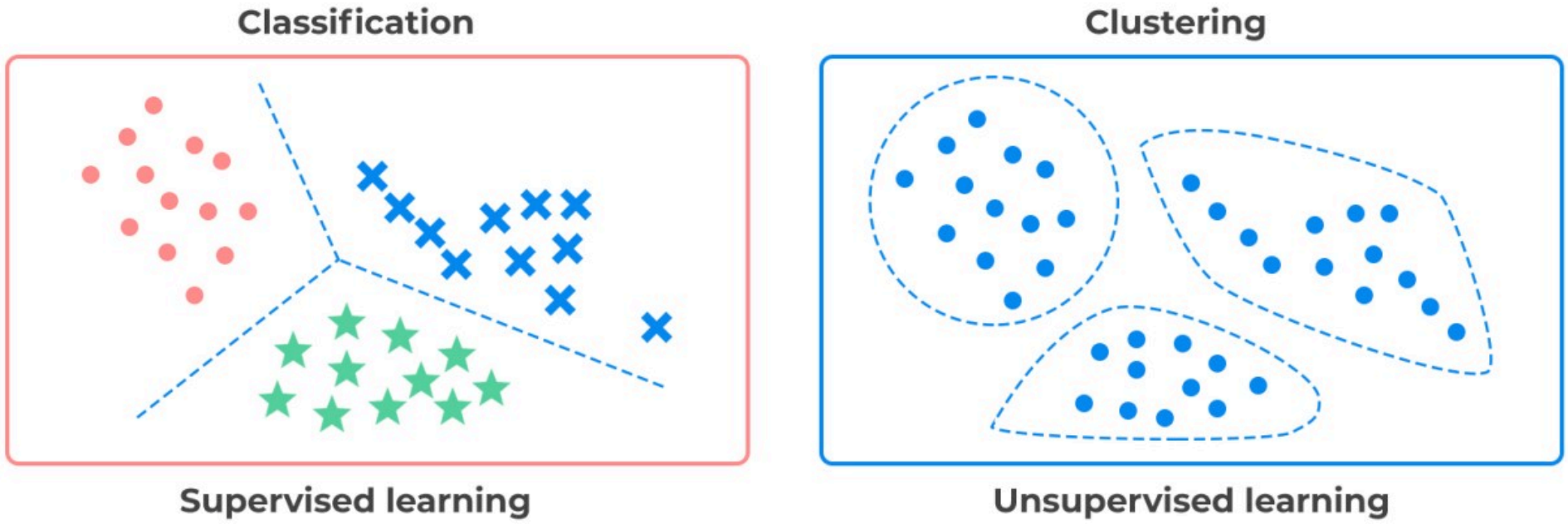
Session 9

Dimensionsreduktion und Clustering — Unsupervised Machine Learning

Lernziele

- Konzept und Anwendung von Dimensionsreduktion und Clustering
- Anwendung der Methoden in R

Supervised vs. unsupervised learning

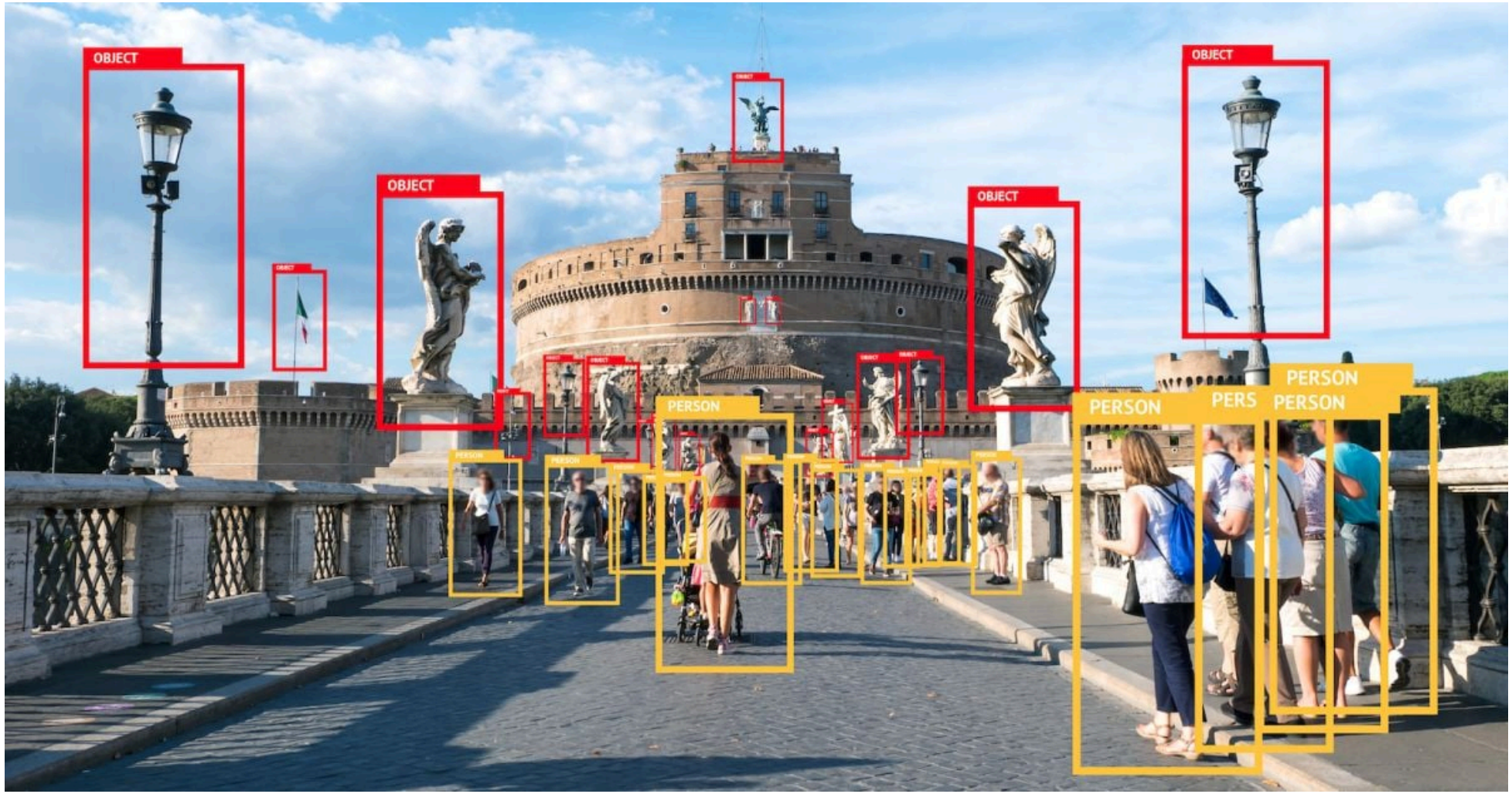


Übersicht

- Was ist unüberwachtes Lernen?
- Typen des unüberwachten Lernens
- Hauptkomponentenanalyse (PCA)
- K-Means Clustering

Was ist unüberwachtes Lernen?

- Unüberwachtes Lernen bezieht sich auf Methoden, bei denen Algorithmen verwendet werden, um Muster und Strukturen in Datensätzen zu erkennen, die keine gelabelten Ausgaben enthalten
- Ziel: Entdeckung von Gruppen, Mustern und Strukturen in den Daten ohne vorgegebene Kategorien


OBJECT
OBJECT
OBJECT
OBJECT
PERSON
PERSON
PERS
PERSON

Anwendungsbeispiele

- Kundensegmentierung im Marketing
- Entdeckung neuer Krankheitsmuster in der Medizin
- Anomalieerkennung in Netzwerksicherheit

Typen des unüberwachten Lernens

Dimensionsreduktion

- Ziel: Reduktion der Anzahl der Variablen unter Beibehaltung der wesentlichen Informationen
- Beispiele: Hauptkomponentenanalyse (PCA), t-SNE, UMAP

Clustering

- Ziel: Gruppierung von Datenpunkten, die ähnliche Eigenschaften teilen
- Beispiele: K-Means, Hierarchisches Clustering

Andere Typen

- Assoziationsregeln
- Regularisierung

Hauptkomponentenanalyse (PCA)

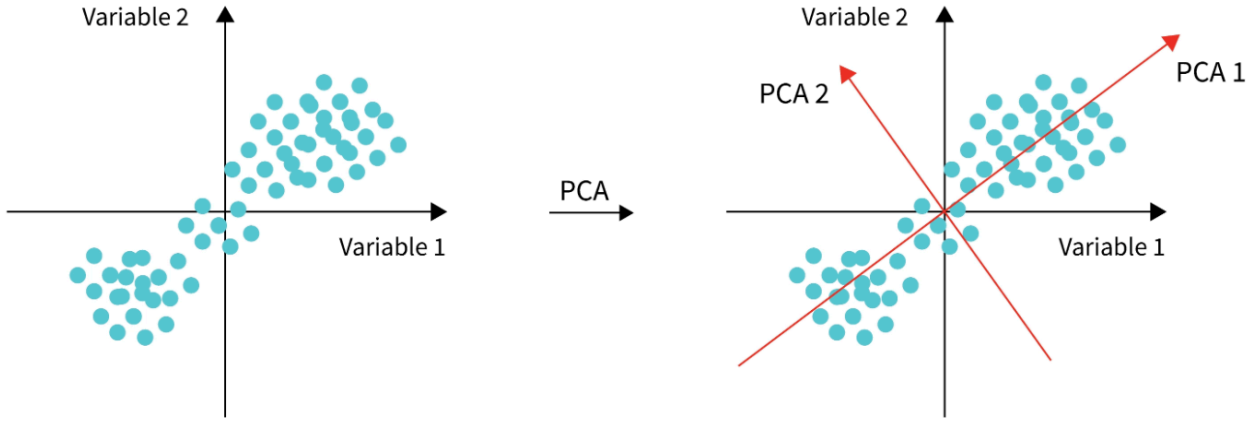
Hauptkomponentenanalyse (PCA)

- PCA ist eine Form unüberwachten Lernens zur Reduzierung der Dimensionalität von Daten
- Ziel: Finden von hilfreichen Mustern in den Daten
- Anwendung: Reduzierung der Dimensionalität, Umgang mit Multikollinearität, Visualisierung von Modellergebnissen und explorative Datenanalyse

Definition

- PCA reduziert Anzahl der Eingabevariablen, während so viel wie möglich der Variation in den Daten erhalten bleibt
- ursprüngliche Daten werden in neue Variablen (Hauptkomponenten) transformiert, die unkorreliert sind

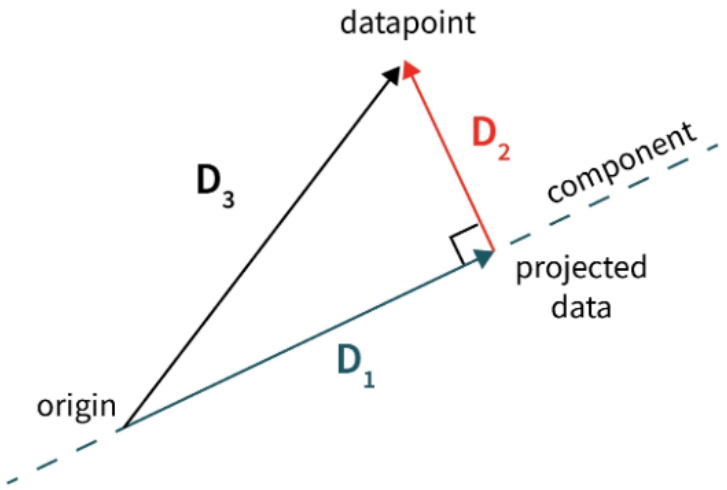
Prinzip



Quelle: scaler.com

- PCA findet Linien (2D), Ebenen (3D) oder Hyper-Ebenen (>3 Dimensionen), die untereinander orthogonal sind
- Diese Hauptkomponenten erfassen die maximale Varianz in den Daten

Idee



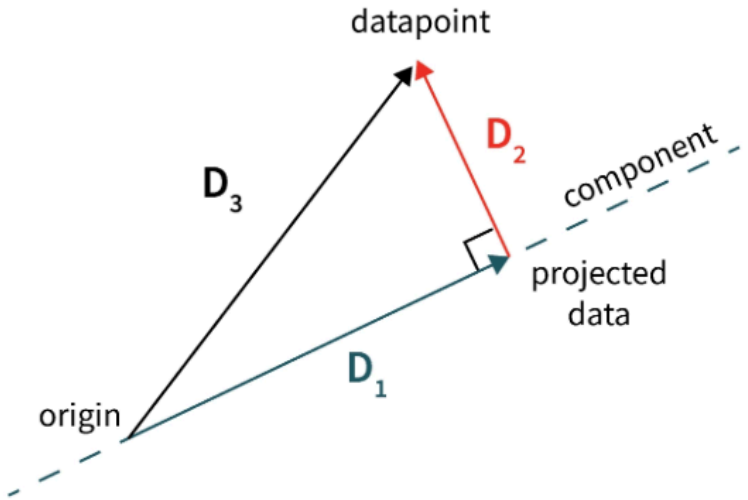
$$D_3^2 = D_1^2 + D_2^2$$

initial variance = remaining variance + lost variance

Quelle: scaler.com

- Minimierung der mittleren senkrechten Distanz zur Hauptkomponentenlinie
- Funktion erreicht Minimum, wenn der Eigenvektor der Kovarianzmatrix gleich dem Richtungsvektor der Hauptkomponente ist

Mathematische Formulierung



$$D_3^2 = D_1^2 + D_2^2$$

initial variance = remaining variance + lost variance

Quelle: scaler.com

$$\min\left(\frac{1}{n}\sum_{i}^{n}\left(x_i^T x_i - (u_1^T x_i)^2\right)\right)$$

Eigenschaften der Hauptkomponenten

- Hauptkomponenten sind lineare Kombinationen der ursprünglichen Merkmale
- Hauptkomponenten sind orthogonal, d.h., die Korrelation zwischen zwei PCs = 0
- “Wichtigkeit” jeder weiteren Hauptkomponente nimmt ab (gemessen an Eigenwerten)
- Anzahl der PC $\leq$ Anzahl an Variablen im Datensatz
  - Typischerweise decken wenige Komponenten mehr als 90% der Varianz ab

Normalisierung der Merkmale

- Normalisierung (oder Standardisierung) u.U. wichtiger Vorverarbeitungsschritt bei PCA
- Idee: Reskalierung der Merkmale, sodass sie Mittelwert = 0 und Standardabweichung = 1 haben

Warum Normalisierung?

- Ohne Normalisierung variieren Komponenten unterschiedlich aufgrund ihrer jeweiligen Skalen
  - z.B. Körpergröße vs. Gewicht in Meter vs. Kilo
- PCA könnte sagen, dass maximale Varianz hauptsächlich mit Merkmal “Gewicht“ korrespondiert

Umgang mit verschiedenen Datentypen

- Kontinuierliche Variablen normalisieren: mit Mittelwert zentrieren und durch Standardabweichung teilen
- Ordinale/kategorische Daten: Mischen mit numerischen Variablen aufgrund der Berechnung von Korrelation/Kovarianz und Standardisierung schwierig
- Besser: Kategorische Versionen von PCA oder nicht-lineare PCA für kategoriale/ordinale Daten verwenden

Beispiel in R

```r
# Daten laden und skalieren
data(iris)
iris_scaled <- scale(iris[, -5])

# PCA durchführen
pca <- prcomp(iris_scaled)

# Zusammenfassung der PCA
summary(pca)

# PCA-Plot
library(ggplot2)
pca_data <- as.data.frame(pca$x)
pca_data$Species <- iris$Species
ggplot(pca_data, aes(x = PC1, y = PC2, color = Species)) +
  geom_point() +
  labs(title = "PCA der Iris-Daten", x = "Hauptkomponente 1", y = "Hauptkomponente 2")
```

Zusammenfassung PCA

- Technik zur Reduzierung der Dimensionalität
- kann zur Visualisierung von Datensätzen oder Erstellung neuer kompakter Merkmale verwendet werden
- Hauptkomponenten sind unkorreliert und orthogonal zueinander
- Auswahl der Hauptkomponenten anhand eines Schwellenwerts für den Prozentsatz der erklärten Varianz
- Möglichkeit der Rücktransformation der Eingabedaten nach der PCA

K-Means Clustering

Grundlagen

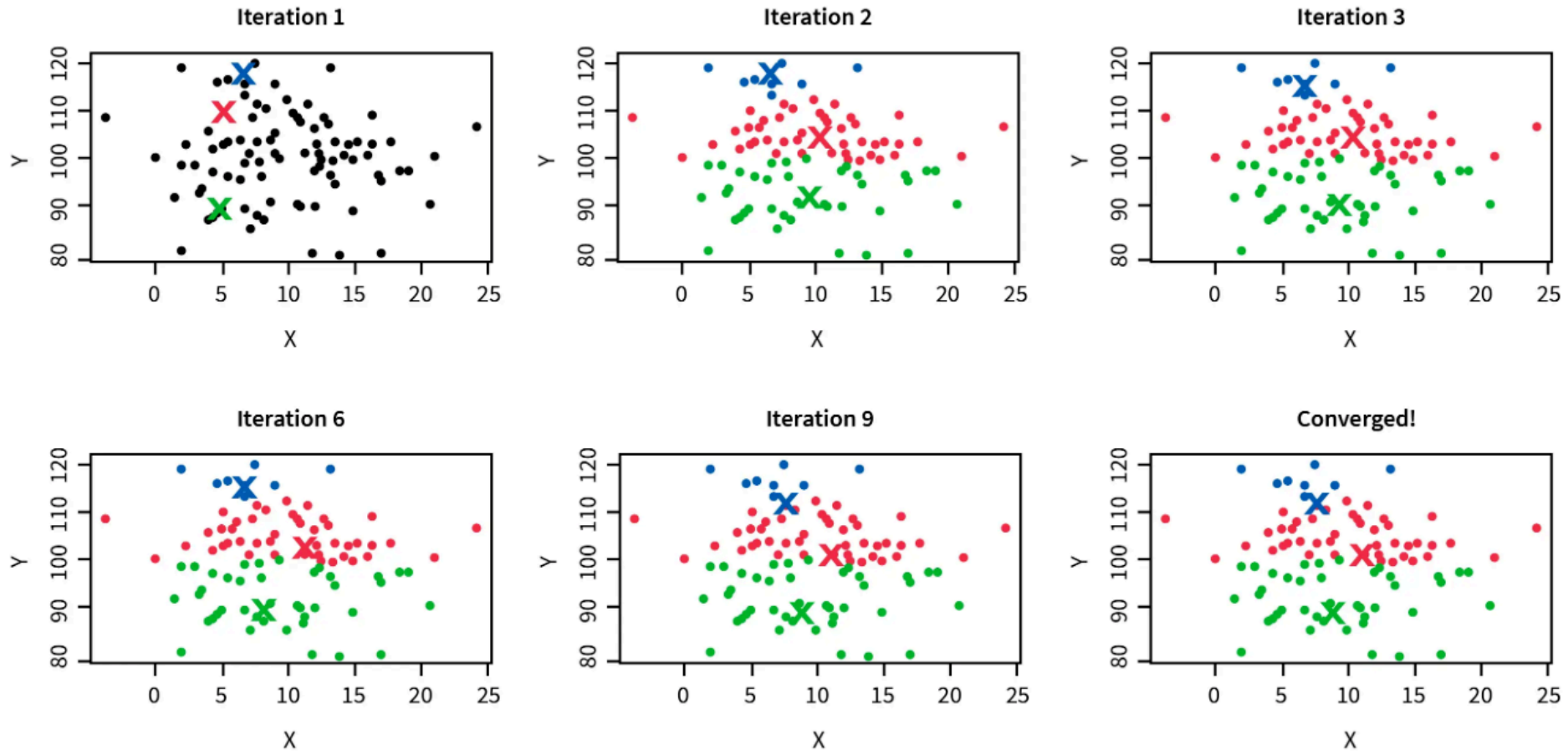
- K-Means Clustering ist eine weit verbreitete und effektive Methode zur Partitionierung eines Datensatzes in eine vorbestimmte Anzahl von Clustern
- Ziel: Gruppierung von Datenpunkten basierend auf ihrer Ähnlichkeit zum Zentrum jedes Clusters

Anwendungsbereiche

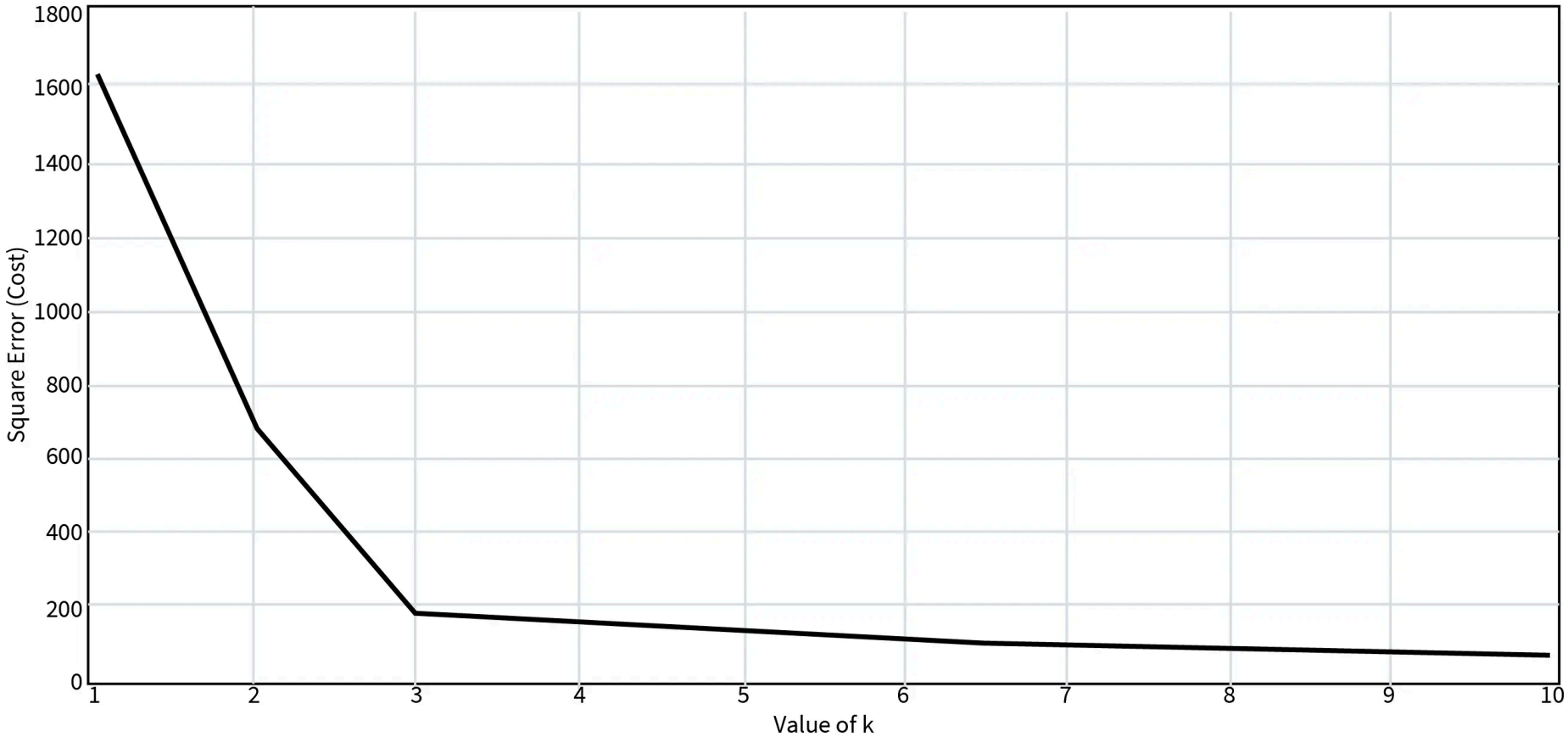
- Kundenklassifizierung im Marketing
- Mustererkennung
- Bildsegmentierung

K-Means Clustering Algorithmus

1. Wähle die Anzahl der Cluster K
2. Initialisiere K Cluster-Zentren zufällig (oder mit einer spezifischen Initialisierungsmethode)
3. Weise jeden Datenpunkt dem nächstgelegenen Zentrum zu
4. Berechne die Zentren der Cluster als Mittelwert der zugewiesenen Datenpunkte (Aktualisierungsschritt)
5. Wiederhole die Schritte 3 und 4 bis zur Konvergenz



Quelle: scaler.com



Quelle: scaler.com

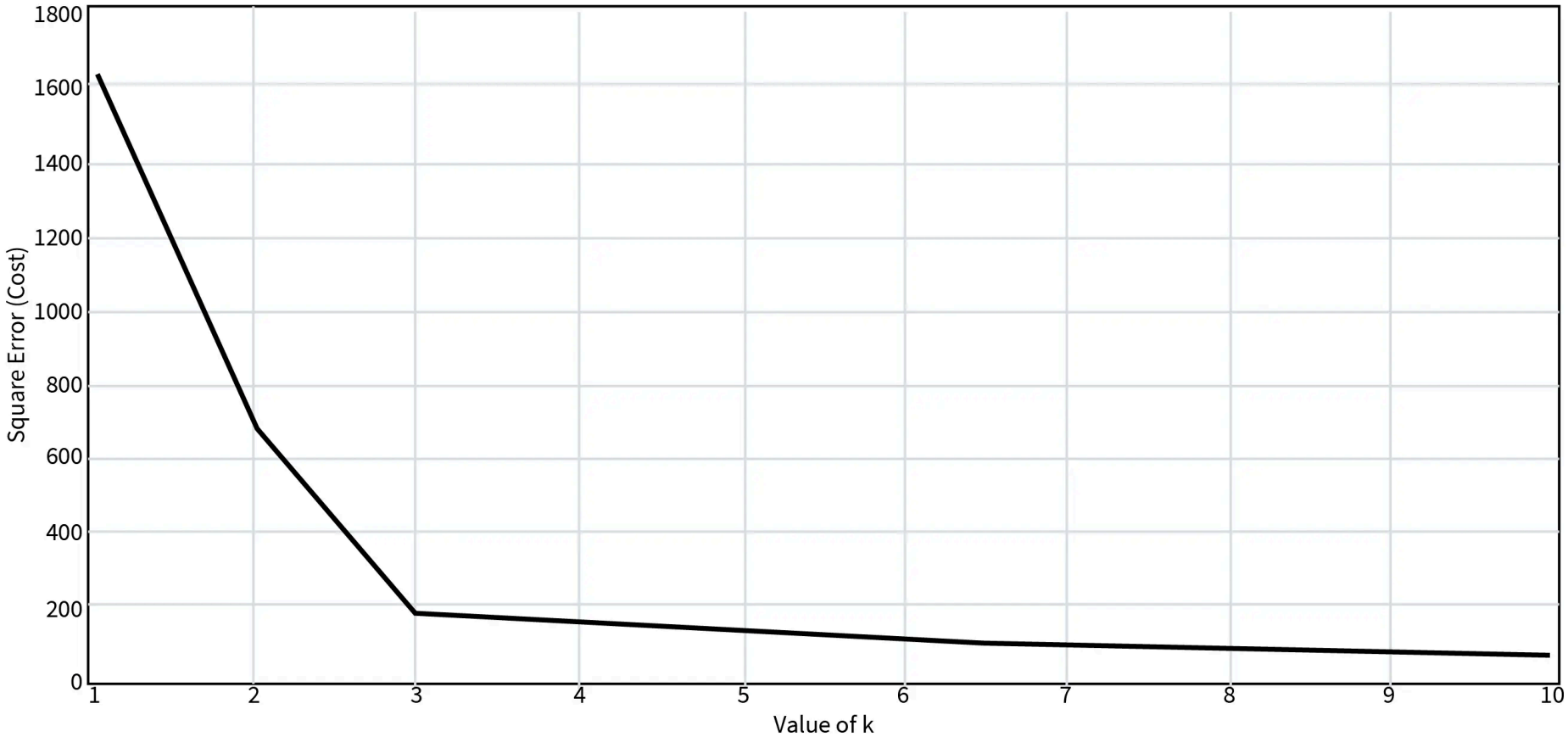
Konvergenz

- Algorithmus konvergiert, wenn sich Cluster-Zuweisungen oder Zentren nicht mehr signifikant ändern
- kann auch nach einer vordefinierten Anzahl von Iterationen beendet werden

Beispiel in R

```r
# K-Means Clustering durchführen
set.seed(123)
kmeans_result <- kmeans(iris_scaled, centers = 3)

# K-Means Plot
kmeans_data <- as.data.frame(iris_scaled)
kmeans_data$Cluster <- factor(kmeans_result$cluster)
ggplot(kmeans_data, aes(x = PC1, y = PC2, color = Cluster)) +
  geom_point() +
  labs(title = "K-Means Clustering der Iris-Daten", x = "Hauptkomponente 1", y = "Hauptkomponente 2")
```



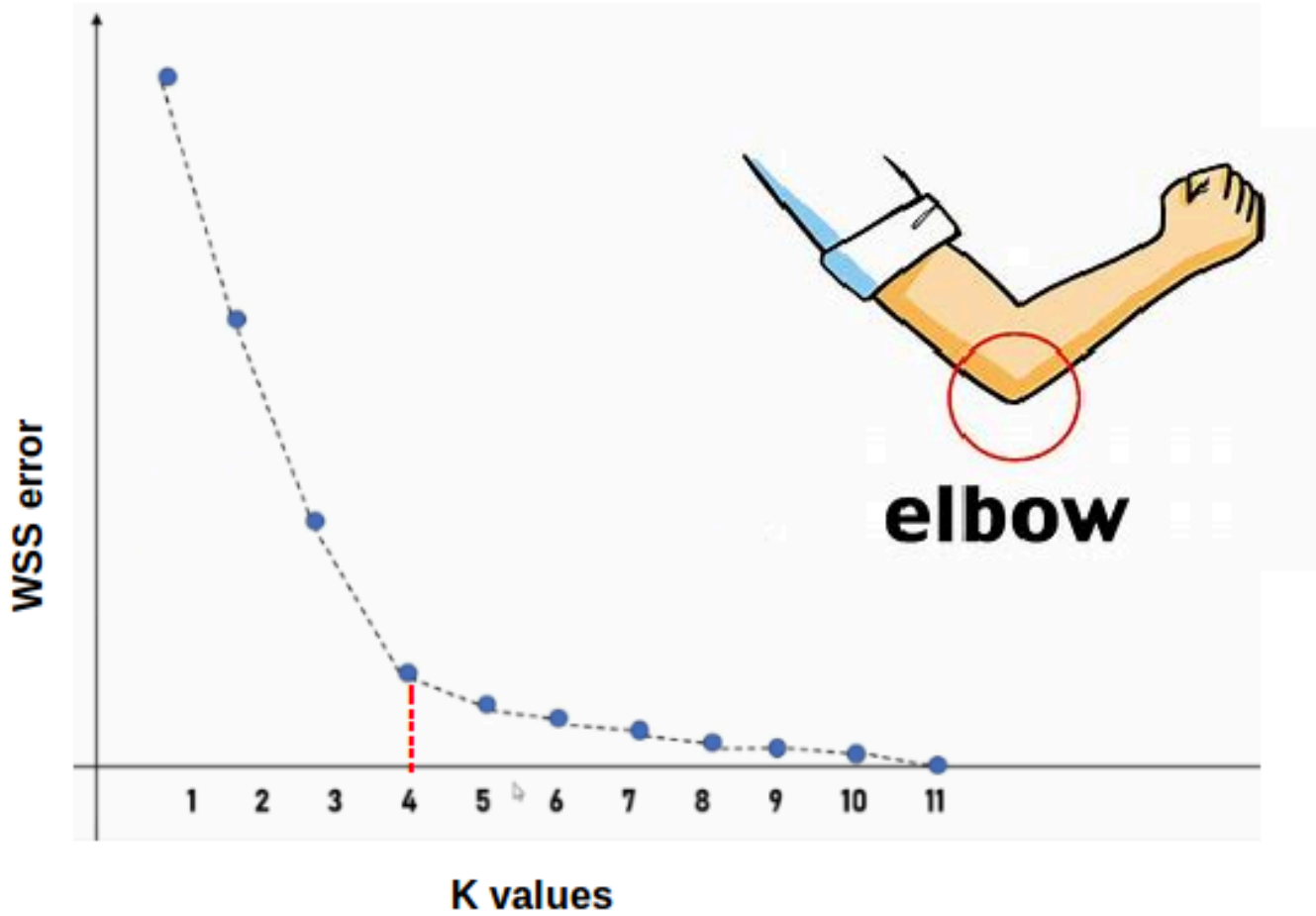
Quelle: scaler.com

Elbow Method

Idee & Umsetzung

- **Ziel:** Finde Punkt, an dem zusätzliche Cluster nur noch geringe SSE-Verbesserung bringen
- Schritte
  1. K = 1 … K_{max} durchlaufen
  2. Sum of Squared Errors (SSE) bzw. *Within-Cluster Sum of Squares* berechnen
  3. „Knick“ im Scree-Plot identifizieren
- **Heuristik:** Optimum ≈ erster deutlicher Krümmungswechsel

Elbow method



Cluster-Validierung

Metrik	Optimum	Idee
Silhouette	↑	Dichte vs. Trennung
Davies-Bouldin	↓	Verhältnis Intra-/Inter-Distanzen
Calinski-Harabasz	↑	Varianz zwischen / innerhalb Cluster
Dunn-Index	↑	Kleinster Inter-Cluster Abstand relativ zu größter Intra-Cluster-Distanz
Gap Statistic	↑	SSE-Lücke zu Referenz (Bootstrapping)

> **Praxis-Tipp:** Kombination mehrerer Metriken

Zusammenfassung K-Means Clustering

- K-Means Clustering ist eine leistungsstarke Methode zur Gruppierung ähnlicher Datenpunkte
- Wichtige Schritte: Auswahl der Clusteranzahl, Zuweisung der Datenpunkte, Aktualisierung der Cluster-Zentren, Iteration bis zur Konvergenz

Semi-Supervised Learning: Constrained Clustering

Must- / Cannot-Link

- **Idee:** Domain-Wissen als paarweise Constraints einfließen lassen
  - *Must-Link*: Punkt-Paar *muss* gleiches Cluster haben
  - *Cannot-Link*: Punkt-Paar *darf nicht* gleiches Cluster haben
- **Algorithmen**
  - **COP-K-Means** (Wagstaff 2001)
  - **Seeded K-Means / PCK-Means** (partial labelling)
  - **Spectral Learning with Constraints**
- **R-Pakete:** `conclust`, `clustConstraint`
- **Use-Cases:** Zusammengehörige Kunden, unterschiedliche Krankheits-Subtypen

Zusammenfassung

- Unüberwachtes Lernen ermöglicht die Entdeckung von Mustern und Strukturen in unmarkierten Daten
- Techniken wie PCA, K-Means und hierarchisches Clustering bieten leistungsstarke Werkzeuge zur Analyse und Visualisierung hochdimensionaler Daten