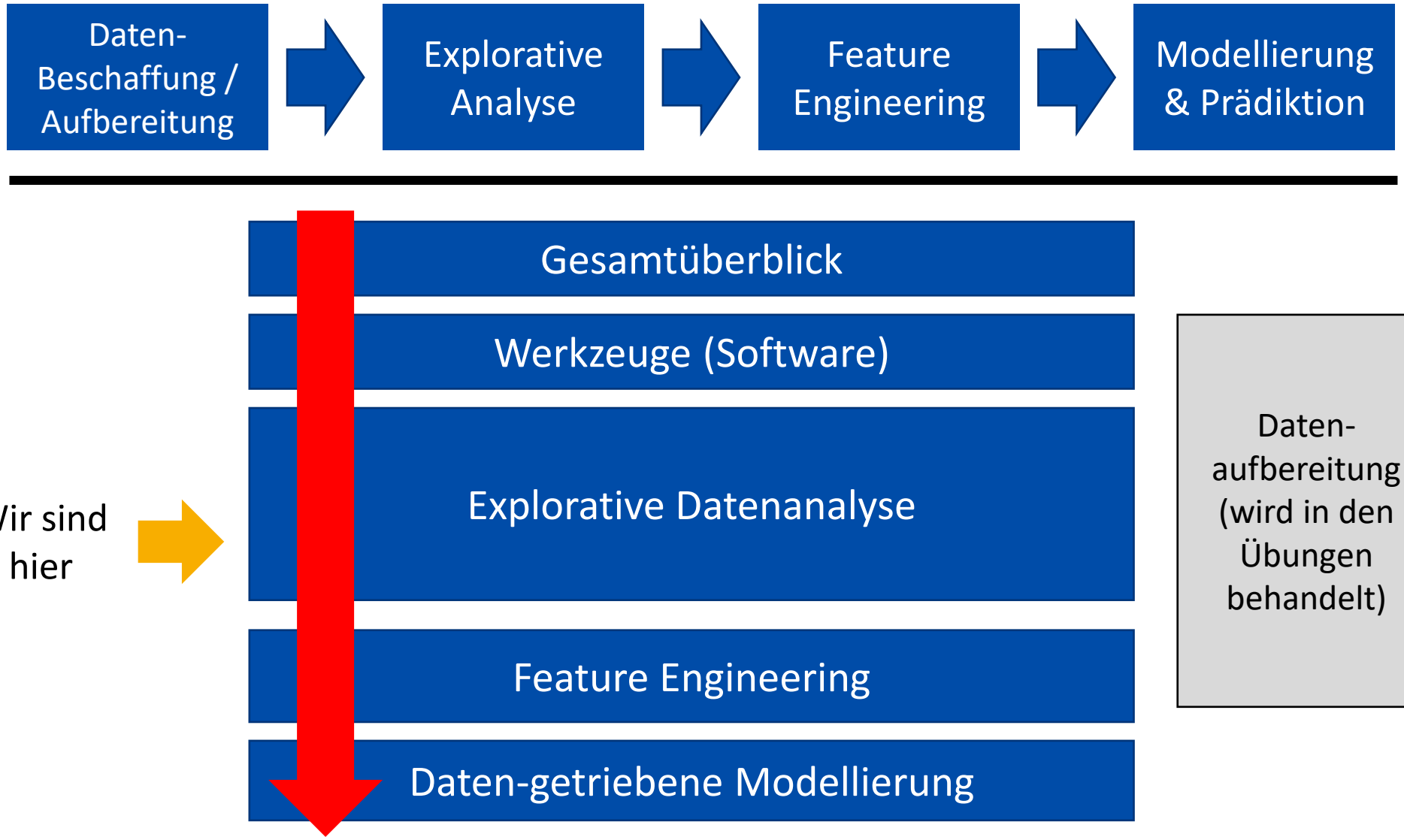
Einführung in Data Science

Unser Plan für heute:

1. Nachträge zur PCA

2. Dimensionsreduktionsverfahren
   - Multidimensional Scaling (MDS)
   - Isomap

Data Science



Curriculum

1. Grundbegriffe / Überblick
2. Zentrale Softwarebibliotheken

3. Univariate explorative Analyse (EDA) Visualisierung (Teil 1)
4. Visualisierung (Teil 2), Multivariate explorative Analyse (Teil 1)
5. Multivariate explorative Analyse (Teil 2)
6. Dimensionsreduktion (Teil 1): PCA
7. Dimensionsreduktion (Teil 2): MDS, Isomap
8. Clustering: K-Means, HCA
9. Clustervalidierung

10. Probeklausur

11. Feature Engineering, Datengetriebene Modellierung (Teil 1)
12. Datengetriebene Modellierung (Teil 2)

Überblick / Begriffe

Explorative Analyse (EDA)

Feature Engineering & Modellierung

Multivariate Explorative Analyse

Methoden der multivariaten explorativen Analyse,
die Sie bisher kennengelernt haben:

1. Multivariate deskriptive Statistik
   (hier vor allem: Visualisierungsarten)

2. Korrelationskoeffizienten
   (bivariate Analyse: Suche nach Zusammenhängen)

Nun:

3. Dimensionsreduktionsverfahren

Dimensionsreduktion | PCA | Nachträge

Principal Component Analysis (PCA)

- *Varianzmaximierung:* neue Achsen (Komponenten) der PCA sind die, bezüglich derer die Daten eine *möglichst große Varianz* aufweisen.
- *Eigenvektoren der Kovarianzmatrix*: Achsen (Richtungen) der PCA
- *Eigenwerte der Kovarianzmatrix*: Varianz der Daten bzgl. der Achse

X : (*N*x*P*) Matrix mit *N* Datenpunkten zu je *P* Merkmalen (Features)
Featurevektoren sind die Zeilen dieser Matrix

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x_n} = \frac{1}{N}X^{\mathrm{T}}\vec{\mathbf{1}}$$ mittlerer Vektor der Daten (Mittelwertsvektor)

Annahme für diese Vorlesungseinheit: Daten X sind schon zentriert, das heißt: $\bar{\mathbf{x}} = \mathbf{0}$

Kovarianzmatrix

$$S = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^{\mathrm{T}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^{\mathrm{T}} = \frac{1}{N}X^{\mathrm{T}}X$$

Dimensionsreduktion | PCA | Nachträge

Eigenwertzerlegung der (*P*x*P*) Kovarianzmatrix $S = \frac{1}{N} X^{\mathrm{T}} X$

Algorithmen zur Eigenwertzerlegung skalieren typischerweise mit $O(P^3)$.

Typische Fälle in der Praxis:

$P \ll N$

- viele Datenpunkte mit „wenig“ Dimensionen

Eigenwertzerlegung der (*P*x*P*) Kovarianzmatrix:

$S = \frac{1}{N} X^{\mathrm{T}} X$

(kennen Sie schon)

$N \ll P$

- wenige Datenpunkte mit „vielen“ Dimensionen

Eigenwertzerlegung der (*N*x*N*) Matrix:

$\frac{1}{N} X X^{\mathrm{T}}$

(betrachten wir jetzt)

Dimensionsreduktion | PCA | Nachträge

Gibt es einen Zusammenhang zwischen Eigenwerten und Eigenvektoren der Kovarianzmatrix $S = \frac{1}{N} X^{\mathrm{T}} X$ und Matrix $\frac{1}{N} X X^{\mathrm{T}}$?
Seien $\mathbf{u}_i$ die Eigenvektoren und λ_i die Eigenwerte der Kovarianzmatrix *S*:

$$S\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

$$\frac{1}{N} X^{\mathrm{T}} X \mathbf{u}_i = \lambda_i \mathbf{u}_i$$

| multipliziere X von links

$$\frac{1}{N} X X^{\mathrm{T}} \underbrace{(X\mathbf{u}_i)}_{\mathbf{v}_i} = \lambda_i \underbrace{(X\mathbf{u}_i)}_{\mathbf{v}_i}$$

Beobachtung: $\frac{1}{N} X X^{\mathrm{T}}$ hat Eigenvektoren $\mathbf{v_i}$

$$\frac{1}{N} X X^{\mathrm{T}} \mathbf{v}_i = \lambda_i \mathbf{v}_i$$

Beobachtung: $\frac{1}{N} X^{\mathrm{T}} X$ und $\frac{1}{N} X X^{\mathrm{T}}$ haben die Eigenwerte λ_i .

$$\underbrace{\frac{1}{N} X^{\mathrm{T}} X}_{S} \left(X^{\mathrm{T}} \mathbf{v}_i\right) = \lambda_i \left(X^{\mathrm{T}} \mathbf{v}_i\right)$$

| multipliziere X^{T} von links

Das bedeutet: Wir können die Eigenvektoren der Kovarianzmatrix indirekt über die (ggf. kleinere) Matrix $\frac{1}{N} X X^{\mathrm{T}}$ bestimmen!

Dimensionsreduktion | PCA | Nachträge

Also:
Falls $N \ll P$ ist es rechnerisch meist günstiger, die PCA über die Eigenwertzerlegung der *NxN* Matrix $\frac{1}{N} X X^{\mathrm{T}}$ durchzuführen:

1. Bestimme Eigenwerte λ_i und Eigenvektoren $\mathbf{v}_i$ von $\frac{1}{N} X X^{\mathrm{T}}$.
   Eigenwerte λ_i sind auch Eigenwerte der Kovarianzmatrix *S*.

2. Wegen $\underbrace{\frac{1}{N} X^{\mathrm{T}} X}_{S} \left(X^{\mathrm{T}} \mathbf{v}_i \right) = \lambda_i X^{\mathrm{T}} \mathbf{v}_i$ (siehe vorherige Folie)

   setze als Eigenvektoren der Kovarianzmatrix S: $\mathbf{u}_i \propto X^{\mathrm{T}} \mathbf{v_i}$

3. Normiere die Eigenvektoren auf Länge 1: $\mathbf{u}_i = \frac{1}{||X^{\mathrm{T}} \mathbf{v_i}||} X^{\mathrm{T}} \mathbf{v_i}$

Multidimensional Scaling (MDS)

Multidimensional Scaling (MDS)

- ist eine Familie von Verfahren für die Visualisierung von Punkten (durch Finden geeigneter Koordinaten) sowie zur Dimensionsreduktion von Daten
- erstes (sog.) klassisches Verfahren wurden 1938 (Young and Householder) und 1952 (Torgerson[1]) entwickelt
  → ist auch bekannt unter dem Namen *Principal Coordinate Analysis (PCoA) bzw. Togerson-Scaling*
- ist Ausgangspunkt für die Entwicklung weiterer Verfahren, z. B. *Isomap* zur Extraktion nichtlinearer Mannigfaltigkeiten

Wir betrachten im Folgenden *classical multidimensional scaling.*
(= Torgerson-Scaling[1])

1) Torgerson WS, Psychometrika 17, 401-419 (1952)

Multidimensional Scaling (MDS)

Ausgangssituation

gegeben:

- **paarweise Distanzen** zwischen *N* Objekten:

  Sei $D = (d_{rs})$ die *NxN* Matrix der *quadrierten* paarweisen Distanzen.

gesucht:

- **Datenmatrix *X*, die die Koordinaten (Features) für jedes der *N* Objekte enthält**, so dass die Objekte in diesen Koordinaten die paarweisen (quadrierten) Distanzen $D = (d_{rs})$ aufweisen.
- Mit den Koordinaten lassen sich die Objekte z.B. visualisieren.

Wichtig

- MDS startet mit einer Distanzmatrix D. Datenmatrix *X* ist meist unbekannt. Bei der PCA ist *X* hingegen bekannt.

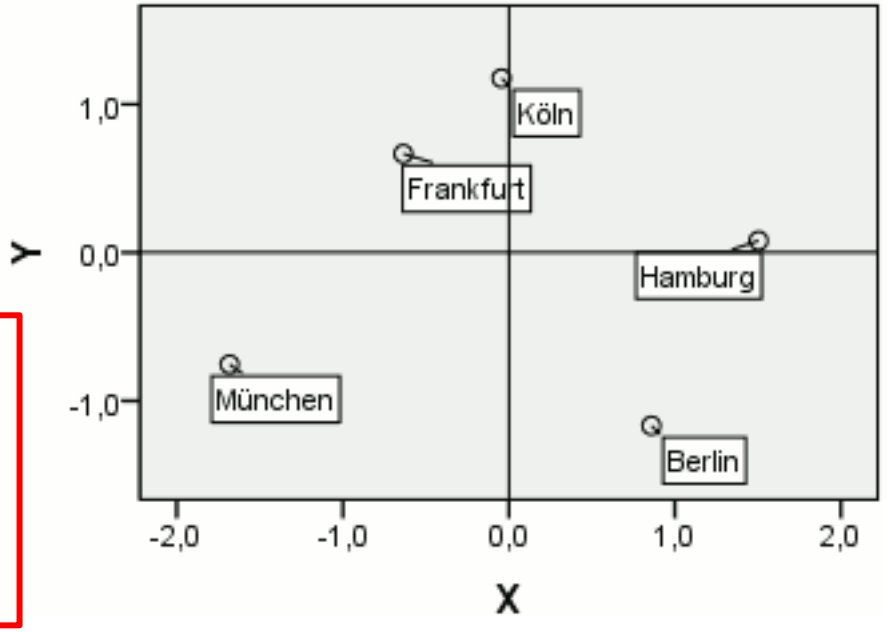
Multidimensional Scaling (MDS) | Beispiel

Paarweise Distanzen der schnellsten Autoverbindungen [km][1]

	Berlin	Frankfurt	Hamburg	Köln	München
Berlin	0	548	289	576	586
Frankfurt	548	0	493	195	392
Hamburg	289	493	0	427	776
Köln	576	195	427	0	577
München	586	392	776	577	0

gegeben:
Distanzmatrix
(hier sind die Einträge noch nicht quadriert)

gesucht:
Koordinatenmatrix *X* der Städte, sodass die Abstände den Distanzen in der Tabelle entsprechen



durch MDS gefundene Koordinaten der Städte

Lösung nicht eindeutig: Distanzen invariant unter orthogonalen Transformationen (Rotation, Spiegelung) sowie Koordinaten-ursprungsverschiebung

1) https://de.wikipedia.org/wiki/Multidimensionale_Skalierung

Multidimensional Scaling (MDS)

MDS als Methode zur Dimensionsreduktion

1. Die Daten seien als *(NxP)* Matrix *X* bekannt:
   Jeder der *N* Datenpunkte hat *P* Dimensionen (Features).

2. Bestimme Distanzmatrix *D* aus *X*, z.B. über die paarweisen euklidischen Distanzen und quadriere jeden Matrixeintrag (→ *classical multidimensional scaling*)

3. Mithilfe von MDS: Bestimme aus *D* neue Datenmatrix X` mit *Q* < *P* Dimensionen.

X` ist eine *(NxQ)* Matrix, so dass die neuen Distanzen D` im niedrigdimensionalen Raum den Distanzen D im ursprünglichen Raum so ähnlich wie möglich sind: $d_{rs} \approx d'_{rs}$

Multidimensional Scaling (MDS)

Herleitung des Classical Multidimensional Scaling (MDS)

Gegeben: (*NxN*) Distanzmatrix $D = (d_{rs})$
(quadrierte euklidische Distanzen)

Gesucht: Datenmatrix *X*

Wir gehen wie folgt vor:

1. Drücke eine (*NxN*) Matrix *B* als Funktion der Distanzmatrix *D* aus, wobei *B* definiert sein soll als: $B = XX^{\mathrm{T}}$

2. Konstruiere Datenmatrix *X* durch eine Eigenwertzerlegung von *B*

Multidimensional Scaling (MDS)

1. Drücke eine (*NxN*) Matrix *B* als Funktion der Distanzmatrix *D* aus, wobei *B* definiert sein soll als: $B = XX^{\mathrm{T}}$
   X ist unbekannt, hat aber die Dimension (*NxP*).

Wir wissen über *B:* $B = XX^{\mathrm{T}}$

Also: $$b_{rs} = \sum_{j=1}^{P} x_{rj} x_{sj} \qquad (1)$$

Wir stellen zunächst $D = (d_{rs})$ als Funktion von $B = (b_{rs})$ dar:

$$d_{rs} = \sum_{j=1}^{P} (x_{rj} - x_{sj})^2$$

(quadrierter **euklidischer Abstand** zwischen Datenpunkten *r* und *s)*

$$= \sum_{j=1}^{P} (x_{rj}^2 - 2x_{rj}x_{sj} + x_{sj}^2)$$

$$= b_{rr} - 2b_{rs} + b_{ss} \qquad (2)$$

Wir müssen nun Gleichung (2) nach b_{rs} auflösen.

Multidimensional Scaling (MDS)

$$d_{rs} = b_{rr} - 2b_{rs} + b_{ss} \quad (2)$$

Bevor wir Gleichung (2) nach b_{rs} auflösen, sammeln wir noch ein paar Fakten:

(3) Wir legen fest: Mittelwertsvektor der Daten sei 0: $\boxed{\sum_{r=1}^{N} x_{rj} = 0 \; \forall j}$

Aus $b_{rs} = \sum_{j=1}^{P} x_{rj} x_{sj}$ folgt mit (3):

(4) $\sum_{r=1}^{N} b_{rs} = \sum_{r=1}^{N} \sum_{j=1}^{P} x_{rj} x_{sj} = 0$ (Zeilensumme verschwindet)

(5) $\sum_{s=1}^{N} b_{rs} = \sum_{s=1}^{N} \sum_{j=1}^{P} x_{rj} x_{sj} = 0$ (Spaltensumme verschwindet)

(6) $\sum_{r=1,s=1}^{N} b_{rs} = \sum_{r=1}^{N} \sum_{s=1}^{N} \sum_{j=1}^{P} x_{rj} x_{sj} = 0$ (Summe über alle Einträge verschwindet)

Multidimensional Scaling (MDS)

$$d_{rs} = b_{rr} - 2b_{rs} + b_{ss} \quad (2)$$

Bevor wir Gleichung (2) nach b_{rs} auflösen, sammeln wir noch ein paar Fakten:

tr = Trace (Spur einer Matrix)

(7) $$\sum_{r=1}^{N} d_{rs} = \sum_{r=1}^{N} b_{rr} - 2\overbrace{\sum_{r=1}^{N} b_{rs}}^{\text{0 wegen (4)}} + \sum_{r=1}^{N} b_{ss} = \operatorname{tr}(B) + Nb_{ss}$$

$$\Longleftrightarrow \boxed{b_{ss} = \frac{1}{N}\left(\sum_{r=1}^{N} d_{rs} - \operatorname{tr}(B)\right)}$$

(8) $$\sum_{s=1}^{N} d_{rs} = \sum_{s=1}^{N} b_{rr} - 2\overbrace{\sum_{s=1}^{N} b_{rs}}^{\text{0 wegen (5)}} + \sum_{s=1}^{N} b_{ss} = Nb_{rr} + \operatorname{tr}(B)$$

$$\Longleftrightarrow \boxed{b_{rr} = \frac{1}{N}\left(\sum_{s=1}^{N} d_{rs} - \operatorname{tr}(B)\right)}$$

(9) $$\sum_{r,s=1}^{N} d_{rs} = \sum_{r,s=1}^{N} b_{rr} - 2\overbrace{\sum_{r,s=1}^{N} b_{rs}}^{\text{0 wegen (6)}} + \sum_{r,s=1}^{N} b_{ss} = 2N\operatorname{tr}(B)$$

$$\Longleftrightarrow \boxed{2\operatorname{tr}(B) = \frac{1}{N}\sum_{r,s=1}^{N} d_{rs}}$$

Multidimensional Scaling (MDS)

Zurück zur Gleichung (2):

$$d_{rs} = b_{rr} - 2b_{rs} + b_{ss} \quad (2)$$

Wir lösen (2) nach b_{rs} auf:

$$b_{rs} = -\tfrac{1}{2}\left(d_{rs} - b_{rr} - b_{ss}\right)$$

Einsetzen der Gleichungen (7) und (8)

$$= -\tfrac{1}{2}\left[d_{rs} - \tfrac{1}{N}\left(\sum_{s=1}^{N} d_{rs} - \mathrm{tr}(B)\right) - \tfrac{1}{N}\left(\sum_{r=1}^{N} d_{rs} - \mathrm{tr}(B)\right)\right]$$

$$= -\tfrac{1}{2}\left[d_{rs} - \tfrac{1}{N}\sum_{s=1}^{N} d_{rs} - \tfrac{1}{N}\sum_{r=1}^{N} d_{rs} + \tfrac{1}{N}2\,\mathrm{tr}(B)\right]$$

Einsetzen der Gleichung (9)

$$= -\tfrac{1}{2}\left[d_{rs} - \tfrac{1}{N}\sum_{s=1}^{N} d_{rs} - \tfrac{1}{N}\sum_{r=1}^{N} d_{rs} + \tfrac{1}{N^2}\sum_{r,s=1}^{N} d_{rs}\right]$$

(10) $B = -\tfrac{1}{2}HDH \quad$ mit $\quad H = I - \tfrac{1}{N}\mathbf{1}\mathbf{1}^{\mathrm{T}}$

Identitätsmatrix

Zentrierungsmatrix H: Setzt Zeilen- und Spaltensummen auf 0.
→ „Doppelzentrierung“

Multidimensional Scaling (MDS)

Herleitung des Classical Multidimensional Scaling (MDS)

Gegeben: (*NxN*) Distanzmatrix $D = (d_{rs})$
(quadrierte euklidische Distanzen)

Gesucht: Datenmatrix *X*

Wir gehen wie folgt vor:

1. Drücke eine (*NxN*) Matrix *B* als Funktion der Distanzmatrix *D* aus, wobei *B* definiert sein soll als: $B = XX^{\mathrm{T}}$

2. Konstruiere Datenmatrix *X* durch eine Eigenwertzerlegung von *B*

Multidimensional Scaling (MDS)

2. Konstruiere Datenmatrix *X* durch eine Eigenwertzerlegung von *B*

Wir wissen, dass *B* folgende Gestalt hat: $B = XX^{\mathrm{T}}$ (aber wir kennen *X* nicht)

Das bedeutet:

- B ist symmetrisch und reellwertig $B^{\mathrm{T}} = B$ ⇒

Aus der linearen Algebra wissen Sie:
Reelle symmetrische Matrizen sind orthogonal diagonalisierbar!

Kurze Erinnerung (lineare Algebra)

$B\mathbf{v}_i = \lambda_i \mathbf{v}_i$ mit Eigenwerten λ_i und Eigenvektoren $\mathbf{v}_i$

$BV = V\Lambda$ (*V*: Matrix der Eigenvektoren; Λ : Diagonalmatrix mit Eigenwerten auf Diagonalen)

$B = V\Lambda V^{-1}$ (spektrale Zerlegung der Matrix B)

Multidimensional Scaling (MDS)

$$\boxed{B = V \Lambda V^{-1}}$$

Aufgrund der Symmetrie von *B* gilt: $V^{-1} = V^{\mathrm{T}}$ (V ist orthogonal)

Begründung:

Für $\lambda_i \neq \lambda_j$ sehen wir:

Symmetrie von B

$$\lambda_i \mathbf{v}_i^{\mathrm{T}} \mathbf{v}_j = (B\mathbf{v}_i)^{\mathrm{T}} \mathbf{v}_j = \mathbf{v}_i^{\mathrm{T}} B^{\mathrm{T}} \mathbf{v}_j \overset{\downarrow}{=} \mathbf{v}_i^{\mathrm{T}} B \mathbf{v}_j = \lambda_j \mathbf{v}_i^{\mathrm{T}} \mathbf{v}_j$$

$$\Rightarrow \mathbf{v}_i^{\mathrm{T}} \mathbf{v}_j = \begin{cases} 1 & \text{falls } i = j \\ 0 & \text{sonst} \end{cases}$$

$V^{\mathrm{T}} V = I$ Identitätsmatrix

$\Rightarrow$ $\boxed{B = V \Lambda V^{\mathrm{T}}}$ (spektrale Zerlegung der Matrix B)

Multidimensional Scaling (MDS)

✓ 2. Konstruiere Datenmatrix *X* durch eine Eigenwertzerlegung von *B*

Wir wissen nun einerseits: $B = V\Lambda V^{\mathrm{T}}$

Andererseits fordern wir: $B = XX^{\mathrm{T}}$

$$\Rightarrow \boxed{X = V\Lambda^{\frac{1}{2}}}$$

Wir konstruieren also die Datenmatrix *X*:

$$X = V\Lambda^{\frac{1}{2}} = [\sqrt{\lambda_1}\mathbf{v}_1, \sqrt{\lambda_2}\mathbf{v}_2, \ldots, \sqrt{\lambda_P}\mathbf{v}_P]$$

wobei wir die Eigenwerte und assozierten Eigenvektoren der Größe absteigend sortiert haben.

Multidimensional Scaling (MDS)

Algorithmus:

1. Ermittle die Distanzmatrix D (**quadrierte** euklidische Distanzen)

2. Ermittle Matrix B durch Zentrierung der Matrix D:

$$B = -\tfrac{1}{2} H D H \quad \text{mit} \quad H = I - \tfrac{1}{N} \mathbf{1}\mathbf{1}^{\mathrm{T}}$$

I: Identitätsmatrix

$\mathbf{1}$: Vektor mit Einser-Einträgen

3. Führe eine Eigenwertzerlegung von B durch und erzeuge mit deren Eigenvektoren und Eigenwerten die Datenmatrix X,

$$X = [\sqrt{\lambda_1}\mathbf{v}_1, \sqrt{\lambda_2}\mathbf{v}_2, \ldots, \sqrt{\lambda_P}\mathbf{v}_P]$$

wobei die Eigenwerte und assoziierten Eigenvektoren der Größe absteigend sortiert wurden.

MDS | Wahl der Dimensionsanzahl

Wie entscheidet man über die Anzahl der Dimensionen,
in der Sie die Daten darstellen wollen?

→ Genau so wie bei der PCA: „Proportion of Variance Explained“

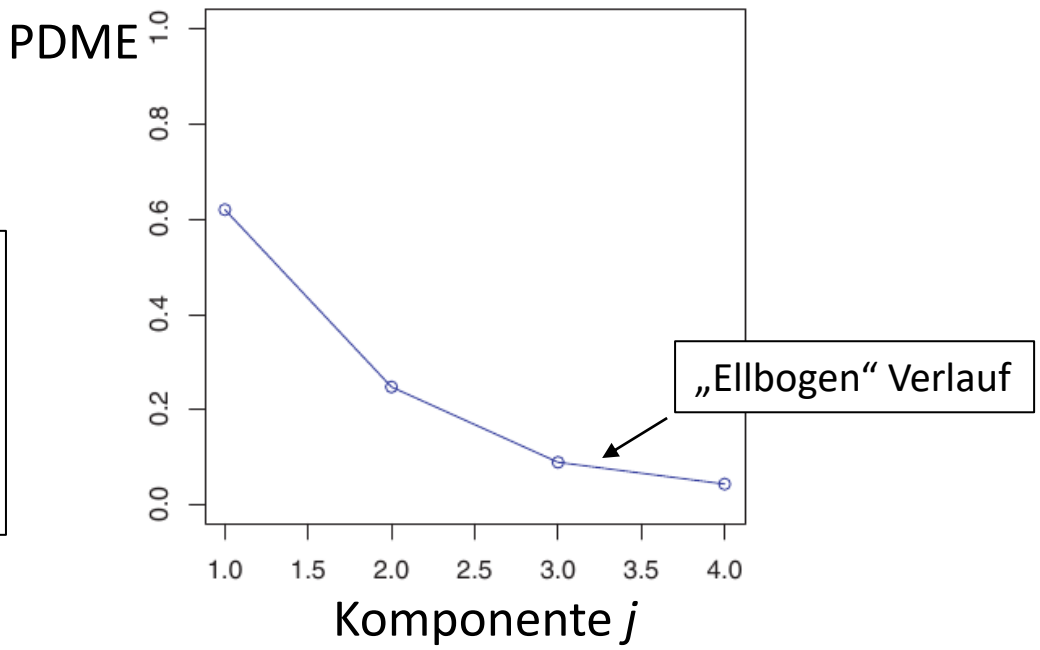
Proportion of Distance Matrix Explained (PDME)[1]

$$\mathrm{PDME}(j) = \frac{\lambda_j}{\sum_{i=1}^{N} |\lambda_i|}$$

$$\forall j \text{ mit } \lambda_j \geq 0$$

Wir entscheiden uns für die Anzahl an Komponenten, bei der die PDME bereits „abgeknickt“ ist (genau wie bei der PCA)!

Beispiel



1) K. V. Mardia, Multivariate Analysis, Academic Press 1979 (London)

Äquivalenz von MDS und PCA

Wenn
- Datenmatrix *X* **zentriert** ist (d.h. Mittelwertvektor ist 0)
- Distanzmatrix *D* aus **euklidischen Distanzen** besteht

dann sind
- MDS-Koordinaten der Daten = PCA-Koordinaten der Daten

Begründung: siehe Folien zu Beginn dieser Vorlesungseinheit. Matrizen $X^{\mathrm{T}}X$ (PCA) und XX^{T} (MDS) haben gemeinsame Eigenwerte und ihre Eigenvektoren sind proportional zueinander.

Warum benötigen wir dann überhaupt MDS, wenn wir PCA schon kennen? Weil:

1. Andere Ausgangssituation (Distanzmatrix bekannt, Koordinaten unbekannt)
2. MDS ist Ausgangspunkt für verschiedene Verfahren, z.B. Isomap

Dimensionsreduktion | PCA

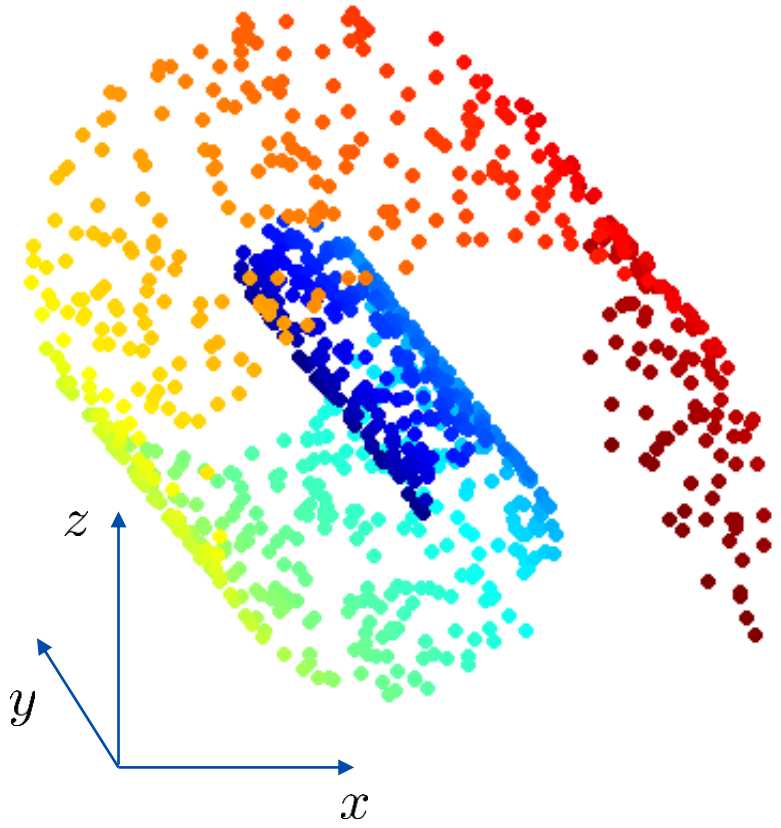
F

Grenzen der PCA

Beispiel:

Frage

- Welches PCA-Komponenten erwarten Sie hier?
- Was macht den dargestellten Datensatz für die PCA so anspruchsvoll?

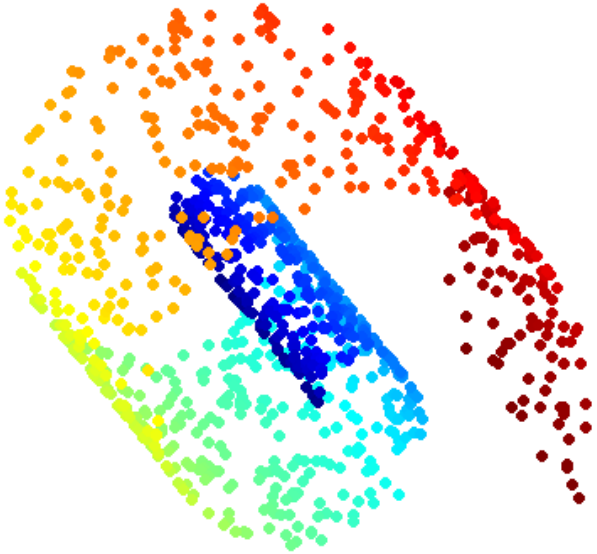


- Daten liegen auf einer nichtlinearen Mannigfaltigkeit.
- PCA hilft uns, lineare Strukturen aufzudecken → keine gute Charakterisierung nichtlinearer Strukturen möglich

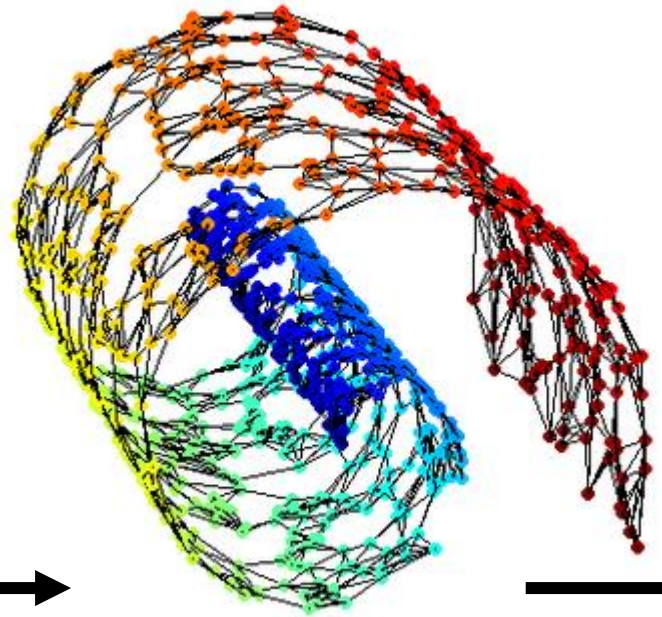
Dimensionsreduktion | Isomap

Beispiel für Methode zur Dimensionsreduktion bei nichtlinearer Strukturen:

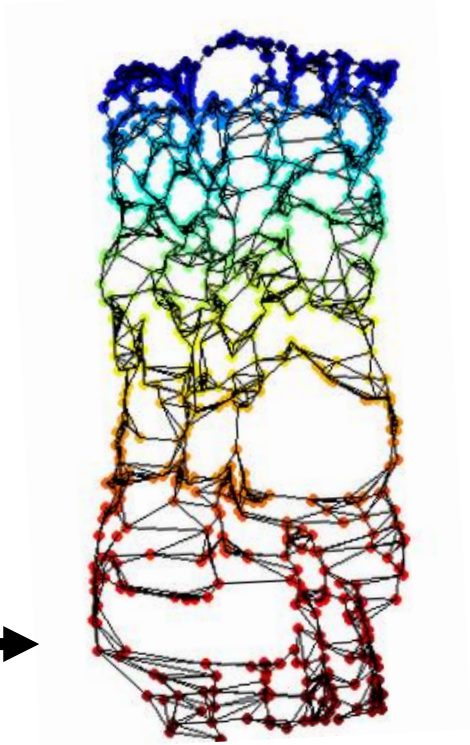
- Isomap – (*isometric feature mapping)*[1]



Erzeugung eines Nachbarschaftsgraphen: Datenpunkte, die nah (euklidisch) zueinander stehen, werden mit einer Kante verbunden.



„Graph-Distanz“ zwischen zwei Datenpunkten ist Summe der Distanzen auf auf dem kürzesten Pfad.



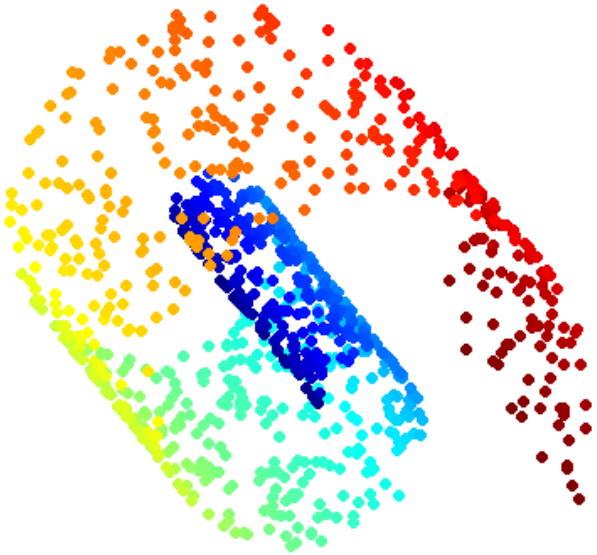
Distanzmatrix liefert Eigenvektoren zur Dimensionsreduktion.

1) Tenenbaum et al, Science 290, 2319-2323, 2000

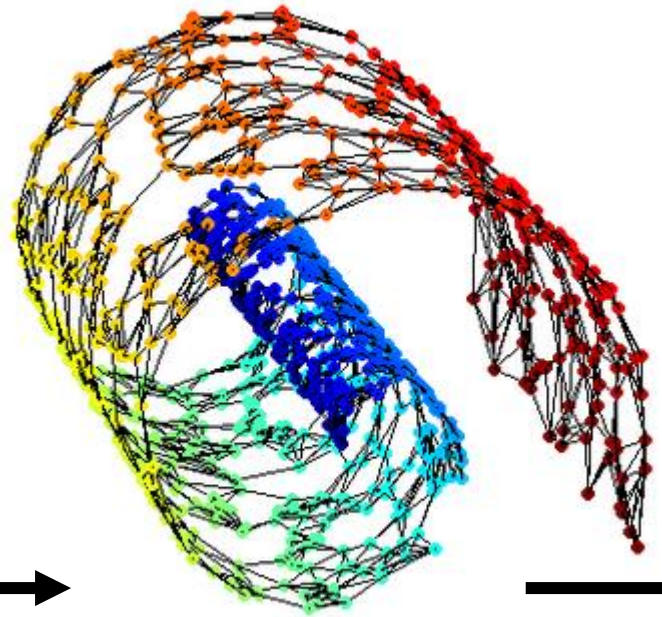
Dimensionsreduktion | isomap

Frage
Welchen Parameter werden Sie bei *isomap* einstellen müssen?

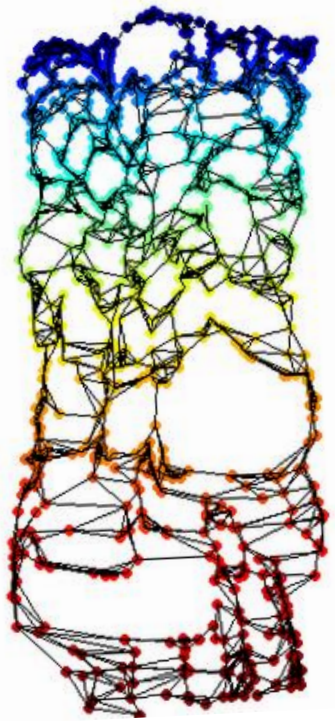
F



Erzeugung eines Nachbarschaftsgraphen: Datenpunkte, die nah (euklidisch) zueinander stehen, werden mit einer Kante verbunden.



„Graph-Distanz“ zwischen zwei Datenpunkten ist Summe der Distanzen auf auf dem kürzesten Pfad.



Distanzmatrix liefert Eigenvektoren zur Dimensionsreduktion.

1) Tenenbaum et al, Science 290, 2319-2323, 2000

Dimensionsreduktion | Isomap

- Isomap nutzt Multidimensional Scaling (MDS), aber errechnet die Distanzmatrix D aus einem Nächste-Nachbarn Graphen der Datenpunkte

Algorithmus

1. Ermittle Nachbarschaftsgraph mit euklidischen Distanzen:
   (a) Verbinde einen Punkt mit seinen k-nächsten Nachbarn
   (b) Verbinde einen Punkt mit allen Nachbarn innerhalb eines Epsilonballs

2. Ermittle paarweise Distanzen zwischen Punkten auf dem Graph; quadriere diese Distanzen → Distanzmatrix D

3. Wende Multidimensional Scaling auf Distanzmatrix D an und erhalte neue Datenmatrix X.