Beginn vl5 , blatt 4

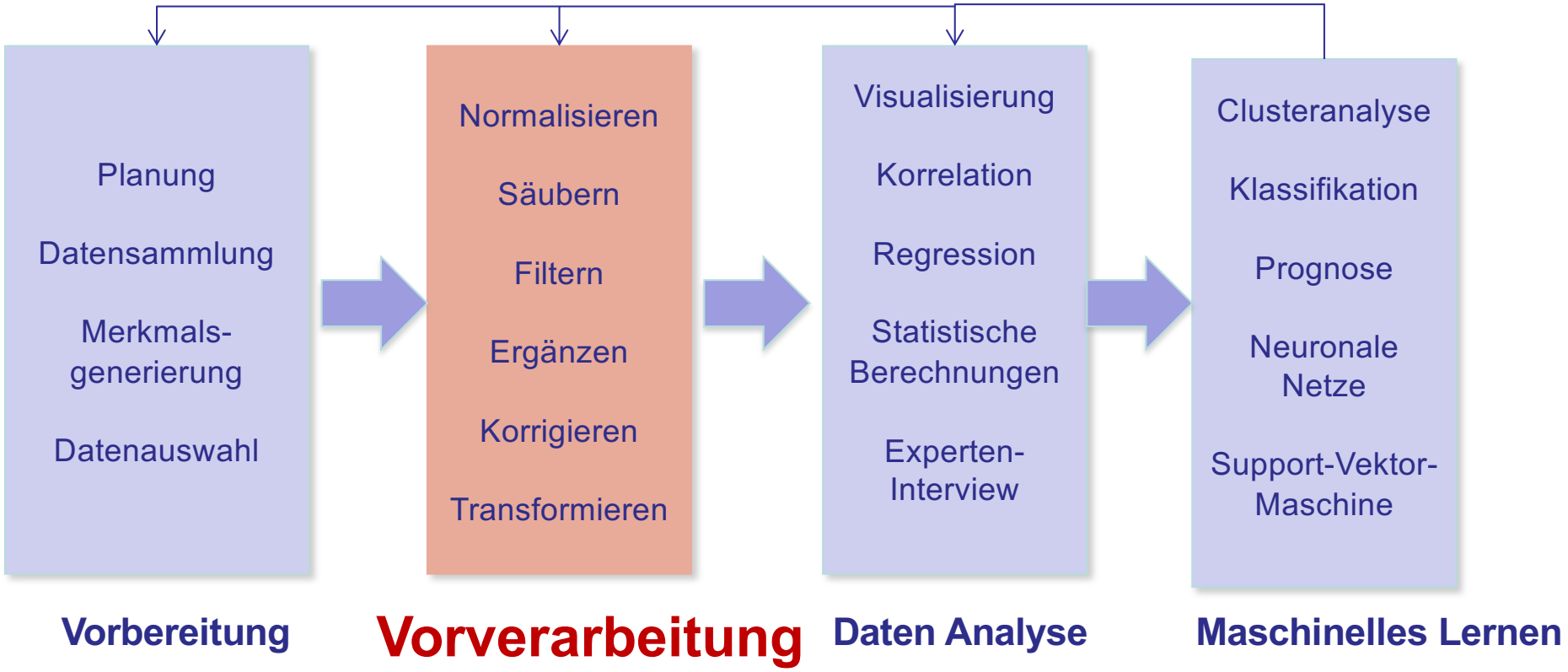
Lehrinhalte heute

1. Einstieg zu Datenanalyse und Einführung in Maschinelles Lernen
2. **Datenanalyse mit Python (Abschluss)**
    1. **Vorverarbeitung**
    2. **Bivariate Datenanalyse**

Lehrinhalte heute

1. Einstieg zu Datenanalyse und Einführung in Maschinelles Lernen
2. **Datenanalyse mit Python (Abschluss)**
    1. **Vorverarbeitung**
    2. Bivariate Datenanalyse

Das Data-Mining ist ein Prozess



Vorverarbeitung

- Fehlerarten
- Ausreißer
- Standardisierung

Vorverarbeitung

- **Fehlerarten**
- Ausreißer
- Standardisierung

Fehlerarten in „gemessenen“ Datensätzen

- ***Die Erkennung und Behandlung von Fehlern in Datensätzen ist ein wichtiger Schritt in der Datenvorverarbeitung beim Data-Mining.***

Welche Fehlerarten treten auf?

Nennen Sie jeweils Beispiele!

Zufällige Fehler

Systematische Fehler

Fehlerarten in Datensätzen

- ***Die Erkennung und Behandlung von Fehlern in Datensätzen ist ein wichtiger Schritt in der Datenvorverarbeitung beim Data-Mining.***

Zufällige Fehler

- .....
- ....
- .....
- .....

Systematische Fehler

- ....
- .....
- .....
- ......

Fehlerarten in Datensätzen

- ***Die Erkennung und Behandlung von Fehlern in Datensätzen ist ein wichtiger Schritt in der Datenvorverarbeitung beim Data-Mining.***

Zufällige Fehler

- Messfehler durch Übertragungsfehler
  - Ausreißer durch manuelle Erfassung
  - Verwechseln von Ziffern
  - Datenkommunikation fällt kurzzeitig aus
  - Störsignale in der Datenkommunikation
- Ausreißer durch Verarbeitungsfehler
  - Kommata/Punkte als Dezimaltrennzeichen falsch übertagen
  - Verschiedene Datenformate in Systemen

Systematische Fehler

- Messfehler durch Kalibrierungsfehler
- Falsche Skalierung
- Können bei bekannter Systematik gut korrigiert werden

Vorverarbeitung

- Fehlerarten
- **Ausreißer**
- Standardisierung

Erinnerung(1/3): Matrixdarstellung

Jeder numerische Merkmalssatz lässt sich als Menge X schreiben:

mit der Anzahl von ***n*** Elementen im Datensatz : $n \in \{1,2,...\}$

Jedes Element ist ein *p* -dimensionaler reellwertiger Merkmalsvektor $p \in \{1,2,...\}$

als Matrix:

$$X = \begin{pmatrix} x_1^{(1)} & . & . & . & x_1^{(p)} \\ . & . & & & . \\ . & & . & & . \\ . & & & . & . \\ x_n^{(1)} & . & . & . & x_n^{(p)} \end{pmatrix}$$

Wobei jede ***Zeile*** einem Element der Datenmenge entspricht und wird auch als ***Merkmalsvektor x_k mit k= 1,...n*** bezeichnet.
Und jede ***Spalte*** einer ***Komponente*** aller Elemente einer Datenmatrix entspricht und wird auch als i-tes Merkmal oder i-te Komponente ***$x^{(i)}$*** bezeichnet mit ***i=1,..,p***
Ein Matrixelement $x_k^{(i)}$ entspricht einer ***Komponente eines Elements.***

Erinnerung (2/3): Mittelwert

Das ***arithmetische Mittel*** des i-ten Merkmals des Datensatzes X:

$$\overline{x}^{(i)} = \frac{1}{n}\sum_{k=1}^{n} x_k^{(i)}$$

- n läuft über alle Messwerte der i-ten Kompomente

Erinnerung (3/3) : Standardabweichung

Die ***Standardabweichung vom arithmetischen Mittelwert*** des i-ten Merkmals des Datensatzes X, sie ist ein Maß für die Streuung einer Verteilung um den Mittelwert:

$$s^{(i)} = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}\left(x_k^{(i)} - \overline{x}^{(i)}\right)^2}$$

- n läuft über alle Messwerte der i-ten Komponente

Erkennung von Ausreißern I

Ausreißer in Datensätzen können das Analyseergebnis stark verfälschen und sollten meistens behandelt werden.

1)<u>Starke Abweichung von den übrigen Daten mit statistischen Maßen :</u>

z.B.

2-Sigma-Regel *: wenn in mind. einer Komponente ein Wert existiert, dessen Abstand vom Mittelwert dieser Komponente größer als die doppelte Standardabweichung in diesem Merkmal ist:*

$$x_k \in X\_ist\_Ausreißer \Leftrightarrow \exists i \in \{1,\ldots,p\} : \left| \frac{x_k^{(i)} - \overline{x^{(i)}}}{s^{(i)}} \right| > 2$$

- ***3-Sigma-Regel*** oder ***4 -Sigma -Regel*** sind auch denkbar.
- ***<u>Achtung:</u>*** Exoten *dürfen so nicht automatisch entfernt werden, sie enthalten meist wertvolle Informationen.*

Beispiel für 2,3-Sigma-Regel

```
from sklearn.datasets import load_iris
iris = load_iris()

outliers = []  #Definition  einer Liste zum Einsammeln der Ausreißer
data_std = iris.data.std()     # Standardabweichung  über alle Elemente u. alle  4 Attribute
data_mean = iris.data.mean()    # mean über alle 150 Werte x 4 Attribute

outlier_cut_off = data_std * 2     #2_Sigma-Regel, sonst *3, *4 etc.

lower_limit  = data_mean - outlier_cut_off #untere Grenze für 2-Sigma-Regel
upper_limit = data_mean + outlier_cut_off  #obere Grenze für 2-Sigma-Regel
print(lower_limit)
print(upper_limit)

n_samples,n_features = iris.data.shape
for elem in range (n_samples):          # über alle Elemente des IRIS-Datensatzes egal welche Art
   for attr in range (n_features):      # alle Attribute auch gleich behandeln
      if iris.data[elem][attr]> upper_limit or iris.data[elem][attr]< lower_limit:
         outliers.append(iris.data[elem][attr])
print(outliers)
```

Output für 2-Sigma_Regel:

```
-0.48318611551965596
7.412186115519655
[7.6, 7.7, 7.7, 7.7, 7.9, 7.7]
```

Wie sieht der Output vermutlich im Vergleich mit der 3-Sigma-Regel aus?

Beispiel für 2,3-Sigma-Regel

```
...
outliers = []  #Definition  einer Liste zum Einsammeln der Ausreißer
data_std = iris.data.std()     # Standardabweichung  über alle Elemente u. alle  4 Attribute
data_mean = iris.data.mean()    # mean über alle 150 Werte x 4 Attribute

outlier_cut_off = data_std * 2     #2_Sigma-Regel, sonst *3, *4 etc.

lower_limit  = data_mean - outlier_cut_off #untere Grenze für 2-Sigma-Regel
upper_limit = data_mean + outlier_cut_off   #obere Grenze für 2-Sigma-Regel
print(lower_limit)
print(upper_limit)

n_samples,n_features = iris.data.shape
for elem in range (n_samples):          # über alle Elemente des IRIS-Datensatzes egal welche Art
   for attr in range (n_features):      # alle Attribute auch gleich behandeln
      if iris.data[elem][attr]> upper_limit or iris.data[elem][attr]< lower_limit:
         outliers.append(iris.data[elem][attr])
print(outliers)
```

Output für 2-Sigma_Regel:

```
-0.4831861155196596
7.412186115519655
[7.6, 7.7, 7.7, 7.7, 7.9, 7.7]
```

Output für 3-Sigma_Regel:

```
-2.457029173279484
9.386029173279484
[]
```

Gibt es im IRIS-Datensatz „Outlier“? Was meinen Sie?

Erkennung von Ausreißern II

2) **Werte liegen außerhalb des Wertebereiches:**

$$x_k \in X \; ist \; Fehler \Leftrightarrow \exists i \in \{1,\ldots,p\} : \left(x_k^{(i)} < x^i_{\min}\right) \vee \left(x_k^{(i)} > x^i_{\max}\right)$$

- *Wertebereichsgrenzen definieren sich durch:* ***?***
  - *Vorzeichen (Temperatur, Zeit, Preis)*
  - *Messbereich des Aufnahmesensors*
  - *Bereich sinnvoller Werte (Expertenbefragung, z.B. Physiker, Umweltforscher, Psychologen, ...*)

3) Konstante Merkmale (sind zwar keine Ausreißer, besitzen aber auch keine Information) :

$$x_k^{(i)} = x_l^{(i)} : \forall k,l = 1,\ldots n$$

Behandlung von Ausreißern I

a) Welche Behandlung können Sie sich für Ausreißer in den Datensätzen vorstellen? Was macht Sinn? Was ist informationstechnisch möglich?
b) Wie würden Sie evtl. Fehlerstellen korrigieren?

?

Behandlung von Ausreißern I

Die Behandlung wird davon abhängig gemacht, wie stark die Ausreißer den Datensatz verändern:

- ***Markierung,***
- ***Korrektur oder***
- ***(teilw.) Entfernung der Ausreißerdaten.***

Markierung als Ausreißer (nachfolgend ist gesonderte Behandlung möglich)
z.B. durch **Erstellen einer Markierungsmatrix M zum Datensatz X:**

$$m_k^{(i)} = \begin{cases} 1\_falls\_x_k^{(i)} gültig \\ 0\_sonst \end{cases}$$

Korrektur der Merkmale von Ausreißern

a) Ersetzung durch den Maximalwert oder Minimalwert der Komponente:

$$x_f^{(i)} = \begin{cases} \max\left\{ x_k^{(i)} \middle| m_k^{(i)} = 1 \right\} falls\ x_f^{(i)} zu\ groß \\ \min\left\{ x_k^{(i)} \middle| m_k^{(i)} = 1 \right\} falls\ x_f^{(i)} zu\ klein \end{cases}$$

Behandlung von Ausreißern II

2)Korrektur der Merkmale von Ausreißern

b) Ersetzung durch den globalen Mittelwert der gültigen Werte:

$$x_f^{(i)} = \frac{\sum_{k=1}^{n} m_k^{(i)} \cdot x_k^{(i)}}{\sum_{k=1}^{n} m_k^{(i)}}$$

c) Ersetzung durch nächste Nachbarn der Ausreißerwerte:

$$\min_{k \in \{1,\ldots,n\}} \left| x_f^{(1,\ldots,i,i+1,\ldots p)} - x_k^{(1,\ldots,i-1,i+1,\ldots p)} \right|$$

wobei Abstände zu ungültigen Daten als unendlich definiert werden:

$$Abstände = \infty, falls$$

$$\exists j \in \{1,\ldots,i-1,i+1,\ldots,p\} \cdot m_k^{(j)} = 0$$

Behandlung von Ausreißern III

2)Korrektur der Merkmale von Ausreißern

d) Lineare Interpolation bei Zeitreihen mit äquidistanten Zeitschritten:

$$x_f^{(i)} = \frac{x_{f-1}^{(i)} + x_{f+1}^{(i)}}{2}$$

d) Lineare Interpolation bei Zeitreihen mit nicht äquidistanten Zeitschritten:

$$x_f^{(i)} = \frac{x_{f-1}^{(i)} \cdot (t_{f+1} - t_f) + x_{f+1}^{(i)} \cdot (t_f - t_{f-1})}{2}$$

e) Weitere Möglichkeiten sind:

- nichtlineare Interpolation z.B. durch Splines
- Filterung des Signals (dann Veränderung aller Daten)
- Ergänzung der Werte durch Schätzung von Wahrscheinlichkeitsdichten
- modellbasierte Ergänzungen
- ...

Vorverarbeitung

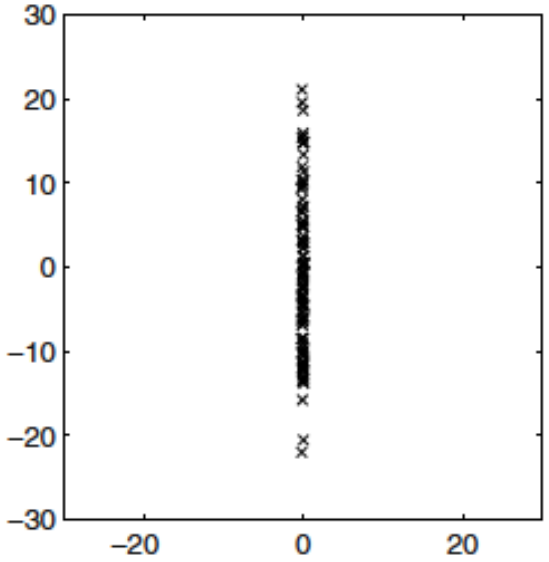
- Fehlerarten
- Ausreißer
- **Standardisierung**

Standardisierung

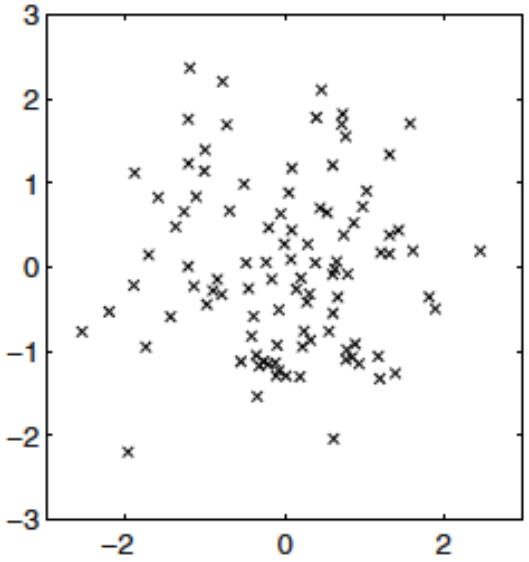
*Einzelne Merkmale in Datensätzen haben häufig **sehr unterschiedliche Wertebereiche**. Numerisch kleinere Merkmale werden dann gegenüber numerisch größeren Merkmalen bei vielen Analyseverfahren (z.B. NN) stark vernachlässigt.*

Was kann man tun?

*Lösung ist die **automatische Berechnung geeigneter Wertebereiche** durch z.B. :*
•Standardisierung des Hyperwürfels



Nicht standardisierter Merkmalsraum



Standardisierter Merkmalsraum

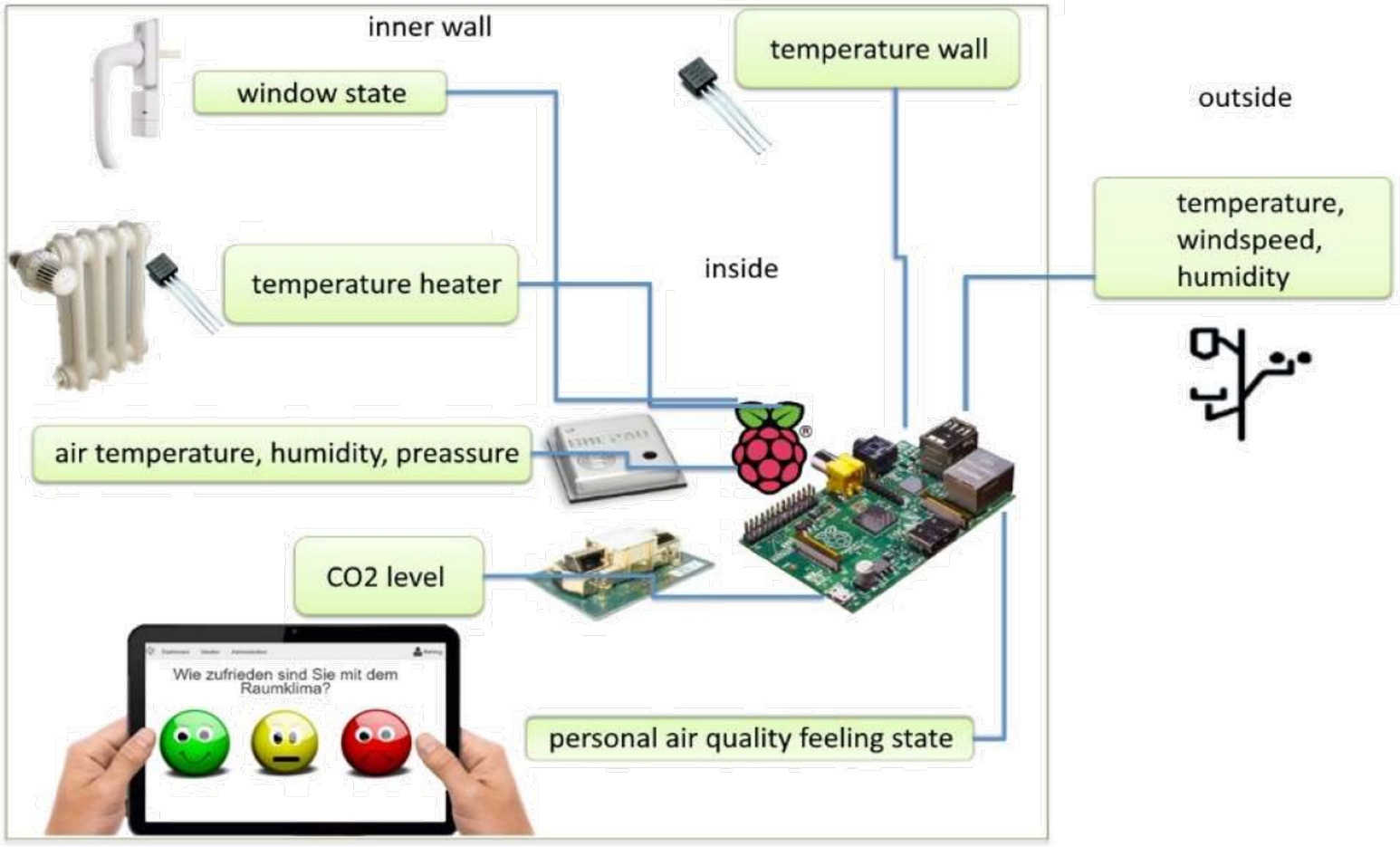
Standardisierung: Beispieldatensatz zur Luftqualität

BMBF-Forschungsprojekt ENVIRON (2019-2022)

Intelligente Datenanalyse mit Maschine Learning:

- ***Schimmelwarner*** (Prognose nach Training mit Lerndatensatz + analytische Formeln)
- Erkennen des ***Fensteröffnungszustandes***
- Ablesen/Berechnen der Heizungszustände (an / aus, Stufe, wie lange)
- Lernen der ***Luftqualität*** angepasst an ***persönliche Vorlieben***
- ***Assistent*** zum effizienten und gesunden ***Lüftungs- und Heizungsverhalten***
- Lernen von ***Verhaltenstypen*** in Bezug auf das Umweltverhalten/Heizungsverhalten

Standardisierung: Beispieldatensatz zur Luftqualität



Standardisierung: Beispieldatensatz zur Luftqualität

	humidity_inside	temperature_inside	co2_inside	temperature_heater	temperature_wall_inside	state_air_quality
0	58.94	21.955	1653	41.187	14.062	1
1	58.97	21.940	1665	40.937	14.062	1
2	57.89	22.353	1707	40.375	14.250	1
3	58.96	22.504	1981	40.875	14.375	1
4	57.74	22.515	1868	40.812	14.375	1

** rawdata_luftqualitaet.csv*

** air_quality : 0 – gut, 1-neutral, 2 - schlecht*

	humidity_inside	temperature_inside	co2_inside	temperature_heater	temperature_wall_inside	state_air_quality
count	916.000000	916.000000	916.000000	916.000000	916.000000	916.000000
min	**34.120000**	**17.194000**	**395.000000**	**16.687000**	**8.812000**	**0.000000**
max	**75.510000**	**27.613000**	**5544.00000**	**52.625000**	**15.750000**	**2.000000**

Standardisierung

Allgemein ist über einem mehrdimensionalen Datensatz X (dim=p) ein jeweils gleichdimensionaler Merkmalsraum $H^*(X)$ aufgespannt, der durch die jeweiligen Wertebereichsgrenzen x_{min} und x_{max} definiert ist:

$$H^*(X) = \left[x_{\min}^{(1)}, x_{\max}^{(1)}\right] \times \left[x_{\min}^{(2)}, x_{\max}^{(2)}\right] \times \ldots \times \left[x_{\min}^{(p)}, x_{\max}^{(p)}\right],$$

$$wobei\ x_{\min}^{(i)} = \min\left\{X^{(i)}\right\}, \quad x_{\max}^{(i)} = \max\left\{X^{(i)}\right\}$$

Standardisierung des Hyperwürfels: Alle Seiten des definierten Merkmalsraumes(Hyperwürfel) sollen gleich lang erscheinen durch eine Transformation aller Merkmalsvektoren aus X in Y mit:

$$y_k^{(i)} = \frac{x_k^{(i)} - x_{\min}^{(i)}}{x_{\max}^{(i)} - x_{\min}^{(i)}}$$

Standardisierung mit sklearn: MinMaxScaler

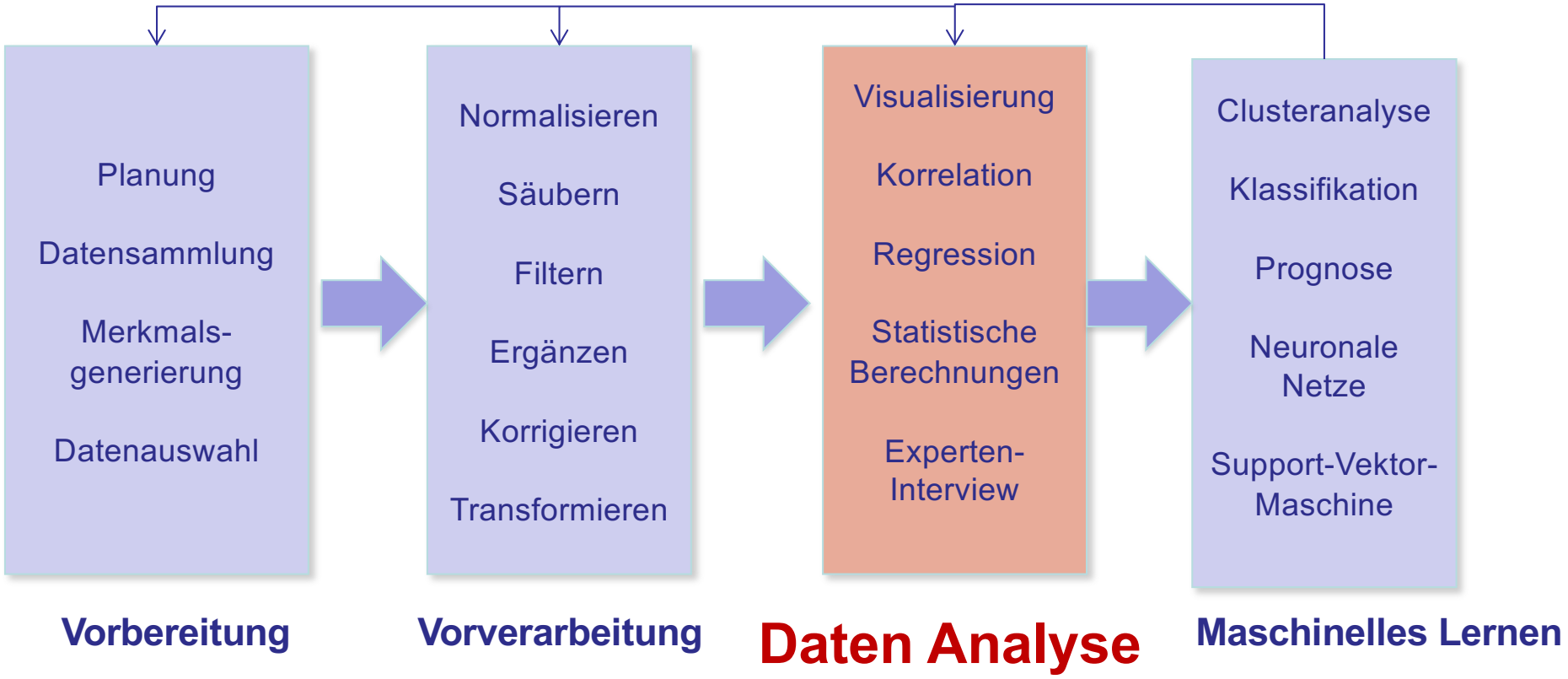
```
from sklearn import preprocessing  # library preprocessing importieren
...
mm_scaler = preprocessing.MinMaxScaler()# initialisieren von MinMaxScaler
X_train_normalized = mm_scaler.fit_transform(X_train)# Transformieren
```

Der MinMaxScaler

- berechnet für jede Spalte im DataFrame(feature) die Werte im standardisierten Hyperwürfel
- D.h. von jedem Wert in der Spalte wird das Minimum in der Spalte abgezogen und dann durch den Wertebereich dividiert.
- Der Wertebereich ist die Differenz zwischen Maximum und Minimum in dieser Spalte

	humidity_inside	temperature_inside	co2_inside	temperature_heater	temperature_wall_inside
0	0.505065	0.252383	0.061589	0.738728	0.524231
1	0.000000	0.513598	0.078834	0.872060	0.320441
2	0.321609	0.912995	0.475878	0.778367	0.912550
3	0.544457	0.600000	0.506262	0.699089	0.902920
4	0.431064	0.512895	0.202423	0.781971	0.708760

Das Data-Mining ist ein Prozess



Lehrinhalte heute

1. Einstieg zu Datenanalyse und Einführung in Maschinelles Lernen
2. Datenanalyse mit Python (Abschluss)
    1. Vorverarbeitung
    2. **Bivariate Datenanalyse**

Bivariate Datenanalyse

- Korrelation
- Heatmap mit Seaborn-Library

Bivariate Datenanalyse untersucht die Zusammenhänge zwischen zwei Merkmalen im Datensatz.

Datenanalyse

- **Korrelation**
- Heatmap mit Seaborn-Library

Korrelationen

*Im Data Mining untersucht man häufig zuerst **Zusammenhänge innerhalb des Datensatzes***

Die Korrelation** beschäftigt sich mit der Analyse des **Zusammenhangs zwischen Merkmalen

- *kann zielgerichtete Effekte ermöglichen, indem Parameter (einzelne Merkmale oder ganze Cluster) mit großer Korrelation zum positivem gewünschten Effekt hin, erhöht werden.*
- *Zusammenhänge können **linearen** oder **nicht linearen** Charakter haben*

Lineare Korrelation auf Grundlage der ***Kovarianzen***

- *Kann häufig gut visualisiert werden in **2D-Plots***
- *zwischen zwei Merkmalen **(->Korrelationsmatrix, Heatmap)***
- *zwischen Clustern von Merkmalen (Algorithmen nötig)*

Lineare Korrelation

***Die Kovarianzmatrix C** eines n-elementigen Datensatzes $X \subset R^p$ besitzt die Einträge:*

$$c_{ij} = \frac{1}{n-1}\sum_{k=1}^{n}\left(x_k^{(i)} - \bar{x}^{(i)}\right)\left(x_k^{(j)} - \bar{x}^{(j)}\right), \quad mit \quad i,j = 1,..,p$$

***C_{ij}** quantifizieren den Zusammenhang zwischen den Merkmalen $x^{(i)}$ und $x^{(j)}$ in einem n-elementigen Datensatz mit Merkmalsvektoren der Dimension p:*
***Große** Werte - **? starker** Zusammenhang*
***Kleine** Werte - **? schwacher** Zusammenhang*

- *Große Varianzen im Merkmalswert selbst vergrößern die Werte in c_{ij}*
- ***Was kann man als Informatiker mit den Merkmalswerten tun?***
- *Lösung: Normalisierung mit dem Produkt der beiden Standardabweichungen*
- *führt zum **Pearsonschen Korrelationskoeffizient s_{ij}:***

einsetzen, umformen, ver-einfachen (Runkler Kap5.1)

$$s^{(i)} = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}\left(x_k^{(i)} - \bar{x}^{(i)}\right)^2}$$

$$s_{ij} = \frac{c_{ij}}{s^{(i)}s^{(j)}} \Rightarrow \frac{c_{ij}}{\sqrt{c_{ii}c_{jj}}}$$

$$mit\ s_{ij} \in [-1,1]$$

c_{ii} und c_{jj} – sind die Diagonalen der Kovarianzmatrix

Datenanalyse

- Korrelation
- **Heatmap mit Seaborn-Library**

HeatMap mit Seaborn

Einbinden der Python- Bibliothek Seaborn

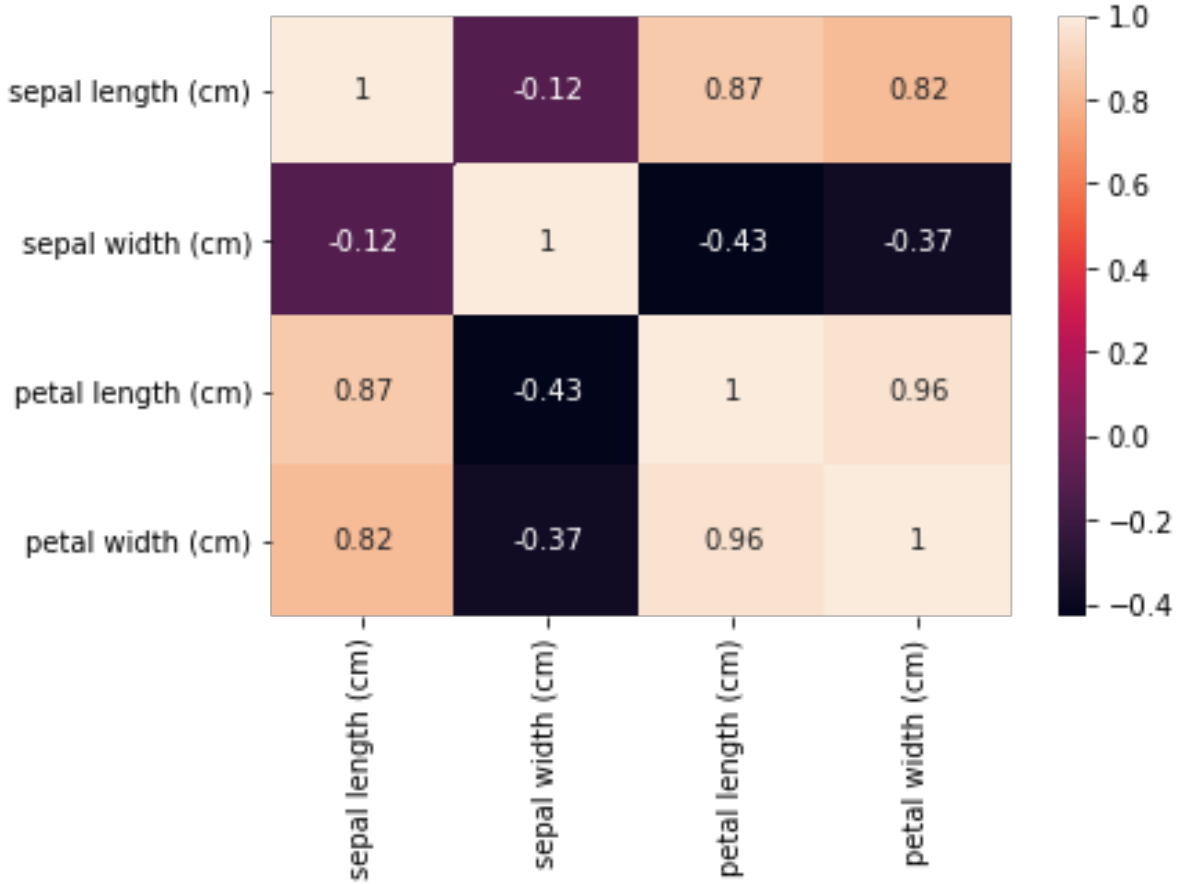
- Basiert auf matplotlib
- realisiert hochwertige Visualisierungen für statistische Daten

https://seaborn.pydata.org/

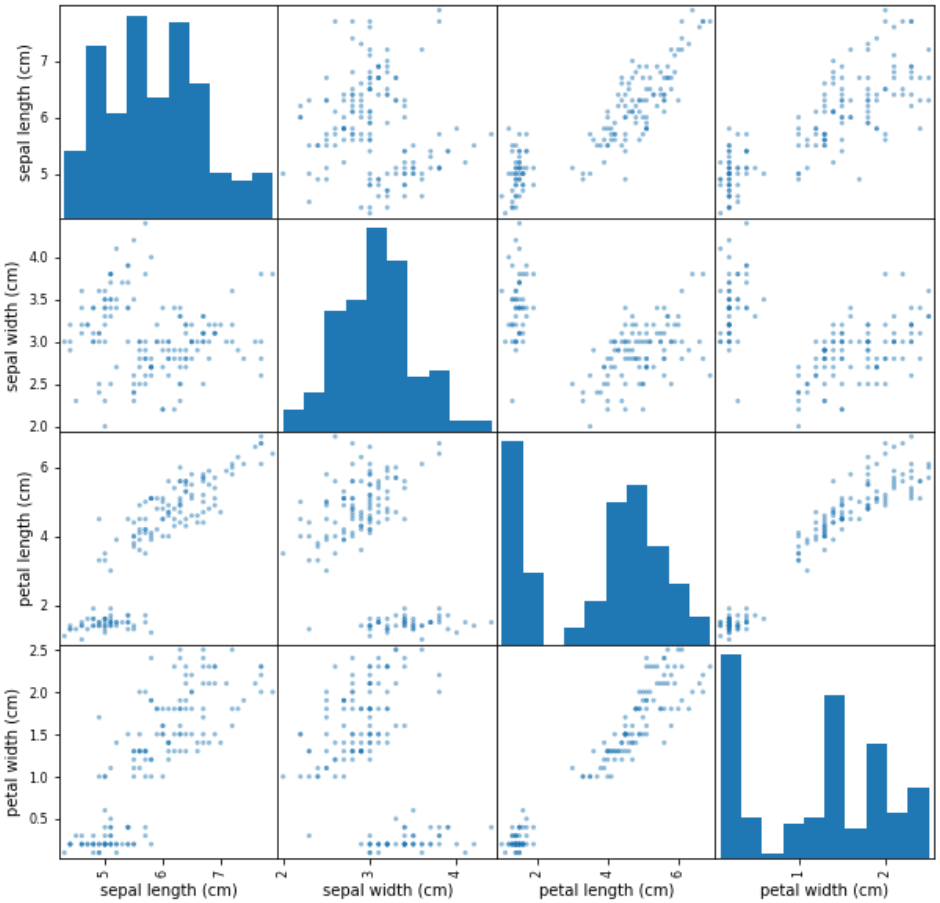
```
from sklearn.datasets import load_iris
import pandas as pd
import seaborn as sns   # Einbinden der Library seaborn
import matplotlib.pyplot as plt

iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)

print (df.corr())          # Berechnen der Korrelationsmatrix
sns.heatmap(df.corr(), annot = True); # Heatmap zeichnen
```

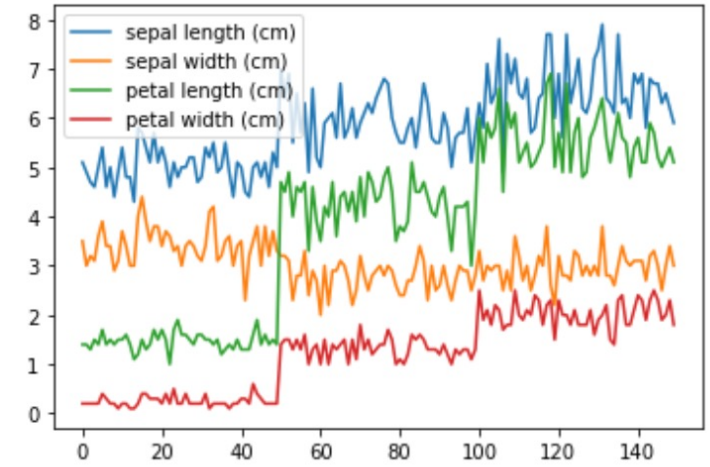
HeatMap mit Seaborn

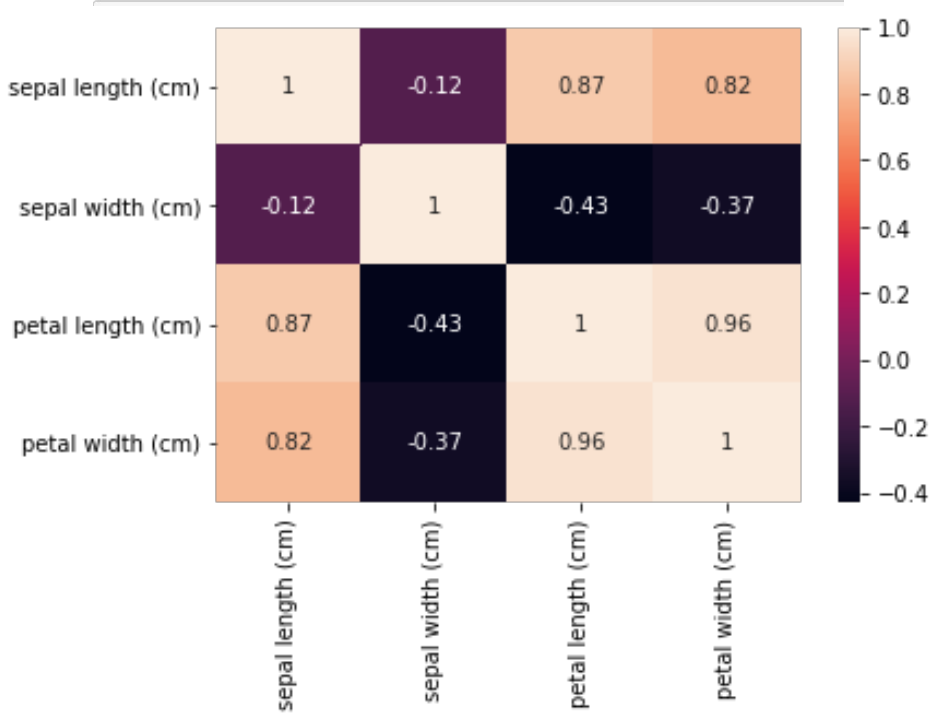
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
sepal length (cm)	1.000000	-0.117570	0.871754	0.817941
sepal width (cm)	-0.117570	1.000000	-0.428440	-0.366126
petal length (cm)	0.871754	-0.428440	1.000000	0.962865
petal width (cm)	0.817941	-0.366126	0.962865	1.000000



Scattermatrix und Liniendiagramm für IRIS im Vergleich:

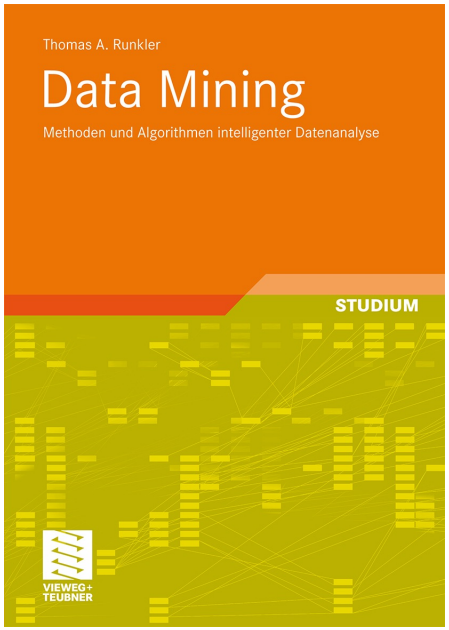






Bis hierhin Blatt 4 Praktikum

Literaturangaben und Quellen



Thomas A. Runkler
„Data Mining – Methoden und Algorithmen intelligenter Datenanalyse“

2. Auflage:
Verlag: Springer-Verlag GmbH
Seitenzahl: 145
Erscheinungstermin: 5. Oktober 2015 Deutsch
ISBN-13: 9783834821713

https://www.python-kurs.eu/maschinelles_lernen.php
https://www.grund-wissen.de/informatik/python/scipy/pandas.html

Lehrinhalte weiter...

1. Einstieg zu Datenanalyse und Einführung in Maschinelles Lernen
2. Datenanalyse mit Python (Abschluss)
   - Vorverarbeitung
   - Datenanalyse für Abhängigkeiten zwischen zwei Komponenten

3. Maschinelles Lernen
4. Zeitreihenanalyse