

DATA SCIENCE 1

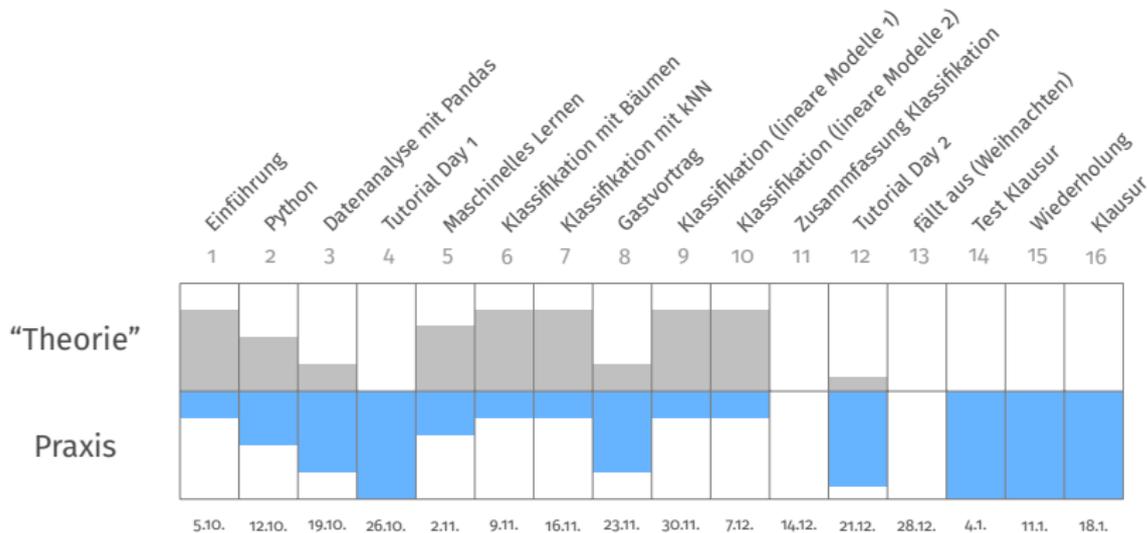
VORLESUNG 3 - INTRO

PROF. DR. CHRISTIAN BOCKERMANN

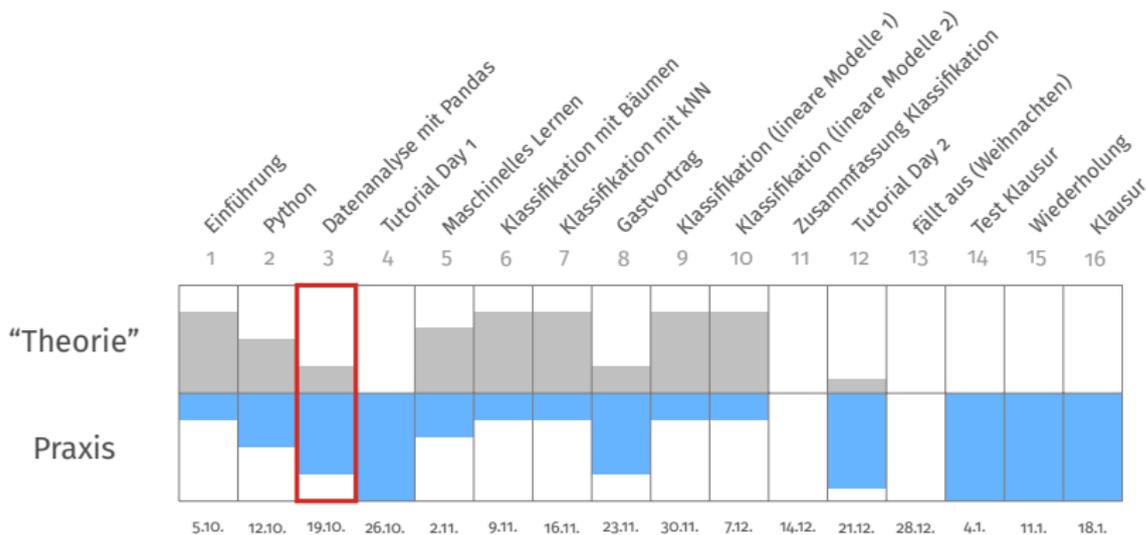
HOCHSCHULE BOCHUM

WINTERSEMESTER 2021/2022

Wo sind wir?



Wo sind wir?



Einführung in Python

- Skript-Sprache Python, Programme als Scripte (Text-Datei)
- Grundlegende Datentypen, Schleifen und eigene Funktionen
- Module zur Erweiterung des Funktionsumfangs

```
# Liste ['A', 'B', 'a', 'a',...]
liste = list("ABaaBccbbb")
anzahlBs = 0

# Zaehle die B's:
for x in liste:
    if x == 'B' or x == 'b':
        anzahlBs = anzahlBs + 1
```

Listen mit Strings

- Strings sind wie Listen von Buchstaben
- mit **in** kann geprüft werden, ob etwas in einer Liste enthalten ist:

```
wort = "Data Science"  
if "en" in wort:  
    print("'en' ist enthalten!")
```

- Listen mit Strings funktionieren natürlich auch:

```
woerter = [ "Data", "Science", "Daten"]  
for wort in woerter:  
    if "at" in wort:  
        print(wort)
```

Eigene Funktionen

- Erweiterung um eigene Funktionen mit **def**
- **return** beendet die Funktion mit Rückgabewert

```
def avg(zahlenListe)
    # Wenn Liste leer => Ergebnis 0
    if len(zahlenListe) == 0:
        return 0

    sum = 0
    for zahl in zahlenListe:
        sum = sum + zahl

    return sum / len(zahlenListe)
```

Listen für Fortgeschrittene – *list comprehension*

- Aus der Mathematik – Mengennotation:

$$M = \{x^2 \mid x \in \mathbb{N}, 5 \leq x \leq 10\}$$

Listen für Fortgeschrittene – *list comprehension*

- Aus der Mathematik – Mengennotation:

$$M = \{x^2 \mid x \in \mathbb{N}, 5 \leq x \leq 10\}$$

- *List-comprehension* als Schreibweise für Listen

```
liste = list("ABaaBccbbb")  
  
# extrahiere grosse und kleine B's  
#  
bs = [x for x in liste if x == 'b' or x == 'B']  
anzahlBs = len(bs)
```

Python - Module

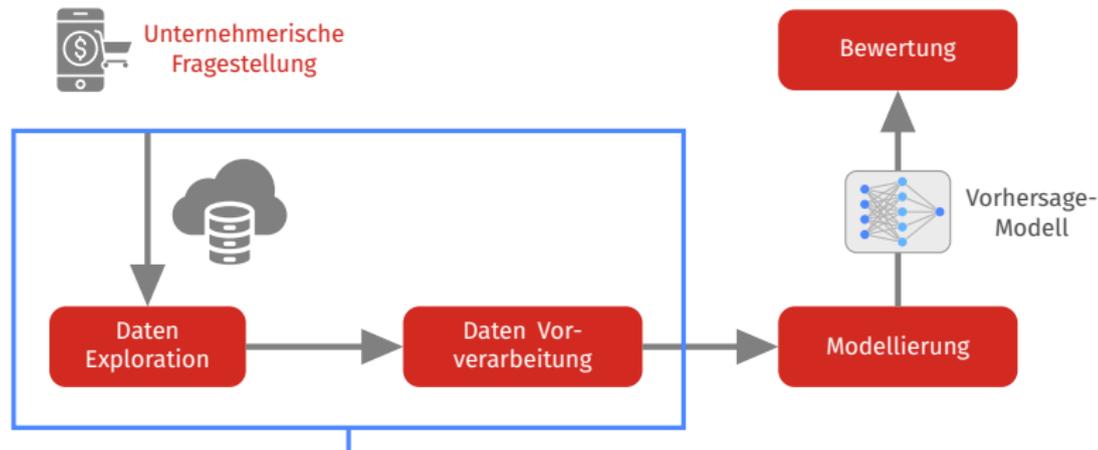
- Module als Bibliotheken mit fertigen Funktionen
- Einbindung von Modulen mit **import**
- Module mit Namensraum

```
# Binde das Modul 'datascience' im Namesraum 'ds' ein  
import datascience as ds  
  
# Einbinden von 'Zufall' in den globalen Namensraum  
from datascience.vorhersage import Zufall  
  
m = Zufall()    # statt ds.Zufall
```

Vorlesung 3

Pandas & Visualisierung

Vorgehen bei der Datenanalyse



Datenvorverarbeitung
hier: Mit Python und Pandas

Vorlesung 3 (heute):

- Vorstellung des Moduls **Pandas**
- Selektieren/Filtern von Daten
- Grundlagen zur Arithmetik mit Pandas

Vorlesung 3 (heute):

- Vorstellung des Moduls **Pandas**
- Selektieren/Filtern von Daten
- Grundlagen zur Arithmetik mit Pandas

Hochschule Bochum
Bochum University
of Applied Sciences



Fachbereich Wirtschaft
Prof. Dr. Christian Bockermann

Data Science

Wintersemester 2021/2022

Übungsblatt 3

Dieses Übungsblatt beschäftigt sich mit dem Einlesen, dem Filtern und der Exploration von Daten. Die Daten liegen als CSV-Dateien in Ihrem Verzeichnis auf dem Notebook Server.

pandas ist ein zentrales Modul für die Datenverarbeitung in Python



Pandas

- Datentypen für Tabellen (DataFrame) und Zeitreihen (Series)
- Funktionen zum Lesen/Transformieren von Daten
- Unterstützung einer Vielzahl von Formaten: CSV, Excel, Datenbanken, usw.

```
import pandas as pd  
  
# Lesen von Daten aus CSV-Datei  
df = pd.read_csv('meine_Daten.csv')
```

Einbinden des Moduls in ein Python Program

```
import pandas as pd

# optional dazu noch:
from pandas import Series, DataFrame
```

- Importiert Pandas als Modul mit Namensraum **pd**
- **pd** ist allgemein verbreiteter Namensraum für Pandas
- Häufig werden **Series** und **DataFrame** noch in den globalen Namensraum importiert (optional)

Pandas bietet Datentypen und Funktionen

- `pandas.read_csv`
- `pandas.read_excel`
- `pandas.to_csv`
- `Series.plot`
- `Series + Series`
- `DataFrame.plot`
- `DataFrame + DataFrame`
- ...

Funktionen

0	4
1	5
2	8
3	2

Series

	a1	a2	a3	a4
0	4	1	2	9
1	5	1	3	6
2	3	8	7	4

DataFrame

Datentyp für Reihe von Meßwerten: **Series**

Index	Werte
0	4
1	5
2	8
3	2

Ein **Series** Objekt aus einer Liste von Werten erzeugen:

```
data = Series([4,5,8,2])
```

Series ist wie Liste von Werten

- Series fast wie normale Python-Liste
- Kombinierbar mit **for**-Schleife:

```
data = Series([4,5,8,2])
laenge = len(data) # ergibt 4

sum = 0
for zahl in data:
    sum = sum + zahl

print(sum / laenge)
```

Pandas Datentypen unterstützen Rechenoperationen

Zum Beispiel:

0	4	* 2 =	0	8
1	5		1	10
2	8		2	16
3	2		3	4

```
a = Series([4,5,8,2])  
b = a * 2
```

DataFrame:

Ein DataFrame `df` ist eine Tabellenstruktur:

	a1	a2	a3
0	4	1	2
1	5	1	3
2	3	8	7

DataFrame:

Ein DataFrame `df` ist eine Tabellenstruktur:

	a1	a2	a3
0	4	1	2
1	5	1	3
2	3	8	7

Spalten-Index

`df.columns`

Zeilen-Index

`df.index`

DataFrame:

Ein DataFrame `df` ist eine Tabellenstruktur:

“Positionsindex”

	0	1	2	
0	0	4	1	2
1	1	5	1	3
2	2	3	8	7

Spalten-Index

`df.columns`

Zeilen-Index

`df.index`

Die Form eines DataFrame

df =

	a1	a2	a3	a4
0	4	1	2	9
1	5	1	3	6
2	3	8	7	4

df.**shape** liefert die Form von df: (zeilen,spalten)

```
# 'shape' des obigen DataFrames df  
zeilen, spalten = df.shape      # zeilen=3, spalten=4
```

Einzelne Spalten sind **Series** Objekte

	a1	a2	a3	a4
0	4	1	2	9
1	5	1	3	6
2	3	8	7	4

Einzelne Spalten sind **Series** Objekte

	a1	a2	a3	a4
0	4	1	2	9
1	5	1	3	6
2	3	8	7	4

```
a2 = df['a2'] # Spalte 'a2' selektieren
type(a2)     # -> pandas.core.series.Series

print(a2[2]) # gibt 8 aus
```

Mehrere Spalten ergeben wieder einen DataFrame



```
# z0, z1, z2 wie in Folie 29
df = DataFrame([z0, z1, z2],
               columns=['a1', 'a2', 'a3', 'a4'])
df23 = df[['a2', 'a3']]
```

Pandas enthält Funktionen zum Lesen von DataFrames

- `pd.read_csv` - Lesen aus CSV-Datei
- `pd.read_excel` - Lesen aus Excel-Datei

DataFrame aus CSV-Datei lesen:

```
# Lesen aus der Datei 'meine-daten.csv'  
df = pd.read_csv("meine-daten.csv")  
  
# Funktioniert auch mit URLs  
u = "https://datascience.hs-bochum.de/data/iris.csv"  
df = pd.read_csv(u)
```

Nachdem Laden – Was ist drin, im DataFrame?

- Wie sieht der DataFrame aus? → `df.head(5)`
- Anzahl der Zeilen/Spalten? → `df.shape`
- Welche Spalten/Datentypen? → `df.columns / df.dtypes`
- Wertebereiche der Spalten? → `df.describe()`

```
import pandas as pd
df = pd.read_csv( "Kurse/DataScience1/data/iris.csv" )

# Anfang anzeigen (ersten 5 Zeilen)
df.head(5)

# Spalten-Statistiken
df.describe()
```

Spalten-Statistiken mit `describe()`

`describe()` berechnet Statistiken für numerische Spalten

```
# Statistiken berechnen:  
df.describe()
```

	a1	a2	a3
count	3.00	3.00	3.00
mean	4.00	3.33	4.00
std	1.00	4.04	2.65
min	3.00	1.00	2.00
25%	3.50	1.00	2.50
50%	4.00	1.00	3.00
75%	4.50	4.50	5.00
max	5.00	8.00	7.00

Spalten-Statistiken mit `describe()`

`describe()` berechnet Statistiken für numerische Spalten

```
# Statistiken berechnen:  
df.describe()
```

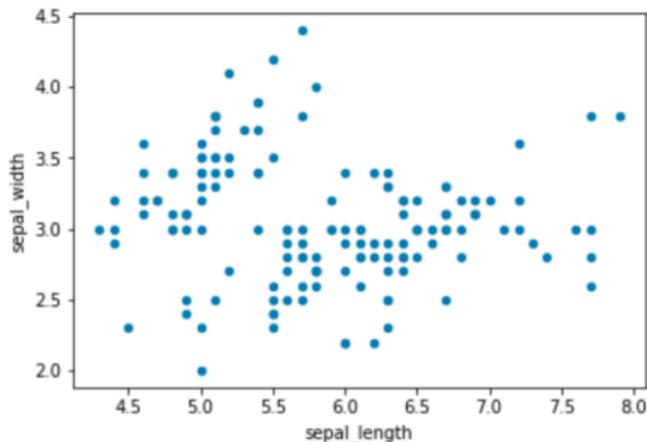
```
# Und das Ergebnis  
# von df.describe()?  
stats = df.describe()
```

Ist natürlich wieder ein DataFrame

	a1	a2	a3
count	3.00	3.00	3.00
mean	4.00	3.33	4.00
std	1.00	4.04	2.65
min	3.00	1.00	2.00
25%	3.50	1.00	2.50
50%	4.00	1.00	3.00
75%	4.50	4.50	5.00
max	5.00	8.00	7.00

Einfache Plots mit DataFrame

```
iris = pd.read_csv('data/iris.csv')  
iris.plot.scatter(x='sepal_length', y='sepal_width')
```



Vorlesung 4 (nächste Woche): **Tutorial Day**

- Keine neuen Inhalte (bzgl. Maschinelles Lernen)
- Wiederholung + Vertiefung
- Zentrales Thema: Python + Pandas

Ablauf:

- Vorlesung um 8:30 Uhr im Big Blue Button
- Es gibt Übungsaufgaben/Folien und Notebooks
- Selbststudium mit Unterstützung über Big Blue Button während der Zeit von 9 bis 12 Uhr (s.t.)