Python Pandas II

Programmierkurs 2 Data Science WS23/24

https://www.scmp.com/news/people-culture/environment/article/3164671/why-are-pandas-so-chonky-despite-their-vegan-diet

Leonard Traeger
M. Sc. Information Systems
leonard.traeger@fh-dortmund.de

Disclaimer

Slides are mainly based on

- https://pandas.pydata.org/docs/index.html and
- https://www.w3schools.com/python/pandas/pandas_intro.asp

→ Find everything you need to know there!

Official Pandas cheat sheet:

- https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf

More beginner-friendly Pandas cheat sheet by Dataquest:

- https://drive.google.com/file/d/1UHK8wtWbADvHKXFC937IS6MTnlSZC_zB/view

Learning Goals Python Pandas II

- **Explain** the synergy effects between NumPy and Pandas, and why Pandas always maintains the data context.
- **Explain, demonstrate, compare,** and **apply** suitable data preprocssing tasks onto examplarly data sets.
- **Propose** reasons for attributes and data being „dirty“.
- **Apply** data masking and **demonstrate** how to insert new records and change values of existing subsets of rows or columns of a DataFrame.
- **Discuss** hindering reasons for initial data exploration.
- **Hypothesize** reasons for missing data and **propose** strategies to annotate, find, and filling these.
- **Describe** common data integration problems and **develop** suitable methods for removing duplicates, union, and join operations onto DataFrames.
- **Justify** the application of data masking, unary functions, and binary functions in the context of data transformation. **Construct** attributes with binary functions and the lambda function.
- **Explain** grouping and aggregation functions in the context of split, apply, combine. **Sketch** cleaning, grouping, aggregation, and sorting functions on DataFrames.
- **Describe** generalization and the four types of concept hierarchies.
- **Give examples** on considerations for preprocessing data and how preprocssing can benefit or harm the analysis.

NumPy (Recap)

- NumPy arrays can be **multidimensional**.
- NumPy adds **efficient manipulation** and operations on that data.
  - Basic arithmetic (addition, subtraction, multiplication, etc.)
  - Sophisticated operations (trigonometric functions, exponential and logarithmic functions, etc.)
  - **Ufuncs benefit from Vectorization; functions that encapsulate loops!**

Operator	Equivalent ufunc	Description
`+`	`np.add`	Addition (e.g., `1 + 1 = 2`)
`-`	`np.subtract`	Subtraction (e.g., `3 - 2 = 1`)
`-`	`np.negative`	Unary negation (e.g., `-2`)
`*`	`np.multiply`	Multiplication (e.g., `2 * 3 = 6`)
`/`	`np.divide`	Division (e.g., `3 / 2 = 1.5`)
`//`	`np.floor_divide`	Floor division (e.g., `3 // 2 = 1`)
`**`	`np.power`	Exponentiation (e.g., `2 ** 3 = 8`)
`%`	`np.mod`	Modulus/remainder (e.g., `9 % 4 = 1`)

VanderPlas, J., "Python Data Science Handbook", O'Reilly, 2017

NumPy `Ufuncs` (Recap)

Which version is more efficient and looks more "pythonic" ?

```
x = np.arange(1, 101)
x1 = x.reshape(10,10)
x2 = x1[5]
for idx, item in enumerate(x2):
  x2[idx] = item + 100
print(x)
```

```
x = np.arange(1, 101)
x1 = x.reshape(10,10)
x1[5] = x1[5] + 100
print(x)
```

```
[  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
  37  38  39  40  41  42  43  44  45  46  47  48  49  50 151 152 153 154
 155 156 157 158 159 160  61  62  63  64  65  66  67  68  69  70  71  72
  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
  91  92  93  94  95  96  97  98  99 100]
```

Could this array be a Pandas Series?

NumPy and Pandas

Pandas is designed to work with NumPy!

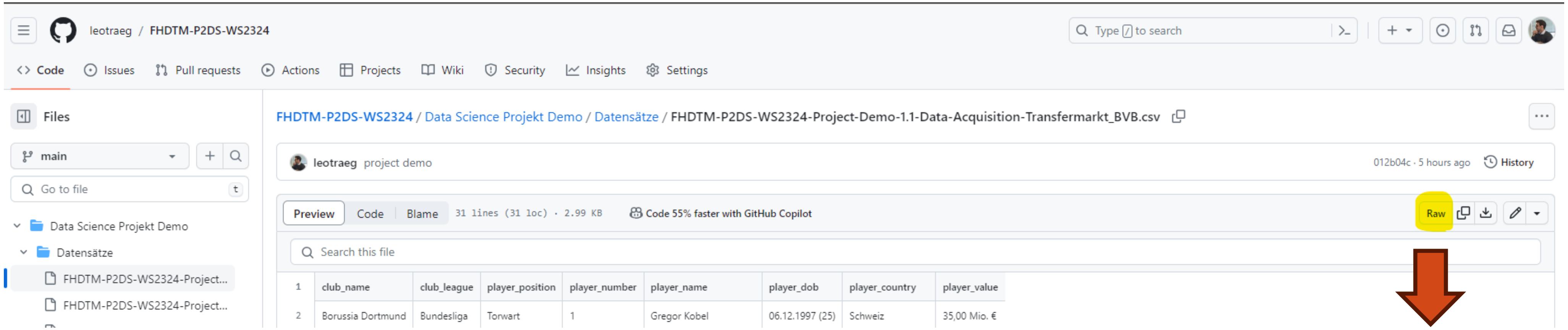
What does that mean?

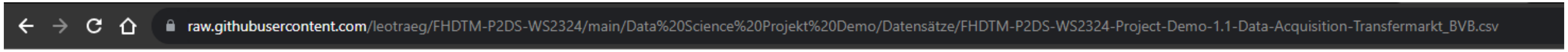
- **Any NumPy ufunc** will **work** on **numeric Pandas Series** and **DataFrame objects**!

Pandas > NumPy

- **Pandas preserves alignment** of **indices** and **columns** and thus operations on data will always **maintain** the **data context**.

I/O API Example (Recap)






```
club_name,club_league,player_position,player_number,player_name,player_dob,player_country,player_value
Borussia Dortmund,Bundesliga,Torwart,1,Gregor Kobel,06.12.1997 (25),Schweiz,"35,00 Mio. €"
Borussia Dortmund,Bundesliga,Torwart,35,Marcel Lotka,25.05.2001 (22),Deutschland,"1,50 Mio. €"
Borussia Dortmund,Bundesliga,Torwart,33,Alexander Meyer,13.04.1991 (32),Deutschland,"1,00 Mio. €"
Borussia Dortmund,Bundesliga,Torwart,31,Silas Ostrzinski,19.11.2003 (19),Deutschland,150 Tsd. €
Borussia Dortmund,Bundesliga,Abwehr,4,Nico Schlotterbeck,01.12.1999 (23),Deutschland,"40,00 Mio. €"
Borussia Dortmund,Bundesliga,Abwehr,25,Niklas Süle,03.09.1995 (27),Deutschland,"35,00 Mio. €"
```

```python
url = "https://raw.githubusercontent.com/leotraeg/FHDTM-P2DS-WS2324/main/Data%20
df_bvb_player = pd.read_csv(url)
```


Technology Arts Sciences TH Köln

Getting to know your data (Recap)

It is pretty **hard** to work, analyze and apply statistical methods on data...

...**if you do not know anything about your data!**

An **initial exploration** of your data can help you decide how to proceed.

Getting to know your data (Recap)

Take the **time** to **open** up your data **file** and have a look.

You might **be surprised** at what you find!

You may **notice obious issues** with the data, e.g.:

- Duplicate records
- Duplicate attributes
- Nonsensical values
- Useless attributes
- Incomplete data formatting during I/O ☺

Too much data to inspect manually? Take a sample!

Viewing Meta Data (Recap)

- `df.info()` returns **meta data** about the **frame**.

```
df_bvb_player.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   club_name        30 non-null     object
 1   club_league      30 non-null     object
 2   player_position  30 non-null     object
 3   player_number    30 non-null     int64 
 4   player_name      30 non-null     object
 5   player_dob       30 non-null     object
 6   player_country   30 non-null     object
 7   player_value     30 non-null     object
dtypes: int64(1), object(7)
memory usage: 2.0+ KB
```

What did go unfavorable during the I/O process?

	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value
0	Borussia Dortmund	Bundesliga	Torwart	1	Gregor Kobel	06.12.1997 (25)	Schweiz	35,00 Mio. €
1	Borussia Dortmund	Bundesliga	Torwart	35	Marcel Lotka	25.05.2001 (22)	Deutschland	1,50 Mio. €

Example: Unary Function and uFunc

Definition: `function(A)` → `A`, where `A` is a set

```
df_bvb_player.player_number %2 == 0
```

OR

```
df_bvb_player.player_number.apply(player_number_even)
```

```
def player_number_even(player_number):
  if player_number %2 == 0:
    return True
  else:
    return False
```

```
df_bvb_player.player_number
```

(

0	1
1	35
2	33
3	31
4	4
5	25
6	15
7	44
8	47
9	5
10	26
11	17
12	24
13	2
14	23
15	6

) =

0	False
1	False
2	False
3	False
4	True
5	False
6	False
7	True
8	False
9	False
10	True
11	False
12	True
13	True
14	False
15	True

For **binary operations** (two operators, e.g., function(A,B) → A*B), Pandas automatically aligns indices!

We just did some **Data Preprocessing - Data Transformation - Attribute construction**

Major Tasks in **Data Preprocessing**

Data Reduction

- Obtains reduced representation in volume but produces the same or similar analytical results.

Data Cleaning

- Fill in missing values, **smooth noisy data**, identify or remove outliers, and resolve inconsistencies caused by data integration.

Data Integration

- Integration of multiple tables, databases, data cubes, or files.

Data Transformation

- Aggregation, generalization, normalization and attribute construction.

Most of these methods require **cyclic application** throughout the **DS process**!

Data Cleaning

“Garbage in, garbage out“

No quality data, no quality analsis!

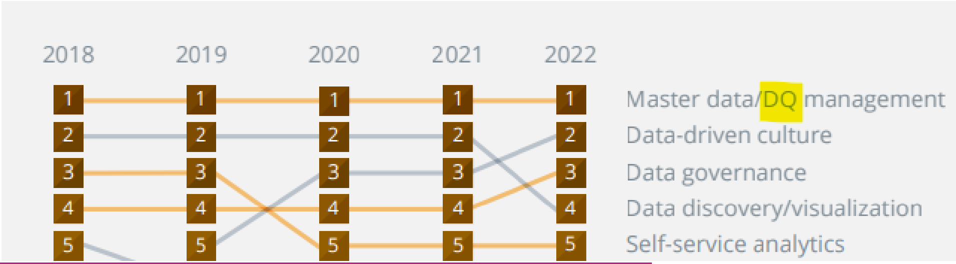
Quality decisions must be based on quality data:

- e.g., duplicate or missing data may cause incorrect or even misleading statistics!

Data **extraction**, **cleaning**, and **transformation** comprises the majority of the work of building a **data warehouse**.

https://barc.com/de/infografik-barc-data-bi-analytics-trend-monitor-2022/

Development of rankings of Data, BI and Analytics trends



Why is Data dirty?

Incomplete data

- “Not applicable“ data value when collected.
- Different considerations between the time, e.g., *when data was collected* versus *used for analysis*.
- Human/hardware/software problems.

Noisy data (incorrect values)

- Faulty data collection instruments.
- Human or computer error at data entry.
- Errors in data transmission.

Inconsistent data

- Different data sources.
- Functional dependency violation, e.g., modify some linked data.
- Duplicate records/attributes during integration process.

Data Cleaning: **smooth noisy data (I)**

Definition: `function(A)` → `A`, where `A` is a set

```python
def get_player_value_numeric(player_value):
    if(type(player_value) == float):
      return player_value
    else:
      player_value_arr = player_value.split(" ")
      #"150 Tsd. €".split(" ") --> ['150,00', 'Tsd.', '€']
      value = player_value_arr[0].replace(",",".")
      unit = 1000 if "Tsd." in player_value_arr[1] else 1000000
      return float(float(value)*unit)
```

```python
df_bvb_player.player_value
```

(

0	35,00 Mio. €
1	1,50 Mio. €
2	1,00 Mio. €
3	150 Tsd. €
4	40,00 Mio. €
5	35,00 Mio. €
6	6,00 Mio. €
7	1,00 Mio. €
8	600 Tsd. €
9	20,00 Mio. €
10	13,00 Mio. €
11	10,00 Mio. €
12	5,00 Mio. €
13	1,00 Mio. €
14	14,00 Mio. €
15	13,00 Mio. €

) =

```python
(df_bvb_player.player_value.
apply(get_player_value_numeric))
```

0	35000000.0
1	1500000.0
2	1000000.0
3	150000.0
4	40000000.0
5	35000000.0
6	6000000.0
7	1000000.0
8	600000.0
9	20000000.0
10	13000000.0
11	10000000.0
12	5000000.0
13	1000000.0
14	14000000.0
15	13000000.0

```python
from datetime import date, timedelta, datetime
```

Data Cleaning: **smooth noisy data (II)**

Definition: `function(A)` → `A`, where `A` is a set

```python
def get_player_dob_date(player_dob):
    if(type(player_dob) == datetime.date):
      return player_dob
    else:
      player_dob_arr = player_dob.split(" ")
      #"06.12.1997 (25)".split(" ") --> ['06.12.1997', '(25)']
      dob_arr = player_dob_arr[0].split(".")
      return date(int(dob_arr[2]), int(dob_arr[1]), int(dob_arr[0]))
```

```python
df_bvb_player.player_dob
```

```
0     06.12.1997 (25)
1     25.05.2001 (22)
2     13.04.1991 (32)
3     19.11.2003 (19)
4     01.12.1999 (23)
5     03.09.1995 (27)
6     16.12.1988 (34)
7     14.10.2003 (19)
8     10.09.1999 (23)
9     16.04.1995 (28)
10    17.11.1997 (25)
11    27.05.1995 (28)
12    12.09.1991 (31)
13    02.03.2000 (23)
14    12.01.1994 (29)
15    11.01.1998 (25)
```

=

```python
(df_bvb_player.player_dob.
apply(get_player_dob_date))
```

```
0     1997-12-06
1     2001-05-25
2     1991-04-13
3     2003-11-19
4     1999-12-01
5     1995-09-03
6     1988-12-16
7     2003-10-14
8     1999-09-10
9     1995-04-16
10    1997-11-17
11    1995-05-27
12    1991-09-12
13    2000-03-02
14    1994-01-12
15    1998-01-11
```

```python
from datetime import date, timedelta, datetime
```

Data Cleaning: **smooth noisy data (III)**

Definition: `function(A)` → `A`, where `A` is a set

```python
def get_age(player_birth_date):
    return ((date.today() - player_birth_date)
        // timedelta(days=365.2425))
```

```python
(df_bvb_player.player_dob.
apply(get_player_dob_date))
```

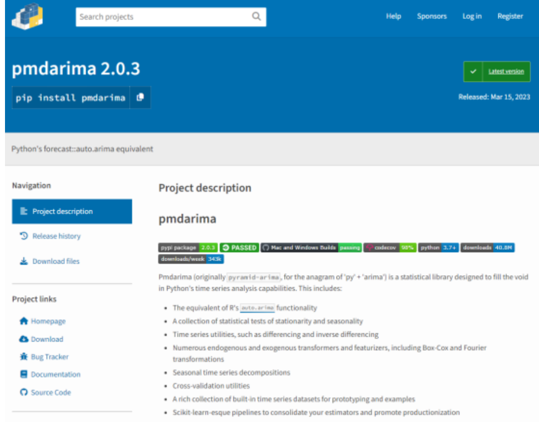
0	1997-12-06
1	2001-05-25
2	1991-04-13
3	2003-11-19
4	1999-12-01
5	1995-09-03
6	1988-12-16
7	2003-10-14
8	1999-09-10
9	1995-04-16
10	1997-11-17
11	1995-05-27
12	1991-09-12
13	2000-03-02
14	1994-01-12
15	1998-01-11

=

```python
(df_bvb_player.player_dob.
 apply(get_player_dob_date).
 apply(get_age))
```

0	25
1	22
2	32
3	19
4	23
5	28
6	34
7	19
8	24
9	28
10	25
11	28
12	32
13	23
14	29
15	25

`.apply` also works on DataFrames. We will see later how.



https://pypi.org/project/pmdarima/

Caution: time dependent data

Time series data is a **sequence** of **data points indexed** in **time order**.

Ordering is typically over **equally spaced time** intervals, such as minutes, hours, days, months, or years.

Analysis of such data is a (magical) domain for itself.

Analysis requires specific modeling strategies e.g., moving average or exponential smoothing, and domain expertise.

In the simple case of dates:

→ Always store the **raw date format,** it gives you more flexibility to run analytics.

https://jakevdp.github.io/PythonDataScienceHandbook/03.11-working-with-time-series.html

```
df_bvb_player.player_dob
```

```
0     06.12.1997 (25)
1     25.05.2001 (22)
2     13.04.1991 (32)
3     19.11.2003 (19)
4     01.12.1999 (23)
5     03.09.1995 (27)
6     16.12.1988 (34)
7     14.10.2003 (19)
8     10.09.1999 (23)
9     16.04.1995 (28)
10    17.11.1997 (25)
11    27.05.1995 (28)
12    12.09.1991 (31)
13    02.03.2000 (23)
14    12.01.1994 (29)
15    11.01.1998 (25)
```

≠

```
(df_bvb_player.player_dob.
 apply(get_player_dob_date).
 apply(get_age))
```

```
0     25
1     22
2     32
3     19
4     23
5     28
6     34
7     19
8     24
9     28
10    25
11    28
12    32
13    23
14    29
15    25
```

Data Cleaning Summary

Depending on the dataset, preprocessing function, and your style, you can:

- Overwrite attributes

```
df_bvb_player["player_value"] = df_bvb_player.player_value.apply(get_player_value_numeric)
df_bvb_player["player_dob"] = df_bvb_player.player_dob.apply(get_player_dob_date)
```

- Create new attributes

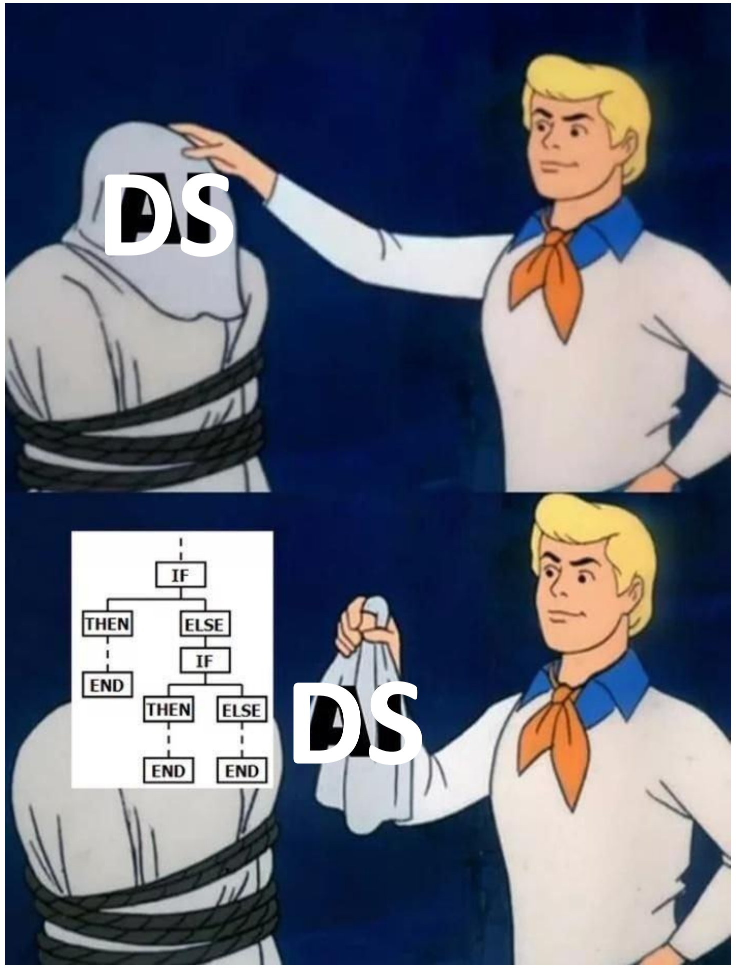
```
df_bvb_player["player_number_even"] = df_bvb_player.player_number.apply(player_number_even)
df_bvb_player["age"] = df_bvb_player.player_dob.apply(get_age)
```

- Reduce (delete) attributes and records.

```
df_bvb_player.head()
```

player_name	player_dob	player_country	player_value	player_number_even	age
Gregor Kobel	1997-12-06	Schweiz	35000000.0	False	25
Marcel Lotka	2001-05-25	Deutschland	1500000.0	False	22

You are becoming a Data Scientist!



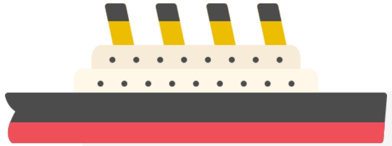
Getting to know your data (Recap)

Simple visualization tools and summary statistics are very useful!

- Make some plots.
- Calculate summary statistics.

...and think:

- Is the distribution consistent with the background knowledge? (You may need to consult domain experts)
- Any obvious outliers?
- Are some attributes heavily correlated with each other?



	Age	Fare
count	714.000000	891.000000
mean	29.699118	32.204208
std	14.526497	49.693429
min	0.420000	0.000000
25%	20.125000	7.910400
50%	28.000000	14.454200
75%	38.000000	31.000000
max	80.000000	512.329200

Initial **Data Exploration**

Once attributes have the **desireable data type**,

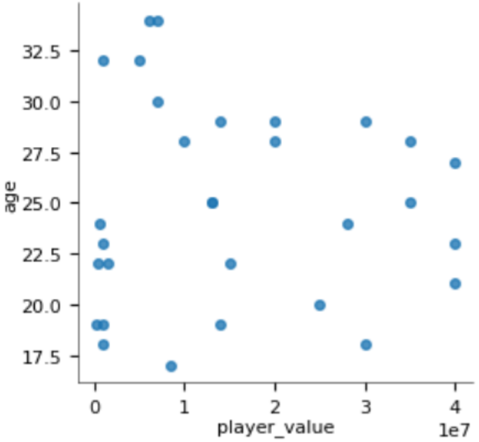
particularly **numeric** ones, analysis becomes more fun!

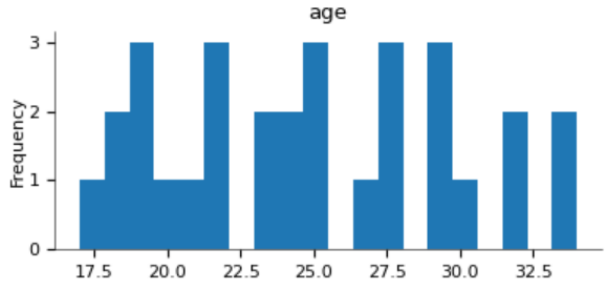
```
df_bvb_player.info()
```

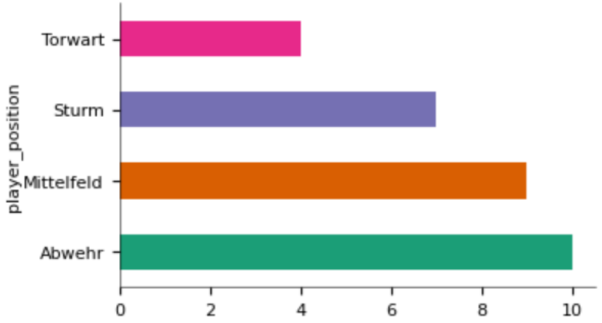
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   club_name           30 non-null     object
 1   club_league         30 non-null     object
 2   player_position     30 non-null     object
 3   player_number       30 non-null     int64
 4   player_name         30 non-null     object
 5   player_dob          30 non-null     object
 6   player_country      30 non-null     object
 7   player_value        30 non-null     float64
 8   player_number_even  30 non-null     bool
 9   age                 30 non-null     int64
dtypes: bool(1), float64(1), int64(2), object(6)
memory usage: 2.3+ KB
```

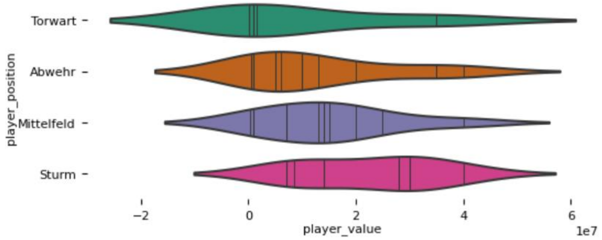
```
df_bvb_player.describe()
```

	player_number	player_value	age
count	30.000000	3.000000e+01	30.000000
mean	20.300000	1.540500e+07	24.866667
std	12.809345	1.368692e+07	4.953113
min	1.000000	1.500000e+05	17.000000
25%	9.250000	2.375000e+06	21.250000
50%	19.500000	1.300000e+07	24.500000
75%	29.250000	2.725000e+07	28.750000
max	47.000000	4.000000e+07	34.000000







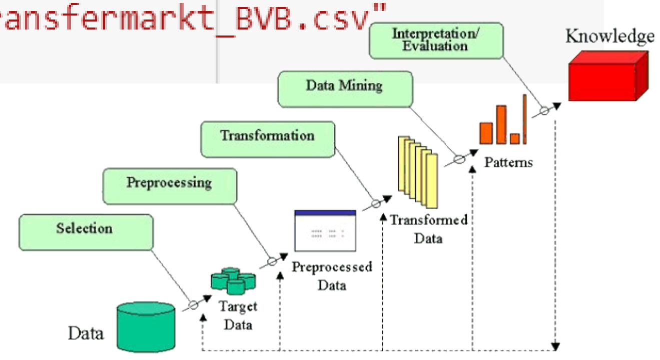


Reaching Data Science Milestone

If you have reached a significant milestone in a Data Science Life Cycle Framework:

- Summarize development in a few sentences (documentation!).
- Verify literature, whether all foreseen processes of the milestone have been incorporated by your team.
- Export the DataFrame to a file to start with in the next incremental component, e.g., *"2.0-Data-Preprocessing-Transfermarkt_BVB.csv"* (Versioning)

```
url = r"/content/drive/MyDrive/WS23_24 PhD/FHDTM-P2DS-WS2324/Data Science Projekt Demo/Datensätze/"
file_name = "FHDTM-P2DS-WS2324-Project-Demo-2.0-Data-Preprocessing-Transfermarkt_BVB.csv"
df_bvb_player.to_csv(url+file_name, index=False)
```



KDD, Fayyad et al., 1996: https://www2.cs.uregina.ca/~dbd/cs831/notes/kdd/1_kdd.html

Training #1

Open a blank .ipynb file and import the .csv file

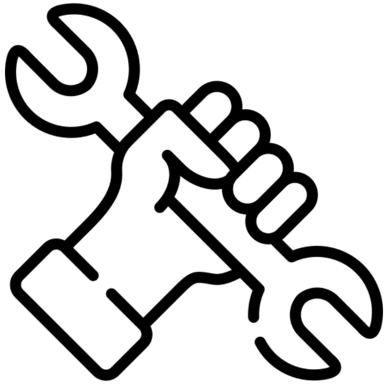
```
url = "https://raw.githubusercontent.com/leotraeg/FHDTM-P2DS-WS2324/main/Data%20Science%20Projekt%20Demo/Datens%C3%A4tze/FHDTM-P2DS-WS2324-Project-Demo-2.0-Data-Preprocessing-Transfermarkt_BVB.csv"
```

as a pandas data frame `pd.read_csv(url)`.

We will work with Hanna in the next slides!

1. “Hanna Muster” joined the BVB club this season. You can add a new record using a `dictionary { }` and `pd.Series` object by `df.loc[len(df)] = pd.Series(data=dictionary)`. Assign the unknown attributes with numpy’s `np.NaN` value.
2. Raise the player_value attribute of the DataFrame for all attackers (“Sturm”) by 15%.
3. Add a new attribute player_talent to the DataFrame in which
   - Players with player_value greater than 10 mil. = “Star”
   - Players aged under and including 21 with player_value greater than 1 mil. = “Rising Star”.

Training #1 (cont.)



What BVB players are star players?

```
df_bvb_player[df_bvb_player.player_talent.str.contains('Star')]
```

index	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value	player_number_even	age	player_talent
0	Borussia Dortmund	Bundesliga	Torwart	1.0	Gregor Kobel	1997-12-06	Schweiz	35000000.0	0.0	25	Star
4	Borussia Dortmund	Bundesliga	Abwehr	4.0	Nico Schlotterbeck	1999-12-01	Deutschland	40000000.0	1.0	23	Star
5	Borussia Dortmund	Bundesliga	Abwehr	25.0	Niklas Süle	1995-09-03	Deutschland	35000000.0	0.0	28	Star
9	Borussia Dortmund	Bundesliga	Abwehr	5.0	Ramy Bensebaini	1995-04-16	Algerien	20000000.0	0.0	28	Star
10	Borussia Dortmund	Bundesliga	Abwehr	26.0	Julian Ryerson	1997-11-17	Norwegen	13000000.0	1.0	25	Star
14	Borussia Dortmund	Bundesliga	Mittelfeld	23.0	Emre Can	1994-01-12	Deutschland	14000000.0	0.0	29	Star
15	Borussia Dortmund	Bundesliga	Mittelfeld	6.0	Salih Özcan	1998-01-11	Türkei	13000000.0	1.0	25	Star
17	Borussia Dortmund	Bundesliga	Mittelfeld	20.0	Marcel Sabitzer	1994-03-17	Österreich	20000000.0	1.0	29	Star
18	Borussia Dortmund	Bundesliga	Mittelfeld	8.0	Felix Nmecha	2000-10-10	Deutschland	15000000.0	1.0	22	Star
20	Borussia Dortmund	Bundesliga	Mittelfeld	19.0	Julian Brandt	1996-05-02	Deutschland	40000000.0	0.0	27	Star
21	Borussia Dortmund	Bundesliga	Mittelfeld	7.0	Giovanni Reyna	2002-11-13	Vereinigte Staaten	25000000.0	0.0	20	Rising Star
23	Borussia Dortmund	Bundesliga	Sturm	27.0	Karim Adeyemi	2002-01-18	Deutschland	46000000.0	0.0	21	Rising Star
24	Borussia Dortmund	Bundesliga	Sturm	43.0	Jamie Bynoe-Gittens	2004-08-08	England	16099999.999999998	0.0	19	Rising Star
26	Borussia Dortmund	Bundesliga	Sturm	21.0	Donyell Malen	1999-01-19	Niederlande	32199999.999999996	0.0	24	Star

…and how many more?

Break

Major Tasks in **Data Preprocessing**

Data Reduction

- Obtains reduced representation in volume but produces the same or similar analytical results.

Data Cleaning

- **Fill in missing values**, smooth noisy data, identify or remove outliers, and resolve inconsistencies caused by data integration.

Data Integration

- Integration of multiple tables, databases, data cubes, or files.

Data Transformation

- Aggregation, generalization, normalization and attribute construction.

Why is Data Missing?

The real-world data is rarely clean and homogeneous!

Incomplete data

- “Not applicable“ data value when collected.
- Different considerations between the time
  when data was collected versus *used for analysis.*
- Human/hardware/software problems.

Inconsistent data

- Different data sources.
- Functional dependency violation (e.g., modify some linked data).
- Duplicate records/attributes during integration process.

Why is Data Missing? (cont.)

Noisy data (incorrect values)

- Faulty data collection instruments.
- Human or computer error at data entry.
- Errors in data transmission.

Data is not always available

- Considered unimportant (at point of entry) and not recorded or deleted.
- Not entered due to misunderstanding.
- Overwritten instead of historically registered changes.

How to **annotate** Missing Data?

Function name	NaN-safe version	Description
np.sum	np.nansum	Compute sum of elements
np.prod	np.nanprod	Compute product of elements
np.mean	np.nanmean	Compute mean of elements
np.std	np.nanstd	Compute standard deviation
np.var	np.nanvar	Compute variance
np.min	np.nanmin	Find minimum value
np.max	np.nanmax	Find maximum value
np.argmin	np.nanargmin	Find index of minimum value
np.argmax	np.nanargmax	Find index of maximum value
np.median	np.nanmedian	Compute median of elements
np.percentile	np.nanpercentile	Compute rank-based statistics of elements
np.any	N/A	Evaluate whether any elements are true
np.all	N/A	Evaluate whether all elements are true

VanderPlas, J., "Python Data Science Handbook", O'Reilly, 2017

Python chose to use two already-existing Python null values:

- Special floating-point `numpy.NaN` value
  - Acronym for Not a Number
  - Of `dtype('float64')`
  - Any arithmetic operation such as `sum()` will throw another `numpy.NaN`.
- Python `None` object
  - Only usable for Series of type `'object'`.
  - Any arithmetic operation such as `sum()` will throw error.

NaN and **None both have** their **place** in **Pandas**!

Pandas treats them as essentially interchangeable for indicating missing or null values.

How to **find** Missing Data?

There are several useful methods for detecting, removing, and replacing null values in Pandas data structures

- `isnull()`: Generate a `boolean` mask indicating missing values.
- `notnull()`: Opposite of `isnull()`.
- `dropna()`: Return a filtered version of the data.
- `fillna()`: Return a copy of the data with missing values filled or imputed.

How to **find** Missing Data? (cont.)

```
df_bvb_player.isnull()
```

1 to 31 of 31 entries Filter

index	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value	player_number_even	age	player_talent
0	false	false	false	false	false	false	false	false	false	false	false
1	false	false	false	false	false	false	false	false	false	false	true
2	false	false	false	false	false	false	false	false	false	false	true
3	false	false	false	false	false	false	false	false	false	false	true
4	false	false	false	false	false	false	false	false	false	false	false
5	false	false	false	false	false	false	false	false	false	false	false
6	false	false	false	false	false	false	false	false	false	false	true
7	false	false	false	false	false	false	false	false	false	false	true
8	false	false	false	false	false	false	false	false	false	false	true
9	false	false	false	false	false	false	false	false	false	false	false
10	false	false	false	false	false	false	false	false	false	false	false
11	false	false	false	false	false	false	false	false	false	false	true
12	false	false	false	false	false	false	false	false	false	false	true
13	false	false	false	false	false	false	false	false	false	false	true
14	false	false	false	false	false	false	false	false	false	false	false
15	false	false	false	false	false	false	false	false	false	false	false
16	false	false	false	false	false	false	false	false	false	false	true
17	false	false	false	false	false	false	false	false	false	false	false
18	false	false	false	false	false	false	false	false	false	false	false
19	false	false	false	false	false	false	false	false	false	false	true
20	false	false	false	false	false	false	false	false	false	false	false
21	false	false	false	false	false	false	false	false	false	false	false
22	false	false	false	false	false	false	false	false	false	false	true
23	false	false	false	false	false	false	false	false	false	false	false
24	false	false	false	false	false	false	false	false	false	false	false
25	false	false	false	false	false	false	false	false	false	false	true
26	false	false	false	false	false	false	false	false	false	false	false
27	false	false	false	false	false	false	false	false	false	false	false
28	false	false	false	false	false	false	false	false	false	false	false
29	false	false	false	false	false	false	false	false	false	false	false
30	false	false	true	true	false	false	false	true	true	false	true

How to **find** Missing Data? (cont.)

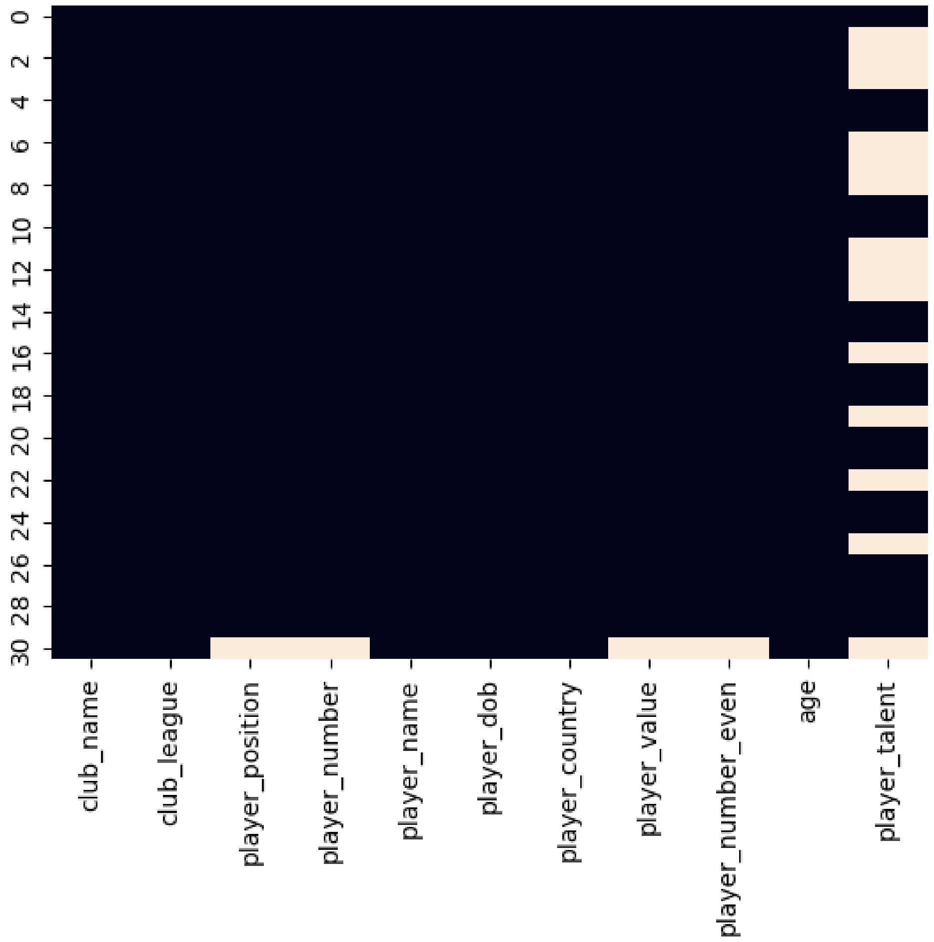
For large data sets, prefer

- Graphical visualizations or
- Summary statistics

about NaN or None values.

```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
plt.figure()
sns.heatmap((df_bvb_player.isnull()), cbar=False)
```



Data Cleaning: **fill in missing values** ignore the records with null

```
df_bvb_player.dropna()
```

1 to 17 of 17 entries Filter

index	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value	player_number_even	age	player_talent
0	Borussia Dortmund	Bundesliga	Torwart	1.0	Gregor Kobel	1997-12-06	Schweiz	35000000.0	0.0	25	Star
4	Borussia Dortmund	Bundesliga	Abwehr	4.0	Nico Schlotterbeck	1999-12-01	Deutschland	40000000.0	1.0	23	Star
5	Borussia Dortmund	Bundesliga	Abwehr	25.0	Niklas Süle	1995-09-03	Deutschland	35000000.0	0.0	28	Star
9	Borussia Dortmund	Bundesliga	Abwehr	5.0	Ramy Bensebaini	1995-04-16	Algerien	20000000.0	0.0	28	Star
10	Borussia Dortmund	Bundesliga	Abwehr	26.0	Julian Ryerson	1997-11-17	Norwegen	13000000.0	1.0	25	Star
14	Borussia Dortmund	Bundesliga	Mittelfeld	23.0	Emre Can	1994-01-12	Deutschland	14000000.0	0.0	29	Star
15	Borussia Dortmund	Bundesliga	Mittelfeld	6.0	Salih Özcan	1998-01-11	Türkei	13000000.0	1.0	25	Star
17	Borussia Dortmund	Bundesliga	Mittelfeld	20.0	Marcel Sabitzer	1994-03-17	Österreich	20000000.0	1.0	29	Star
18	Borussia Dortmund	Bundesliga	Mittelfeld	8.0	Felix Nmecha	2000-10-10	Deutschland	15000000.0	1.0	22	Star
20	Borussia Dortmund	Bundesliga	Mittelfeld	19.0	Julian Brandt	1996-05-02	Deutschland	40000000.0	0.0	27	Star
21	Borussia Dortmund	Bundesliga	Mittelfeld	7.0	Giovanni Reyna	2002-11-13	Vereinigte Staaten	25000000.0	0.0	20	Rising Star
23	Borussia Dortmund	Bundesliga	Sturm	27.0	Karim Adeyemi	2002-01-18	Deutschland	46000000.0	0.0	21	Rising Star
24	Borussia Dortmund	Bundesliga	Sturm	43.0	Jamie Bynoe-Gittens	2004-08-08	England	16099999.999999998	0.0	19	Rising Star
26	Borussia Dortmund	Bundesliga	Sturm	21.0	Donyell Malen	1999-01-19	Niederlande	32200000.0	0.0	24	Star
27	Borussia Dortmund	Bundesliga	Sturm	16.0	Julien Duranville	2006-05-05	Belgien	9775000.0	1.0	17	Rising Star
28	Borussia Dortmund	Bundesliga	Sturm	9.0	Sébastien Haller	1994-06-22	Elfenbeinküste	34500000.0	0.0	29	Star
29	Borussia Dortmund	Bundesliga	Sturm	18.0	Youssoufa Moukoko	2004-11-20	Deutschland	34500000.0	1.0	18	Rising Star

Not effective when % of missing values per attribute varies considerably!

Data Cleaning: **fill in missing values with some value**

Works also on Series

```
df_bvb_player.fillna("No Category")
```

1 to 31 of 31 entries Filter

index	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value	player_number_even	age	player_talent
0	Borussia Dortmund	Bundesliga	Torwart	1	Gregor Kobel	1997-12-06	Schweiz	35000000	0	25	Star
1	Borussia Dortmund	Bundesliga	Torwart	35	Marcel Lotka	2001-05-25	Deutschland	1500000	0	22	No Category
2	Borussia Dortmund	Bundesliga	Torwart	33	Alexander Meyer	1991-04-13	Deutschland	1000000	0	32	No Category
3	Borussia Dortmund	Bundesliga	Torwart	31	Silas Ostrzinski	2003-11-19	Deutschland	150000	0	19	No Category
4	Borussia Dortmund	Bundesliga	Abwehr	4	Nico Schlotterbeck	1999-12-01	Deutschland	40000000	1	23	Star
5	Borussia Dortmund	Bundesliga	Abwehr	25	Niklas Süle	1995-09-03	Deutschland	35000000	0	28	Star
6	Borussia Dortmund	Bundesliga	Abwehr	15	Mats Hummels	1988-12-16	Deutschland	6000000	0	34	No Category
7	Borussia Dortmund	Bundesliga	Abwehr	44	Soumaïla Coulibaly	2003-10-14	Frankreich	1000000	1	19	No Category
8	Borussia Dortmund	Bundesliga	Abwehr	47	Antonios Papadopoulos	1999-09-10	Deutschland	600000	0	24	No Category
9	Borussia Dortmund	Bundesliga	Abwehr	5	Ramy Bensebaini	1995-04-16	Algerien	20000000	0	28	Star
10	Borussia Dortmund	Bundesliga	Abwehr	26	Julian Ryerson	1997-11-17	Norwegen	13000000	1	25	Star
11	Borussia Dortmund	Bundesliga	Abwehr	17	Marius Wolf	1995-05-27	Deutschland	10000000	0	28	No Category
12	Borussia Dortmund	Bundesliga	Abwehr	24	Thomas Meunier	1991-09-12	Belgien	5000000	1	32	No Category
13	Borussia Dortmund	Bundesliga	Abwehr	2	Mateu Morey Bauzà	2000-03-02	Spanien	1000000	1	23	No Category
14	Borussia Dortmund	Bundesliga	Mittelfeld	23	Emre Can	1994-01-12	Deutschland	14000000	0	29	Star
15	Borussia Dortmund	Bundesliga	Mittelfeld	6	Salih Özcan	1998-01-11	Türkei	13000000	1	25	Star
16	Borussia Dortmund	Bundesliga	Mittelfeld	32	Abdoulaye Kamara	2004-11-06	Frankreich	1000000	1	18	No Category
17	Borussia Dortmund	Bundesliga	Mittelfeld	20	Marcel Sabitzer	1994-03-17	Österreich	20000000	1	29	Star
18	Borussia Dortmund	Bundesliga	Mittelfeld	8	Felix Nmecha	2000-10-10	Deutschland	15000000	1	22	Star
19	Borussia Dortmund	Bundesliga	Mittelfeld	30	Ole Pohlmann	2001-04-05	Deutschland	400000	1	22	No Category
20	Borussia Dortmund	Bundesliga	Mittelfeld	19	Julian Brandt	1996-05-02	Deutschland	40000000	0	27	Star
21	Borussia Dortmund	Bundesliga	Mittelfeld	7	Giovanni Reyna	2002-11-13	Vereinigte Staaten	25000000	0	20	Rising Star
22	Borussia Dortmund	Bundesliga	Mittelfeld	11	Marco Reus	1989-05-31	Deutschland	7000000	0	34	No Category
23	Borussia Dortmund	Bundesliga	Sturm	27	Karim Adeyemi	2002-01-18	Deutschland	46000000	0	21	Rising Star
24	Borussia Dortmund	Bundesliga	Sturm	43	Jamie Bynoe-Gittens	2004-08-08	England	16099999.999999998	0	19	Rising Star
25	Borussia Dortmund	Bundesliga	Sturm	10	Thorgan Hazard	1993-03-29	Belgien	8049999.999999999	1	30	No Category
26	Borussia Dortmund	Bundesliga	Sturm	21	Donyell Malen	1999-01-19	Niederlande	32200000	0	24	Star
27	Borussia Dortmund	Bundesliga	Sturm	16	Julien Duranville	2006-05-05	Belgien	9775000	1	17	Rising Star
28	Borussia Dortmund	Bundesliga	Sturm	9	Sébastien Haller	1994-06-22	Elfenbeinküste	34500000	0	29	Star
29	Borussia Dortmund	Bundesliga	Sturm	18	Youssoufa Moukoko	2004-11-20	Deutschland	34500000	1	18	Rising Star
30	Borussia Dortmund	Bundesliga	No Category	No Category	Hanna Muster	2000-07-17	Deutschland	No Category	No Category	23	No Category

```
df_bvb_player.player_talent.fillna("No Category")
```

```
0            Star
1     No Category
2     No Category
3     No Category
4            Star
5            Star
6     No Category
7     No Category
8     No Category
9            Star
10           Star
11    No Category
12    No Category
13    No Category
14           Star
15           Star
16    No Category
17           Star
18           Star
19    No Category
20           Star
21    Rising Star
22    No Category
23    Rising Star
24    Rising Star
25    No Category
26           Star
27    Rising Star
28           Star
29    Rising Star
30    No Category
Name: player_talent, dtype: object
```

Data Cleaning: **fill in missing values** with forward-fill or back-fill method

```
df_bvb_player.player_talent
```

```
0            Star
1             NaN
2             NaN
3             NaN
4            Star
5            Star
6             NaN
7             NaN
8             NaN
9            Star
10           Star
11            NaN
12            NaN
13            NaN
14           Star
15           Star
16            NaN
17           Star
18           Star
19            NaN
20           Star
21    Rising Star
22            NaN
23    Rising Star
24    Rising Star
25            NaN
26           Star
27    Rising Star
28           Star
29    Rising Star
30            NaN
Name: player_talent, dtype: object
```

```
df_bvb_player.player_talent.fillna(method="ffill")
```

```
0            Star
1            Star
2            Star
3            Star
4            Star
5            Star
6            Star
7            Star
8            Star
9            Star
10           Star
11           Star
12           Star
13           Star
14           Star
15           Star
16           Star
17           Star
18           Star
19           Star
20           Star
21    Rising Star
22    Rising Star
23    Rising Star
24    Rising Star
25    Rising Star
26           Star
27    Rising Star
28           Star
29    Rising Star
30    Rising Star
Name: player_talent, dtype: object
```

```
df_bvb_player.player_talent.fillna(method="bfill")
```

```
0            Star
1            Star
2            Star
3            Star
4            Star
5            Star
6            Star
7            Star
8            Star
9            Star
10           Star
11           Star
12           Star
13           Star
14           Star
15           Star
16           Star
17           Star
18           Star
19           Star
20           Star
21    Rising Star
22    Rising Star
23    Rising Star
24    Rising Star
25           Star
26           Star
27    Rising Star
28           Star
29    Rising Star
30            NaN
Name: player_talent, dtype: object
```

How to **handle** Missing Data?

Strategy	**Recommendation**
Ignore (delete) records / attributes.	If majority of record / attribute’s values are missing. *Caution: deletion generally introduces bias!*
Fill in the missing values manually.	Most effective but also tedious and sometimes infeasible approach.
Fill in the missing values automatically with	
Tools.	Search providers, LLMs, and API calls can help in large scales.
A global constant, e.g., “unknown”.	Effective when NaN or None has some meaning.
The attribute mean.	Rarely effective but not a NO-NO.
The attribute mean of most similar group.	Better.
The most probable value (Bayesian, ML).	State of the art.

Major Tasks in **Data Preprocessing**

Data Reduction

- Obtains reduced representation in volume but produces the same or similar analytical results.

Data Cleaning

- Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies caused by data integration.

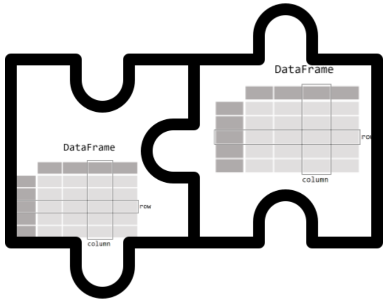
Data Integration

- **Integration of multiple tables**, databases, data cubes, or files.

Data Transformation

- Aggregation, generalization, normalization and attribute construction.

Data Integration



Combines data from multiple sources into a coherent store.

Integration problems (scientific domain called Entity Resolution):

- **Data type** conflicts, e.g., date of birth stored as date and string.
- **Labeling** conflicts, e.g., attribute customer id and client id.
- **Structure** conflicts: different normalization or cardinality, e.g., for grades.
- **Naming** conflicts, e.g., Mercedes Benz and Daimler.
- **Domain** conflicts, e.g., table product of car manufacturer and wheel supplier.

...and a few more.

Data Integration (cont.)

Key task is handling redundancy by identifying

object A ≡ object B ≡ real world entity

both on attribute and instance level.

- Remove duplicates
- Union (concat) DataFrames
- Join (merge) Data Frames

Careful integration of the data form multiple sources may help **reduce / avoid redundancies** and inconsistencies. It also improves quality of data analysis.

Data Integration/Cleaning: **remove duplicates**

```
df_top_100_clubs
```

1 to 100 of 2942 entries Filter

index	club_name	club_number_player	club_avg_age	club_league	club_number_foreign_players	club_number_national_players	club_stadium	club_stadium_seats	club_current_transfer_balance
0	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
1	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
2	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
3	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
4	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
5	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
6	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
7	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
8	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
9	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
10	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
11	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
12	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
13	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
14	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
15	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
16	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
17	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
18	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
19	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
20	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
21	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
22	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
23	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
24	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
25	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
26	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
27	FC Arsenal	33	24,7	Premier League	24	22	Emirates Stadium	60.704 Plätze	-197,75 Mio. €
28	FC Arsenal	33	24,7	Premier League	24	22	Emirates Stadium	60.704 Plätze	-197,75 Mio. €
29	FC Arsenal	33	24,7	Premier League	24	22	Emirates Stadium	60.704 Plätze	-197,75 Mio. €
30	FC Arsenal	33	24,7	Premier League	24	22	Emirates Stadium	60.704 Plätze	-197,75 Mio. €

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop_duplicates.html

Return DataFrame with duplicate (identical) rows removed.

Data Integration/Cleaning: remove duplicates (cont.)

```
df_top_100_clubs.drop_duplicates()
```

1 to 25 of 100 entries Filter

index	club_name	club_number_player	club_avg_age	club_league	club_number_foreign_players	club_number_national_players	club_stadium	club_stadium_seats	club_current_transfer_balance
0	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
27	FC Arsenal	33	24,7	Premier League	24	22	Emirates Stadium	60.704 Plätze	-197,75 Mio. €
60	FC Paris Saint-Germain	39	25,3	Ligue 1	27	22	Parc des Princes	49.691 Plätze	-128,00 Mio. €
99	Real Madrid	24	26,7	LaLiga	16	19	Santiago Bernabéu	81.044 Plätze	-124,50 Mio. €
123	FC Chelsea	30	23,8	Premier League	19	18	Stamford Bridge	40.853 Plätze	+46,90 Mio. €
153	FC Bayern München	27	25,3	Bundesliga	16	20	Allianz Arena	75.024 Plätze	+51,75 Mio. €
180	Manchester United	36	25,2	Premier League	23	22	Old Trafford	74.879 Plätze	-168,60 Mio. €
216	FC Barcelona	23	25,8	LaLiga	12	18	Spotify Camp Nou	99.354 Plätze	+44,50 Mio. €
239	Tottenham Hotspur	35	25,5	Premier League	24	21	Tottenham Hotspur Stadium	62.062 Plätze	-182,00 Mio. €
274	FC Liverpool	22	26,3	Premier League	17	14	Anfield	54.074 Plätze	-51,30 Mio. €
296	Newcastle United	30	27,6	Premier League	16	13	St James' Park	52.338 Plätze	-108,60 Mio. €
326	Aston Villa	28	26,3	Premier League	18	13	Villa Park	42.682 Plätze	-85,10 Mio. €
354	AC Mailand	32	25,8	Serie A	25	17	Giuseppe Meazza	75.923 Plätze	-46,50 Mio. €
386	SSC Neapel	30	25,5	Serie A	18	17	Stadio Diego Armando Maradona	54.726 Plätze	+7,50 Mio. €
416	Atlético Madrid	29	27,6	LaLiga	17	17	Civitas Metropolitano	67.829 Plätze	+43,30 Mio. €
445	Juventus Turin	35	25,7	Serie A	19	17	Allianz Stadium	41.507 Plätze	-41,60 Mio. €
480	Inter Mailand	26	26,7	Serie A	15	16	Giuseppe Meazza	75.923 Plätze	+74,00 Mio. €
506	Borussia Dortmund	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
536	Bayer 04 Leverkusen	26	24,2	Bundesliga	19	14	BayArena	30.210 Plätze	+16,20 Mio. €
562	Brighton & Hove Albion	30	25,0	Premier League	22	15	AMEX Stadium	31.800 Plätze	-2,75 Mio. €
592	FC Brentford	28	25,6	Premier League	21	16	Brentford Community Stadium	17.250 Plätze	-58,35 Mio. €
620	RasenBallsport Leipzig	26	24,7	Bundesliga	19	17	Red Bull Arena	47.069 Plätze	+114,70 Mio. €
646	Real Sociedad San Sebastián	27	24,7	LaLiga	5	7	Reale Arena	39.313 Plätze	+6,30 Mio. €

Data Integration/Cleaning: remove duplicates (cont.)

```
df_top_100_clubs.drop_duplicates().reset_index(drop=True)
```

1 to 25 of 100 entries Filter

index	club_name	club_number_player	club_avg_age	club_league	club_number_foreign_players	club_number_national_players	club_stadium	club_stadium_seats	club_current_transfer_balance
0	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
1	FC Arsenal	33	24,7	Premier League	24	22	Emirates Stadium	60.704 Plätze	-197,75 Mio. €
2	FC Paris Saint-Germain	39	25,3	Ligue 1	27	22	Parc des Princes	49.691 Plätze	-128,00 Mio. €
3	Real Madrid	24	26,7	LaLiga	16	19	Santiago Bernabéu	81.044 Plätze	-124,50 Mio. €
4	FC Chelsea	30	23,8	Premier League	19	18	Stamford Bridge	40.853 Plätze	+46,90 Mio. €
5	FC Bayern München	27	25,3	Bundesliga	16	20	Allianz Arena	75.024 Plätze	+51,75 Mio. €
6	Manchester United	36	25,2	Premier League	23	22	Old Trafford	74.879 Plätze	-168,60 Mio. €
7	FC Barcelona	23	25,8	LaLiga	12	18	Spotify Camp Nou	99.354 Plätze	+44,50 Mio. €
8	Tottenham Hotspur	35	25,5	Premier League	24	21	Tottenham Hotspur Stadium	62.062 Plätze	-182,00 Mio. €
9	FC Liverpool	22	26,3	Premier League	17	14	Anfield	54.074 Plätze	-51,30 Mio. €
10	Newcastle United	30	27,6	Premier League	16	13	St James' Park	52.338 Plätze	-108,60 Mio. €
11	Aston Villa	28	26,3	Premier League	18	13	Villa Park	42.682 Plätze	-85,10 Mio. €
12	AC Mailand	32	25,8	Serie A	25	17	Giuseppe Meazza	75.923 Plätze	-46,50 Mio. €
13	SSC Neapel	30	25,5	Serie A	18	17	Stadio Diego Armando Maradona	54.726 Plätze	+7,50 Mio. €
14	Atlético Madrid	29	27,6	LaLiga	17	17	Civitas Metropolitano	67.829 Plätze	+43,30 Mio. €
15	Juventus Turin	35	25,7	Serie A	19	17	Allianz Stadium	41.507 Plätze	-41,60 Mio. €
16	Inter Mailand	26	26,7	Serie A	15	16	Giuseppe Meazza	75.923 Plätze	+74,00 Mio. €
17	Borussia Dortmund	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
18	Bayer 04 Leverkusen	26	24,2	Bundesliga	19	14	BayArena	30.210 Plätze	+16,20 Mio. €
19	Brighton & Hove Albion	30	25,0	Premier League	22	15	AMEX Stadium	31.800 Plätze	-2,75 Mio. €
20	FC Brentford	28	25,6	Premier League	21	16	Brentford Community Stadium	17.250 Plätze	-58,35 Mio. €
21	RasenBallsport Leipzig	26	24,7	Bundesliga	19	17	Red Bull Arena	47.069 Plätze	+114,70 Mio. €
22	Real Sociedad San Sebastián	27	24,7	LaLiga	5	7	Reale Arena	39.313 Plätze	+6,30 Mio. €

Data Integration: **concat** ∪ ...but also many more

Concatenate pandas objects (DataFrames and Series) along a particular axis.

Returns a **new DataFrame** consisting of the **rows of *all objects*** *in a* ***list***.

Why use `pd.concat([df_a, df_b])`?

- It is not recommended to build DataFrames by adding single rows in a for loop.
- Why? Computation is inefficient!
- Use parameter setting `ignore_index=True` if indices are meaningless.
- `Concat` has various kinds of set logic for the indexes, integrity checks, hierarchy ladders, and relational algebra functionality…

https://pandas.pydata.org/pandas-docs/stable/user_guide/merging.html

Data Integration: **concat** (cont.)

```python
pd.concat([df_psg_player, df_bvb_player], ignore_index=True)
```

```
df_bvb_player
```

1 to 25 of 31 entries Filter

index	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value	player_number_even	age	player_talent
0	Borussia Dortmund	Bundesliga	Torwart	1.0	Gregor Kobel	1997-12-06	Schweiz	35000000.0	0.0	25	Star
1	Borussia Dortmund	Bundesliga	Torwart	35.0	Marcel Lotka	2001-05-25	Deutschland	1500000.0	0.0	22	NaN
2	Borussia Dortmund	Bundesliga	Torwart	33.0	Alexander Meyer	1991-04-13	Deutschland	1000000.0	0.0	32	NaN
3	Borussia Dortmund	Bundesliga	Torwart	31.0	Silas Ostrzinski	2003-11-19	Deutschland	150000.0	0.0	19	NaN
4	Borussia Dortmund	Bundesliga	Abwehr	4.0	Nico Schlotterbeck	1999-12-01	Deutschland	40000000.0	1.0	23	Star
5	Borussia Dortmund	Bundesliga	Abwehr	25.0	Niklas Süle	1995-09-03	Deutschland	35000000.0	0.0	28	Star
6	Borussia Dortmund	Bundesliga	Abwehr	15.0	Mats Hummels	1988-12-16	Deutschland	6000000.0	0.0	34	NaN
7	Borussia Dortmund	Bundesliga	Abwehr	44.0	Soumaïla Coulibaly	2003-10-14	Frankreich	1000000.0	1.0	19	NaN
8	Borussia Dortmund	Bundesliga	Abwehr	47.0	Antonios Papadopoulos	1999-09-10	Deutschland	600000.0	0.0	24	NaN
9	Borussia Dortmund	Bundesliga	Abwehr	5.0	Ramy Bensebaini	1995-04-16	Algerien	20000000.0	0.0	28	Star
10	Borussia Dortmund	Bundesliga	Abwehr	26.0	Julian Ryerson	1997-11-17	Norwegen	13000000.0	1.0	25	Star
11	Borussia Dortmund	Bundesliga	Abwehr	17.0	Marius Wolf	1995-05-27	Deutschland	10000000.0	0.0	28	NaN
12	Borussia Dortmund	Bundesliga	Abwehr	24.0	Thomas Meunier	1991-09-12	Belgien	5000000.0	1.0	32	NaN
13	Borussia Dortmund	Bundesliga	Abwehr	2.0	Mateu Morey Bauzà	2000-03-02	Spanien	1000000.0	1.0	23	NaN
14	Borussia Dortmund	Bundesliga	Mittelfeld	23.0	Emre Can	1994-01-12	Deutschland	14000000.0	0.0	29	Star
15	Borussia Dortmund	Bundesliga	Mittelfeld	6.0	Salih Özcan	1998-01-11	Türkei	13000000.0	1.0	25	Star
16	Borussia Dortmund	Bundesliga	Mittelfeld	32.0	Abdoulaye Kamara	2004-11-06	Frankreich	1000000.0	1.0	18	NaN
17	Borussia Dortmund	Bundesliga	Mittelfeld	20.0	Marcel Sabitzer	1994-03-17	Österreich	20000000.0	1.0	29	Star
18	Borussia Dortmund	Bundesliga	Mittelfeld	8.0	Felix Nmecha	2000-10-10	Deutschland	15000000.0	1.0	22	Star
19	Borussia Dortmund	Bundesliga	Mittelfeld	30.0	Ole Pohlmann	2001-04-05	Deutschland	400000.0	1.0	22	NaN
20	Borussia Dortmund	Bundesliga	Mittelfeld	19.0	Julian Brandt	1996-05-02	Deutschland	40000000.0	0.0	27	Star
21	Borussia Dortmund	Bundesliga	Mittelfeld	7.0	Giovanni Reyna	2002-11-13	Vereinigte Staaten	25000000.0	0.0	20	Rising Star
22	Borussia Dortmund	Bundesliga	Mittelfeld	11.0	Marco Reus	1989-05-31	Deutschland	7000000.0	0.0	34	NaN
23	Borussia Dortmund	Bundesliga	Sturm	27.0	Karim Adeyemi	2002-01-18	Deutschland	46000000.0	0.0	21	Rising Star
24	Borussia Dortmund	Bundesliga	Sturm	43.0	Jamie Bynoe-Gittens	2004-08-08	England	16099999.999999998	0.0	19	Rising Star

∪

```
df_psg_player
```

1 to 25 of 39 entries Filter

index	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value
0	FC Paris Saint-Germain	Ligue 1	Torwart	99	Gianluigi Donnarumma	25.02.1999 (24)	Italien	45,00 Mio. €
1	FC Paris Saint-Germain	Ligue 1	Torwart	40	Arnau Tenas	30.05.2001 (22)	Spanien	5,00 Mio. €
2	FC Paris Saint-Germain	Ligue 1	Torwart	1	Keylor Navas	15.12.1986 (36)	Costa Rica	4,00 Mio. €
3	FC Paris Saint-Germain	Ligue 1	Torwart	16	Sergio Rico	01.09.1993 (29)	Spanien	3,00 Mio. €
4	FC Paris Saint-Germain	Ligue 1	Torwart	30	Alexandre Letellier	11.12.1990 (32)	Frankreich	300 Tsd. €
5	FC Paris Saint-Germain	Ligue 1	Abwehr	5	Marquinhos	14.05.1994 (29)	Brasilien	65,00 Mio. €
6	FC Paris Saint-Germain	Ligue 1	Abwehr	37	Milan Skriniar	11.02.1995 (28)	Slowakei	50,00 Mio. €
7	FC Paris Saint-Germain	Ligue 1	Abwehr	21	Lucas Hernández	14.02.1996 (27)	Frankreich	45,00 Mio. €
8	FC Paris Saint-Germain	Ligue 1	Abwehr	3	Presnel Kimpembe	13.08.1995 (27)	Frankreich	28,00 Mio. €
9	FC Paris Saint-Germain	Ligue 1	Abwehr	22	Abdou Diallo	04.05.1996 (27)	Senegal	10,00 Mio. €
10	FC Paris Saint-Germain	Ligue 1	Abwehr	25	Nuno Mendes	19.06.2002 (21)	Portugal	65,00 Mio. €
11	FC Paris Saint-Germain	Ligue 1	Abwehr	14	Juan Bernat	01.03.1993 (30)	Spanien	10,00 Mio. €
12	FC Paris Saint-Germain	Ligue 1	Abwehr	32	Layvin Kurzawa	04.09.1992 (30)	Frankreich	3,00 Mio. €
13	FC Paris Saint-Germain	Ligue 1	Abwehr	36	Serif Nhaga	01.09.2005 (17)	Portugal	150 Tsd. €
14	FC Paris Saint-Germain	Ligue 1	Abwehr	2	Achraf Hakimi	04.11.1998 (24)	Marokko	65,00 Mio. €
15	FC Paris Saint-Germain	Ligue 1	Abwehr	26	Nordi Mukiele	01.11.1997 (25)	Frankreich	18,00 Mio. €
16	FC Paris Saint-Germain	Ligue 1	Abwehr	29	Timothée Pembélé	09.09.2002 (20)	Frankreich	5,00 Mio. €
17	FC Paris Saint-Germain	Ligue 1	Abwehr	-	Colin Dagba	09.09.1998 (24)	Frankreich	4,50 Mio. €
18	FC Paris Saint-Germain	Ligue 1	Mittelfeld	4	Manuel Ugarte	11.04.2001 (22)	Uruguay	50,00 Mio. €
19	FC Paris Saint-Germain	Ligue 1	Mittelfeld	-	Leandro Paredes	29.06.1994 (29)	Argentinien	12,00 Mio. €
20	FC Paris Saint-Germain	Ligue 1	Mittelfeld	15	Danilo Pereira	09.09.1991 (31)	Portugal	10,00 Mio. €
21	FC Paris Saint-Germain	Ligue 1	Mittelfeld	17	Vitinha	13.02.2000 (23)	Portugal	42,00 Mio. €
22	FC Paris Saint-Germain	Ligue 1	Mittelfeld	6	Marco Verratti	05.11.1992 (30)	Italien	40,00 Mio. €
23	FC Paris Saint-Germain	Ligue 1	Mittelfeld	8	Fabián Ruiz	03.04.1996 (27)	Spanien	32,00 Mio. €
24	FC Paris Saint-Germain	Ligue 1	Mittelfeld	28	Carlos Soler	02.01.1997 (26)	Spanien	25,00 Mio. €

Data Integration: **concat** (cont.)

```
pd.concat([df_psg_player, df_bvb_player], ignore_index=True)
```

index	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value	player_number_even	age	player_talent
25	FC Paris Saint-Germain	Ligue 1	Mittelfeld	33	Warren Zaïre-Emery	08.03.2006 (17)	Frankreich	20,00 Mio. €	NaN	NaN	NaN
26	FC Paris Saint-Germain	Ligue 1	Mittelfeld	18	Renato Sanches	18.08.1997 (25)	Portugal	15,00 Mio. €	NaN	NaN	NaN
27	FC Paris Saint-Germain	Ligue 1	Mittelfeld	-	Georginio Wijnaldum	11.11.1990 (32)	Niederlande	8,00 Mio. €	NaN	NaN	NaN
28	FC Paris Saint-Germain	Ligue 1	Mittelfeld	38	Edouard Michut	04.03.2003 (20)	Frankreich	2,50 Mio. €	NaN	NaN	NaN
29	FC Paris Saint-Germain	Ligue 1	Mittelfeld	27	Cher Ndour	27.07.2004 (19)	Italien	1,50 Mio. €	NaN	NaN	NaN
30	FC Paris Saint-Germain	Ligue 1	Mittelfeld	19	Kang-in Lee	19.02.2001 (22)	Südkorea	22,00 Mio. €	NaN	NaN	NaN
31	FC Paris Saint-Germain	Ligue 1	Mittelfeld	35	Ismaël Gharbi	10.04.2004 (19)	Spanien	5,00 Mio. €	NaN	NaN	NaN
32	FC Paris Saint-Germain	Ligue 1	Sturm	10	Neymar	05.02.1992 (31)	Brasilien	60,00 Mio. €	NaN	NaN	NaN
33	FC Paris Saint-Germain	Ligue 1	Sturm	34	Julian Draxler	20.09.1993 (29)	Deutschland	6,00 Mio. €	NaN	NaN	NaN
34	FC Paris Saint-Germain	Ligue 1	Sturm	11	Marco Asensio	21.01.1996 (27)	Spanien	25,00 Mio. €	NaN	NaN	NaN
35	FC Paris Saint-Germain	Ligue 1	Sturm	7	Kylian Mbappé	20.12.1998 (24)	Frankreich	180,00 Mio. €	NaN	NaN	NaN
36	FC Paris Saint-Germain	Ligue 1	Sturm	9	Gonçalo Ramos	20.06.2001 (22)	Portugal	50,00 Mio. €	NaN	NaN	NaN
37	FC Paris Saint-Germain	Ligue 1	Sturm	44	Hugo Ekitiké	20.06.2002 (21)	Frankreich	20,00 Mio. €	NaN	NaN	NaN
38	FC Paris Saint-Germain	Ligue 1	Sturm	39	Ilyes Housni	14.05.2005 (18)	Frankreich	3,50 Mio. €	NaN	NaN	NaN
39	Borussia Dortmund	Bundesliga	Torwart	1	Gregor Kobel	1997-12-06	Schweiz	35000000	0.0	25.0	Star
40	Borussia Dortmund	Bundesliga	Torwart	35	Marcel Lotka	2001-05-25	Deutschland	1500000	0.0	22.0	NaN
41	Borussia Dortmund	Bundesliga	Torwart	33	Alexander Meyer	1991-04-13	Deutschland	1000000			
42	Borussia Dortmund	Bundesliga	Torwart	31	Silas Ostrzinski	2003-11-19	Deutschland	150000			
43	Borussia Dortmund	Bundesliga	Abwehr	4	Nico Schlotterbeck	1999-12-01	Deutschland	40000000			
44	Borussia Dortmund	Bundesliga	Abwehr	25	Niklas Süle	1995-09-03	Deutschland	35000000			

Check DataFrame before and afterwards for all (redundant) columns and other inconsistencies.

Data Integration: **merge/join**

Merge ***DataFrame*** *or* ***named Series objects*** *with a* ***database-style join****.*

- Joining columns on columns: DataFrame indexes will be ignored.
- Joining indexes on indexes: DataFrame indexes will be passed on.

Parameters:

- how: `{'left', 'right', 'outer', 'inner', 'cross'}, default 'inner'` $\bowtie$
- on: column label or index, `default` intersecting columns in both DataFrames.

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.merge.html

Data Integration: **merge/join** (cont.)

```python
pd.merge(df_bvb_player, df_top_100_clubs, how="inner", on="club_name")
```

df_bvb_player

1 to 31 of 31 entries

index	club_name	club_league	player_position	player_number	player_name	player_dob	player_country	player_value	player_number_even	age	player_talent
0	Borussia Dortmund	Bundesliga	Torwart	1.0	Gregor Kobel	1997-12-06	Schweiz	35000000.0	0.0	25	Star
1	Borussia Dortmund	Bundesliga	Torwart	35.0	Marcel Lotka	2001-05-25	Deutschland	1500000.0	0.0	22	NaN
2	Borussia Dortmund	Bundesliga	Torwart	33.0	Alexander Meyer	1991-04-13	Deutschland	1000000.0	0.0	32	NaN
3	Borussia Dortmund	Bundesliga	Torwart	31.0	Silas Ostrzinski	2003-11-19	Deutschland	150000.0	0.0	19	NaN
4	Borussia Dortmund	Bundesliga	Abwehr	4.0	Nico Schlotterbeck	1999-12-01	Deutschland	40000000.0	1.0	23	Star
5	Borussia Dortmund	Bundesliga	Abwehr	25.0	Niklas Süle	1995-09-03	Deutschland	35000000.0	0.0	28	Star
6	Borussia Dortmund	Bundesliga	Abwehr	15.0	Mats Hummels	1988-12-16	Deutschland	6000000.0	0.0	34	NaN
7	Borussia Dortmund	Bundesliga	Abwehr	44.0	Soumaïla Coulibaly	2003-10-14	Frankreich	1000000.0	1.0	19	NaN
8	Borussia Dortmund	Bundesliga	Abwehr	47.0	Antonios Papadopoulos	1999-09-10	Deutschland	600000.0	0.0	24	NaN
9	Borussia Dortmund	Bundesliga	Abwehr	5.0	Ramy Bensebaini	1995-04-16	Algerien	20000000.0	0.0	28	Star
10	Borussia Dortmund	Bundesliga	Abwehr	26.0	Julian Ryerson	1997-11-17	Norwegen	13000000.0	1.0	25	Star
11	Borussia Dortmund	Bundesliga	Abwehr	17.0	Marius Wolf	1995-05-27	Deutschland	10000000.0	0.0	28	NaN
12	Borussia Dortmund	Bundesliga	Abwehr	24.0	Thomas Meunier	1991-09-12	Belgien	5000000.0	1.0	32	NaN
13	Borussia Dortmund	Bundesliga	Abwehr	2.0	Mateu Morey Bauzà	2000-03-02	Spanien	1000000.0	1.0	23	NaN
14	Borussia Dortmund	Bundesliga	Mittelfeld	23.0	Emre Can	1994-01-12	Deutschland	14000000.0	0.0	29	Star
15	Borussia Dortmund	Bundesliga	Mittelfeld	6.0	Salih Özcan	1998-01-11	Türkei	13000000.0	1.0	25	Star
16	Borussia Dortmund	Bundesliga	Mittelfeld	32.0	Abdoulaye Kamara	2004-11-06	Frankreich	1000000.0	1.0	18	NaN
17	Borussia Dortmund	Bundesliga	Mittelfeld	20.0	Marcel Sabitzer	1994-03-17	Österreich	20000000.0	1.0	29	Star
18	Borussia Dortmund	Bundesliga	Mittelfeld	8.0	Felix Nmecha	2000-10-10	Deutschland	15000000.0	1.0	22	Star
19	Borussia Dortmund	Bundesliga	Mittelfeld	30.0	Ole Pohlmann	2001-04-05	Deutschland	400000.0	1.0	22	NaN
20	Borussia Dortmund	Bundesliga	Mittelfeld	19.0	Julian Brandt	1996-05-02	Deutschland	40000000.0	0.0	27	Star
21	Borussia Dortmund	Bundesliga	Mittelfeld	7.0	Giovanni Reyna	2002-11-13	Vereinigte Staaten	25000000.0	0.0	20	Rising Star
22	Borussia Dortmund	Bundesliga	Mittelfeld	11.0	Marco Reus	1989-05-31	Deutschland	7000000.0	0.0	34	NaN
23	Borussia Dortmund	Bundesliga	Sturm	27.0	Karim Adeyemi	2002-01-18	Deutschland	46000000.0	0.0	21	Rising Star
24	Borussia Dortmund	Bundesliga	Sturm	43.0	Jamie Bynoe-Gittens	2004-08-08	England	16099999.999999998	0.0	19	Rising Star
25	Borussia Dortmund	Bundesliga	Sturm	10.0	Thorgan Hazard	1993-03-29	Belgien	8049999.999999999	1.0	30	NaN
26	Borussia Dortmund	Bundesliga	Sturm	21.0	Donyell Malen	1999-01-19	Niederlande	32200000.0	0.0	24	Star
27	Borussia Dortmund	Bundesliga	Sturm	16.0	Julien Duranville	2006-05-05	Belgien	9775000.0	1.0	17	Rising Star
28	Borussia Dortmund	Bundesliga	Sturm	9.0	Sébastien Haller	1994-06-22	Elfenbeinküste	34500000.0	0.0	29	Star
29	Borussia Dortmund	Bundesliga	Sturm	18.0	Youssoufa Moukoko	2004-11-20	Deutschland	34500000.0	1.0	18	Rising Star
30	Borussia Dortmund	Bundesliga	NaN	NaN	Hanna Muster	2000-07-17	Deutschland	NaN	NaN	23	NaN

⋈

df_top_100_clubs

1 to 25 of 100 entries

index	club_name	club_number_player	club_avg_age	club_league	club_number_foreign_players	club_number_national_players	club_stadium	club_stadium_seats	club_current_transfer_balance
0	Manchester City	27	26,3	Premier League	17	19	Etihad Stadium	55.017 Plätze	-79,10 Mio. €
1	FC Arsenal	33	24,7	Premier League	24	22	Emirates Stadium	60.704 Plätze	-197,75 Mio. €
2	FC Paris Saint-Germain	39	25,3	Ligue 1	27	22	Parc des Princes	49.691 Plätze	-128,00 Mio. €
3	Real Madrid	24	26,7	LaLiga	16	19	Santiago Bernabéu	81.044 Plätze	-124,50 Mio. €
4	FC Chelsea	30	23,8	Premier League	19	18	Stamford Bridge	40.853 Plätze	+46,90 Mio. €
5	FC Bayern München	27	25,3	Bundesliga	16	20	Allianz Arena	75.024 Plätze	+51,75 Mio. €
6	Manchester United	36	25,2	Premier League	23	22	Old Trafford	74.879 Plätze	-168,60 Mio. €
7	FC Barcelona	23	25,8	LaLiga	12	18	Spotify Camp Nou	99.354 Plätze	+44,50 Mio. €
8	Tottenham Hotspur	35	25,5	Premier League	24	21	Tottenham Hotspur Stadium	62.062 Plätze	-182,00 Mio. €
9	FC Liverpool	22	26,3	Premier League	17	14	Anfield	54.074 Plätze	-51,30 Mio. €
10	Newcastle United	30	27,6	Premier League	16	13	St James' Park	52.338 Plätze	-108,60 Mio. €
11	Aston Villa	28	26,3	Premier League	18	13	Villa Park	42.682 Plätze	-85,10 Mio. €
12	AC Mailand	32	25,8	Serie A	25	17	Giuseppe Meazza	75.923 Plätze	-46,50 Mio. €
13	SSC Neapel	30	25,5	Serie A	18	17	Stadio Diego Armando Maradona	54.726 Plätze	+7,50 Mio. €
14	Atlético Madrid	29	27,6	LaLiga	17	17	Civitas Metropolitano	67.829 Plätze	+43,30 Mio. €
15	Juventus Turin	35	25,7	Serie A	19	17	Allianz Stadium	41.507 Plätze	-41,60 Mio. €
16	Inter Mailand	26	26,7	Serie A	15	16	Giuseppe Meazza	75.923 Plätze	+74,00 Mio. €
17	Borussia Dortmund	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
18	Bayer 04 Leverkusen	26	24,2	Bundesliga	19	14	BayArena	30.210 Plätze	+16,20 Mio. €
19	Brighton & Hove Albion	30	25,0	Premier League	22	15	AMEX Stadium	31.800 Plätze	-2,75 Mio. €
20	FC Brentford	28	25,6	Premier League	21	16	Brentford Community Stadium	17.250 Plätze	-58,35 Mio. €

Data Integration: **merge/join** (cont.)

```
pd.merge(df_bvb_player, df_top_100_clubs, how="inner", on="club_name")
```

```
pd.merge(df_bvb_player, df_top_100_clubs, how="inner", on="club_name")
```

1 to 25 of 31 entries Filter

index	club_name	club_league_x	player_position	player_number	player_name	player_dob	player_country	player_value	player_number_even	age	player_talent	club_number_player	club_avg_age	club_league_y	club_number_foreign_players	club_number_national_players	club_stadium	club_stadium_seats	club_current_transfer_balance
0	Borussia Dortmund	Bundesliga	Torwart	1.0	Gregor Kobel	1997-12-06	Schweiz	35000000.0	0.0	25	Star	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
1	Borussia Dortmund	Bundesliga	Torwart	35.0	Marcel Lotka	2001-05-25	Deutschland	1500000.0	0.0	22	NaN	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
2	Borussia Dortmund	Bundesliga	Torwart	33.0	Alexander Meyer	1991-04-13	Deutschland	1000000.0	0.0	32	NaN	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
3	Borussia Dortmund	Bundesliga	Torwart	31.0	Silas Ostrzinski	2003-11-19	Deutschland	150000.0	0.0	19	NaN	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
4	Borussia Dortmund	Bundesliga	Abwehr	4.0	Nico Schlotterbeck	1999-12-01	Deutschland	40000000.0	1.0	23	Star	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
5	Borussia Dortmund	Bundesliga	Abwehr	25.0	Niklas Süle	1995-09-03	Deutschland	35000000.0	0.0	28	Star	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
6	Borussia Dortmund	Bundesliga	Abwehr	15.0	Mats Hummels	1988-12-16	Deutschland	6000000.0	0.0	34	NaN	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
7	Borussia Dortmund	Bundesliga	Abwehr	44.0	Soumaila Coulibaly	2003-10-14	Frankreich	1000000.0	1.0	19	NaN	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
8	Borussia Dortmund	Bundesliga	Abwehr	47.0	Antonios Papadopoulos	1999-09-10	Deutschland	600000.0	0.0	24	NaN	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
9	Borussia Dortmund	Bundesliga	Abwehr	5.0	Ramy Bensebaini	1995-04-16	Algerien	20000000.0	0.0	28	Star	30	24,8	Bundesliga	15	18	SIGNAL IDUNA PARK	81.365 Plätze	+59,35 Mio. €
10	Borussia Dortmund	Bundesliga	Abwehr	26.0	Julian Ryerson	1997-11-17	Norwegen	13000000.0	1.0	25	Star	30	24,8	Bundesliga	15	18	SIGNAL	81.365 Plätze	+59,35 Mio. €
11	Borussia Dortmund	Bundesliga	Abwehr	17.0	Marius Wolf	1995-05-27	Deutschland	10000000.0	0.0	28	NaN	30	24,8	Bundesliga	15				
12	Borussia Dortmund	Bundesliga	Abwehr	24.0	Thomas Meunier	1991-09-12	Belgien	5000000.0	1.0	32	NaN	30	24,8	Bundesliga	15				
13	Borussia Dortmund	Bundesliga	Abwehr	2.0	Mateu Morey Bauzà	2000-03-02	Spanien	1000000.0	1.0	23	NaN	30	24,8	Bundesliga	15				
14	Borussia Dortmund	Bundesliga	Mittelfeld	23.0	Emre Can	1994-01-12	Deutschland	14000000.0	0.0	29	Star	30	24,8	Bundesliga	15				
15	Borussia Dortmund	Bundesliga	Mittelfeld	6.0	Salih Özcan	1998-01-11	Türkei	13000000.0	1.0	25	Star	30	24,8	Bundesliga	15				

Check DataFrame before and afterwards for all (redundant) columns and other inconsistencies.

Major Tasks in **Data Preprocessing**

Data Reduction

- Obtains reduced representation in volume but produces the same or similar analytical results.

Data Cleaning

- Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies caused by data integration.

Data Integration

- Integration of multiple tables, databases, data cubes, or files.

Data Transformation

- Aggregation, generalization, normalization and **attribute construction**.

Data Transformation

*A function that **maps** the entire **set** of values of a given attribute **to** a **new set** of **replacement values**.*

Attribute Construction:

- Unary function definition `f(A)` → `A`, where A is a set or
- Binary function definition `f(A, B)` → `A*B`, whera A.index ≡ B.index, and * some operation

Aggregation: involves grouping and computations such as sum(), mean(), median(), min(), and max(), to generate insights into the nature of numeric values.

Generalization: concept hierarchy climbing.

Normalization: series transformation to a scale so values lie within a specified range (usually smaller and positive).

Some DS frameworks consider **Data Transformation** seperated from Data Preprocessing.

Data Transformation: **binary function**

Training #1

Open a blank .ipynb file and import the .csv file

```
url = "https://raw.githubusercontent.com/leotraeg/FHDTM-P2DS-WS2324/main/Data%20Science%20Projekt%20Demo/Datens%C3%A4tze/FHDTM-P2DS-WS2324-Project-Demo-2.0-Data-Preprocessing-Transfermarkt_BVB.csv"
```

as a pandas data frame `pd.read_csv(url)`.

We will work with Hanna in the next slides!

1. "Hanna Muster" joined the BVB club this season. You can add a new record using a `dictionary { }` and `pd.Series` object by `df.loc[len(df)] = pd.Series(data=dictionary)`. Assign the unknown attributes with numpy's `np.NaN` value.
2. Raise the player_value attribute of the DataFrame for all attackers ("Sturm") by 15%.
3. Add a new attribute player_talent to the DataFrame in which
   - Players with player_value greater than 10 mil. = "Star"
   - Players aged under and including 21 with player_value greater than 1 mil. = "Rising Star".

...you may have used **masking** and **value** setting for this task ☺

As long as it works, that is fine, but actually this task is a **great example** for applying a **binary function**

Advantage with functional programming: reproducibility, readability, and documentation.

Data Transformation: **attribute construction via binary function**

```python
def player_talent(player_value, age):
  if (player_value > 1000000) and (age <= 21):
    return "Rising Star"
  elif player_value > 10000000:
    return "Star"
  else:
    return "No Category"
```

We can also use `.apply` on `DataFrames` and then define a `lambda` function calling internally the user-defined function with multiple input values.

```python
df_bvb_player.apply(lambda x: player_talent(x.player_value, x.age), axis=1)
```

`axis=1` forces row-wise computation

Lambda functions

Definition: `lambda` *`arguments`* `:` *`expression`*

- Small **anonymous** function.
- Take **any number** of **arguments but** can only have **one expression.**

```
x = lambda a : a + 1
print(x(5))          #Prints "6"

x = lambda a, b : a * b
print(x(5, 4))       #Prints "20"

x = lambda a, b, c : a + b + c
print(x(5, 4, 3))    #Prints "12"
```

Data Transformation: attribute construction via binary function (cont.)

Definition binary function: `f(A, B)` → `A*B`

```python
def player_talent(player_value, age):
  if (player_value > 1000000) and (age <= 21):
    return "Rising Star"
  elif player_value > 10000000:
    return "Star"
  else:
    return "No Category"
```

```python
df_bvb_player.player_value
```

```
0     35000000.0
1      1500000.0
2      1000000.0
3       150000.0
4     40000000.0
5     35000000.0
6      6000000.0
7      1000000.0
8       600000.0
9     20000000.0
10    13000000.0
11    10000000.0
12     5000000.0
13     1000000.0
14    14000000.0
15    13000000.0

29    34500000.0
30           NaN
```

```python
df_bvb_player.age
```

```
0     25
1     22
2     32
3     19
4     23
5     28
6     34
7     19
8     24
9     28
10    25
11    28
12    32
13    23
14    29
15    25

29    18
30    23
```

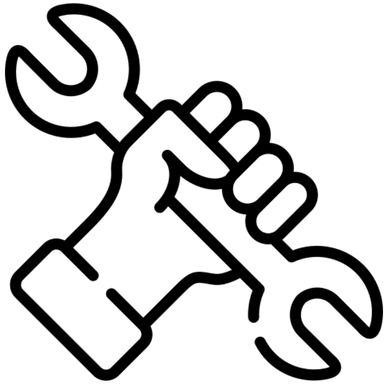
(,) =

```python
df_bvb_player.apply(
    lambda x:
    player_talent(x.player_value, x.age),
    axis=1)
```

```
0            Star
1     No Category
2     No Category
3     No Category
4            Star
5            Star
6     No Category
7     No Category
8     No Category
9            Star
10           Star
11    No Category
12    No Category
13    No Category
14           Star
15           Star

29    Rising Star
30    No Category
```

Training #2



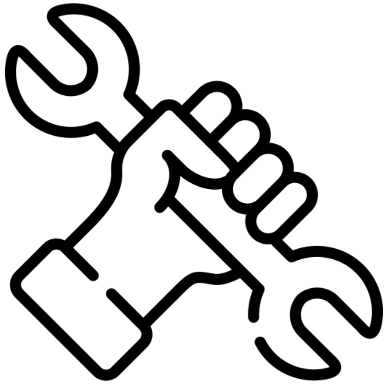
Open a blank .ipynb file and import the .csv file

```
url = "https://raw.githubusercontent.com/leotraeg/FHDTM-P2DS-WS2324/main/Data%20Science%20Projekt%20Demo/Datens%C3%A4tze/FHDTM-P2DS-WS2324-Project-Demo-2.0-Data-Preprocessing-Transfermarkt_Top_100_Clubs.csv"
```

as a pandas data frame `pd.read_csv(url)`.

1. Write a function that takes the number_player and number_foreign_players of a club as the input parameter.
2. Compute the ratio of foreign players by the total number of players of a club.
3. If the ratio is above 85%, the club is “extremely international”.
   If the ratio is above 20%, the club is “very international”.
   Else, the club is “international”.
4. Append a new column “club_internationality” using the `df.apply` and `lambda` function.

Training #2



What clubs are “extremely international”?

index	club_name	club_number_player	club_avg_age	club_league	club_number_foreign_players	club_number_national_players	club_stadium	club_stadium_seats	club_current_transfer_balance	club_internationality
25	Wolverhampton Wanderers	26	25,8	Premier League	23	14	Molineux Stadium	32.050 Plätze	+44,70 Mio. €	extremly international
36	AS Monaco	27	23,9	Ligue 1	27	10	Stade Louis-II	18.523 Plätze	+26,00 Mio. €	extremly international

…and how many more?

Major Tasks in **Data Preprocessing**

Data Reduction

- Obtains reduced representation in volume but produces the same or similar analytical results.

Data Cleaning

- Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies caused by data integration.

Data Integration

- Integration of multiple tables, databases, data cubes, or files.

Data Transformation

- **Aggregation,** generalization, normalization and attribute construction.

Data Transformation: **aggregation**

- *Aggregation operation **reduce** the **entire “array”** to a **single summarizing value**!* (Recap NumPy)
- Essential piece of analysis of large data is **efficient summarization.**
- Aggregation is applicable on both DataFrame and Series objects.

`df.describe()` applies a common subset of all these

Aggregation	**Description**
`count()`	Total number of items
`first(), last()`	First and last item
`mean(), median()`	Mean and median
`min(), max()`	Minimum and maximum
`std(), var()`	Standard deviation and variance
`mad()`	Mean absolute deviation
`prod()`	Product of all items
`sum()`	Sum of all items

Data Transformation: aggregation (cont.)

```
df_bvb_player.count()
```

```
club_name             31
club_league           31
player_position       30
player_number         30
player_name           31
player_dob            31
player_country        31
player_value          30
player_number_even    30
age                   31
player_talent         17
```

```
df_bvb_player.age.sum()
```

```
769
```

```
df_bvb_player.age.mean()
```

```
24.806451612903224
```

```
df_bvb_player.age.median()
```

```
24.0
```

```
df_bvb_player.age.std()
```

```
4.881388838153268
```

```
df_bvb_player.player_value.sum()
```

```
485775000.0
```

```
df_bvb_player.player_value.mean()
```

```
16192500.0
```

```
df_bvb_player.player_value.median()
```

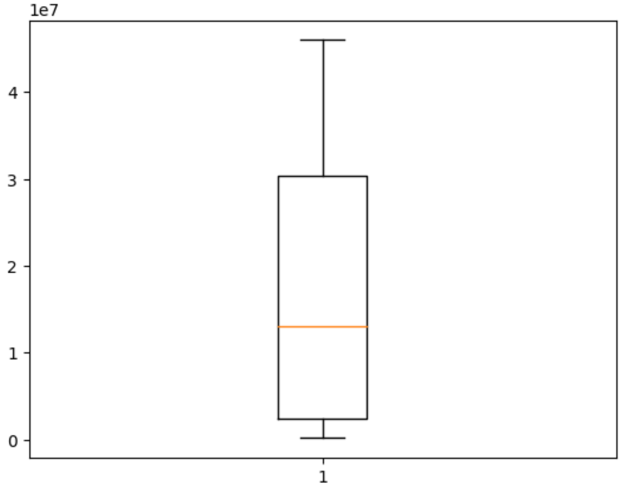
```
13000000.0
```

```
df_bvb_player.player_value.std()
```

```
14548117.47212534
```

```
import matplotlib.pyplot as plt
```

```
fig = plt.figure()
plt.boxplot(df_bvb_player.player_value.dropna().values)
plt.show()
```



Technology Arts Sciences TH Köln

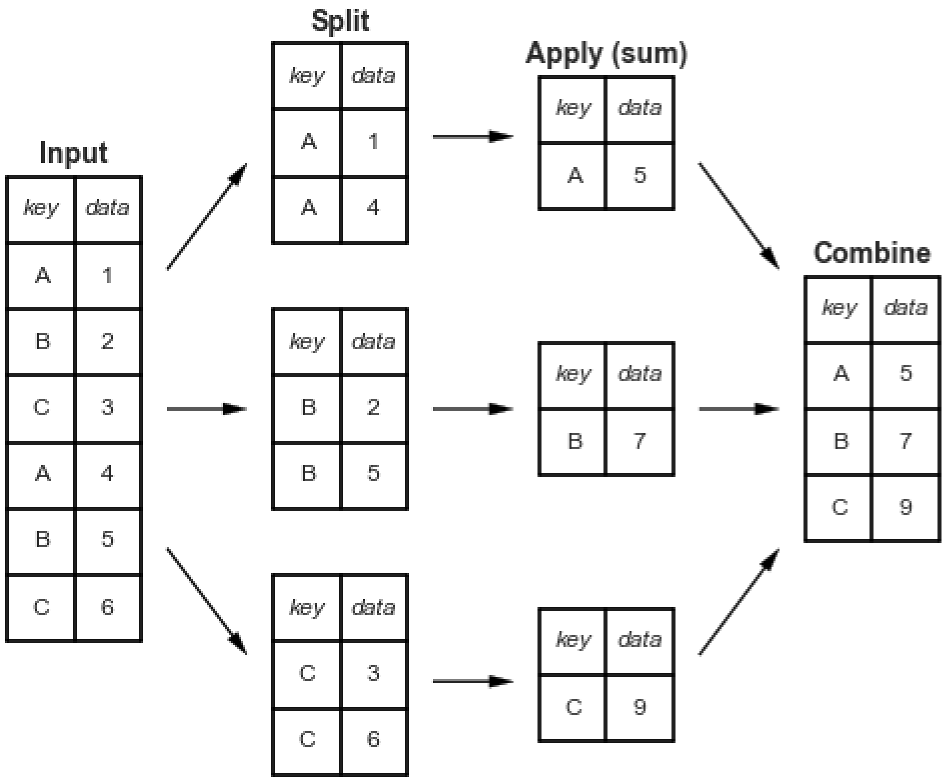
Data Transformation: **grouping and aggregation**

- To go deeper into the data, however, **simple aggregates** are often **not enough**.
- `groupby` operation: *quickly and efficiently compute **aggregates** on **subsets of data**.*
- Aggregate conditionally on some label, e.g., player_position or student_stereotype.
- "group by" comes from SQL database language.

https://www.soccermaniak.com/soccer-positions.html

https://charlie.csu.edu.au/2016/02/26/five-types-of-students-youll-meet-at-uni/

Data Transformation: grouping and aggregation (cont.)



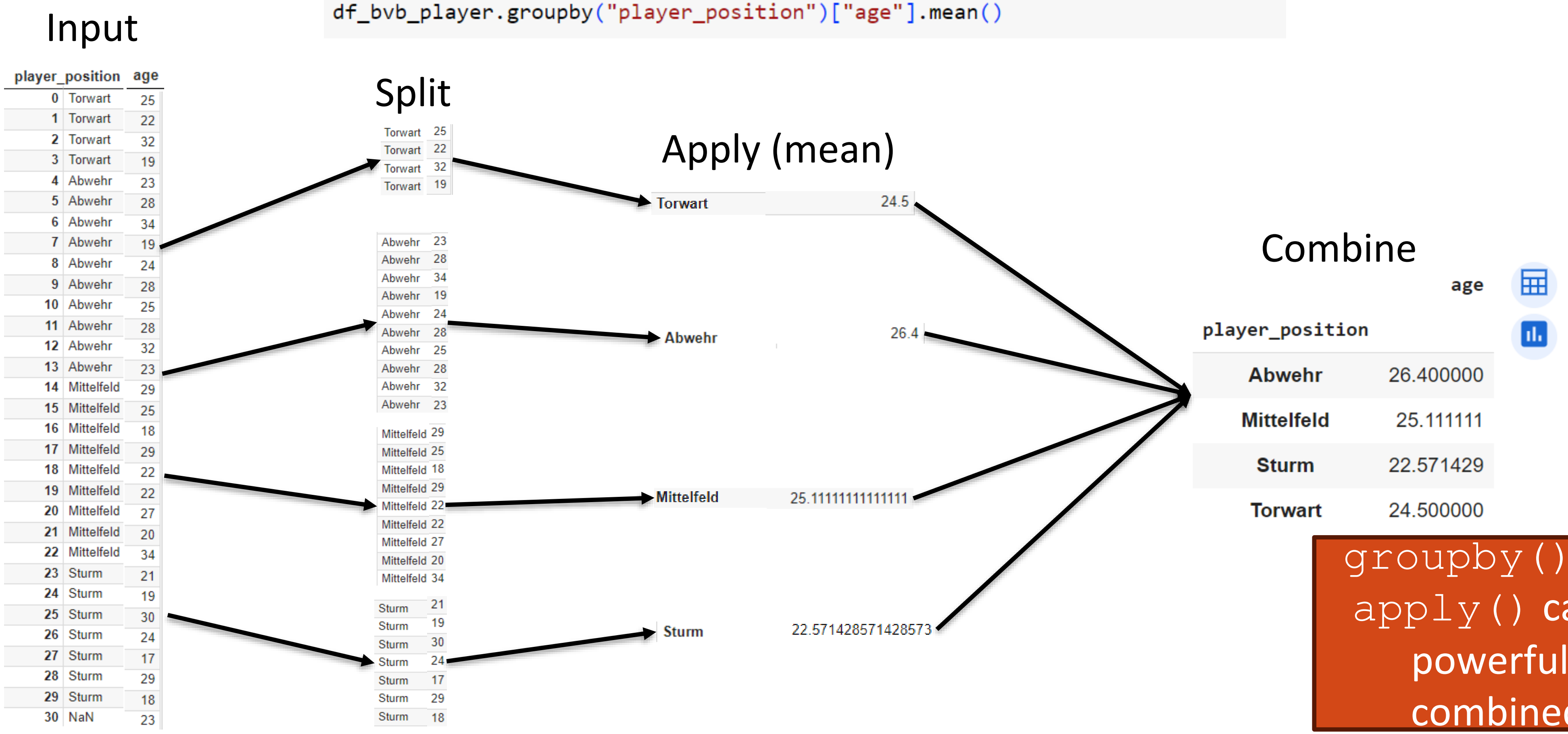
Hadley Wickham of Rstats fame: *split, apply, combine*.

1. **Split**: grouping a DataFrame depending on the value of the specified key.
2. **Apply**: computing some function, usually an aggregate, transformation, or filtering, within the individual groups.
3. **Combine:** merging the results of these operations into an output array.

`df.groupby()` does this in a **single pass** over the data.

Data Transformation: grouping and aggregation (cont.)

```
df_bvb_player.groupby("player_position")["age"].mean()
```

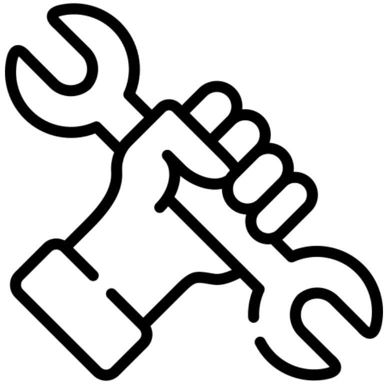


	player_position	age
0	Torwart	25
1	Torwart	22
2	Torwart	32
3	Torwart	19
4	Abwehr	23
5	Abwehr	28
6	Abwehr	34
7	Abwehr	19
8	Abwehr	24
9	Abwehr	28
10	Abwehr	25
11	Abwehr	28
12	Abwehr	32
13	Abwehr	23
14	Mittelfeld	29
15	Mittelfeld	25
16	Mittelfeld	18
17	Mittelfeld	29
18	Mittelfeld	22
19	Mittelfeld	22
20	Mittelfeld	27
21	Mittelfeld	20
22	Mittelfeld	34
23	Sturm	21
24	Sturm	19
25	Sturm	30
26	Sturm	24
27	Sturm	17
28	Sturm	29
29	Sturm	18
30	NaN	23

player_position	age
Abwehr	26.400000
Mittelfeld	25.111111
Sturm	22.571429
Torwart	24.500000

`groupby()` and `apply()` can be powerfully combined!

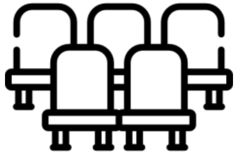
Training #3



Open a blank .ipynb file and import the .csv file

```
url = "https://raw.githubusercontent.com/leotraeg/FHDTM-P2DS-WS2324/main/Data%20Science%20Projekt%20Demo/Datens%C3%A4tze/FHDTM-P2DS-WS2324-Project-Demo-2.0-Data-Preprocessing-Transfermarkt_Top_100_Clubs.csv"
```

as a pandas data frame `pd.read_csv(url)`.



1. Transform club_stadium_seats to a numeric attribute.
2. Group the DataFrame by the club_league and compute the **summation** of seats.

1. Transform club_current_transfer_balance to a numeric attribute.
   Hint: only millions in this column
2. Group the DataFrame by the club_league and compute the **mean** transfer balance.

Training #3

You can use `.sort_values(ascending=False)` to rank the attribute descendingly.

```
club_league
Premier League
Bundesliga
LaLiga
Serie A
Ligue 1
Saudi Pro League
Liga Portugal
Süper Lig
Campeonato Brasileiro Série A
Eredivisie
Premier Liga
Scottish Premiership
Championship
Liga Profesional de Fútbol
Jupiler Pro League
SuperSport HNL                      35123
Name: club_stadium_seats, dtype: int64
```

```
club_league
Saudi Pro League
Premier League
Ligue 1
SuperSport HNL
Süper Lig
Campeonato Brasileiro Série A
Premier Liga
Serie A
LaLiga
Jupiler Pro League
Scottish Premiership
Liga Portugal
Bundesliga
Eredivisie
Championship
Liga Profesional de Fútbol       3.367000e+07
Name: club_current_transfer_balance, dtype: float64
```

Major Tasks in **Data Preprocessing**

Data Reduction

- Obtains reduced representation in volume but produces the same or similar analytical results.

Data Cleaning

- Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies caused by data integration.

Data Integration

- Integration of multiple tables, databases, data cubes, or files.

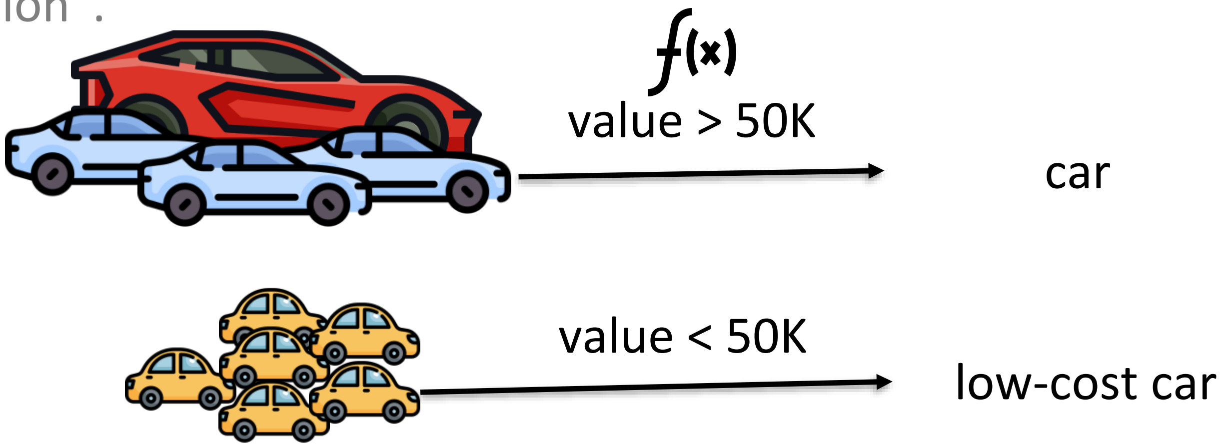
Data Transformation

- Aggregation**, generalization**, normalization and attribute construction.

Data Transformation: generalization

*Process of **transforming low-level attributes** into **high-level** ones by using a **hierarchy**.*

- Also known as data binning and data categorization.
- Strongly related to attribute construction.
- **Declarative**: **manually** deciding how large your data bin sizes are.
- Automated: ultimate goal of machine learning (clustering) and algorithms such as “k-anonymization“.



`is_even` and `player_talent` is an example. Closely related to **attribute construction**.

Data Transformation: generalization hierarchies

- **Schema hierarchy**: partial order to reflect relationships among the attributes in a database.

$$house\_number \prec street \prec city \prec province \prec country.$$

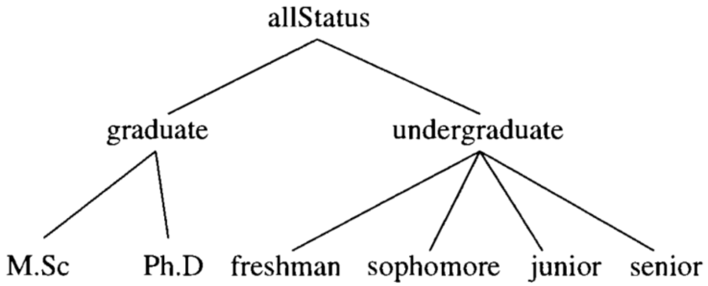
- **Set-grouping hierarchy**: defined on the set of instances of an attribute.

$$\{\text{freshman, sophomore, junior, senior}\} \prec \text{undergraduate}$$

$$\{\text{M.Sc, Ph.D}\} \prec \text{graduate}$$

$$\{\text{undergraduate, graduate}\} \prec \text{allStatus}$$

$\equiv$



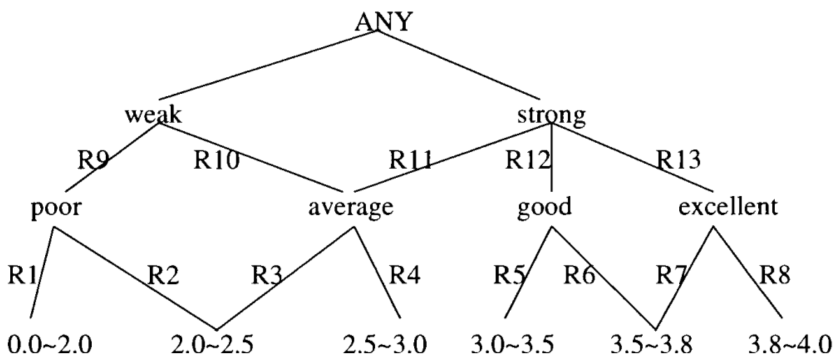
Yijun **Lu**. Specification, **generation** and implementation **concept hierarchy in data mining**. December 1997

Data Transformation: generalization hierarchies (cont.)

- **Operation-derived hierarchy**: defined by a set of operations onto (usually numeric) data.

$$\{20,000.00, \ldots, 39,999.99\} \subset 20 \sim 40K,$$

- **Rule-based hierarchy**: nested conditional rules defining higher level correspondence.



*Yijun **Lu**. Specification, **generation** and implementation **concept hierarchy in data mining**. December 1997*

Major Tasks in **Data Preprocessing**

Data Reduction

- Obtains reduced representation in volume but produces the same or similar analytical results.

Data Cleaning

- Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies caused by data integration.

Data Integration

- Integration of multiple tables, databases, data cubes, or files.

Data Transformation

- Aggregation, generalization, **normalization** and attribute construction.

Data Transformation: normalization

Adjusting numeric values *measured on* ***different scales*** *to a* ***notionally common scale.***

Linear scaling, feature scaling, or min-max normalization (0..1):

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Formula can be adapted to desired [new_minA, new_maxA] scale.

Works also for negative ranges!!!

Apply after outlier removal!!!

Why do normalization?

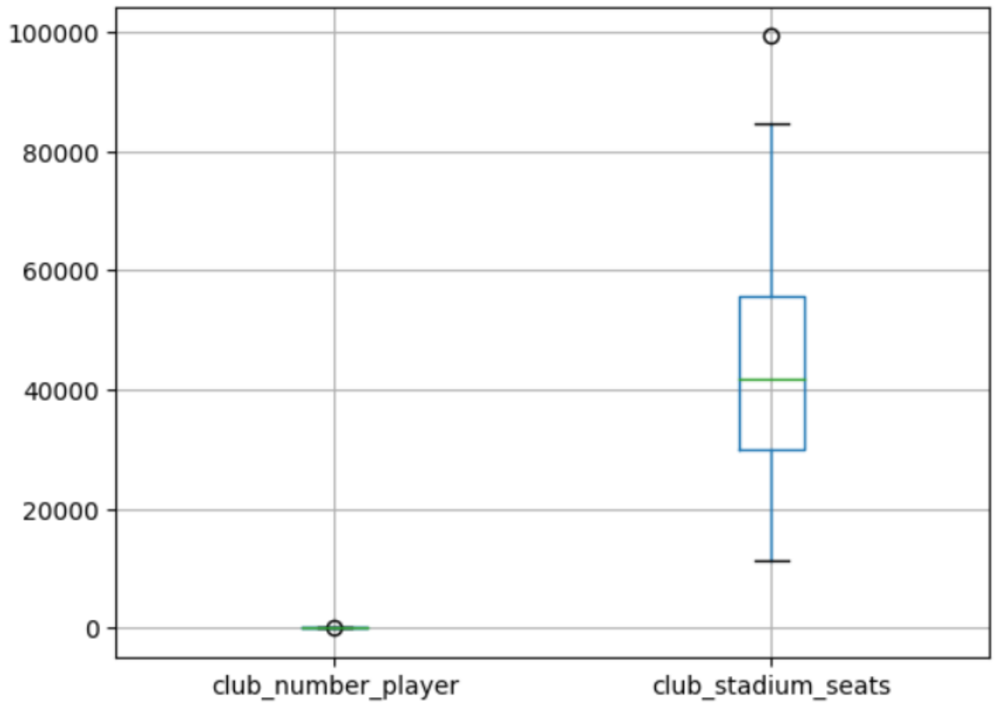
- Improved visualization.
- Neccesarry for almost all Machine Learning techniques.
- Other methods: Clipping, Log Scaling, Z-score

Data Transformation: normalization (cont.)

...ranges of numeric attributes differ with each other.

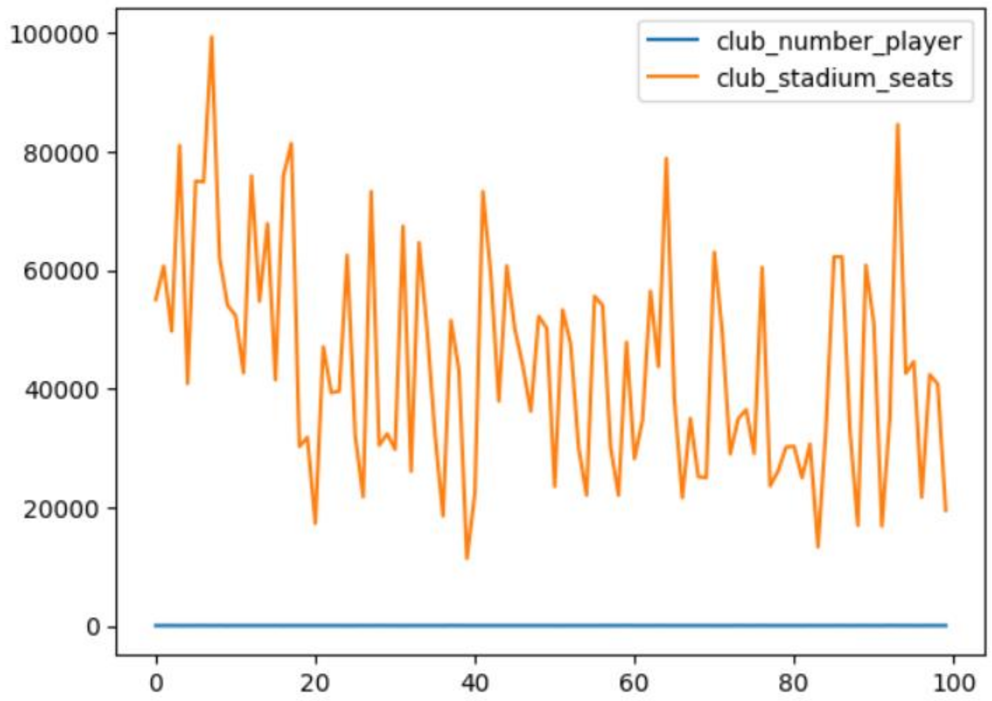
```python
df_top_100_clubs[["club_number_player", "club_stadium_seats"]].boxplot()
```

```
<Axes: >
```




```python
df_top_100_clubs[["club_number_player", "club_stadium_seats"]].plot()
```

```
<Axes: >
```

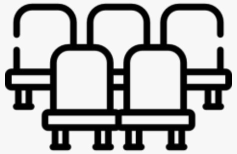


Data Transformation: normalization (cont.)

```
min = df_top_100_clubs.club_number_player.min()
print(min)
max = df_top_100_clubs.club_number_player.max()
print(max)
df_top_100_clubs["club_number_player_scaled"] = (df_top_100_clubs.club_number_player - min) / (max - min)
```


```
22
43
```

```
min = df_top_100_clubs.club_stadium_seats.min()
print(min)
max = df_top_100_clubs.club_stadium_seats.max()
print(max)
df_top_100_clubs["club_stadium_seats_scaled"] = (df_top_100_clubs.club_stadium_seats - min) / (max - min)
```



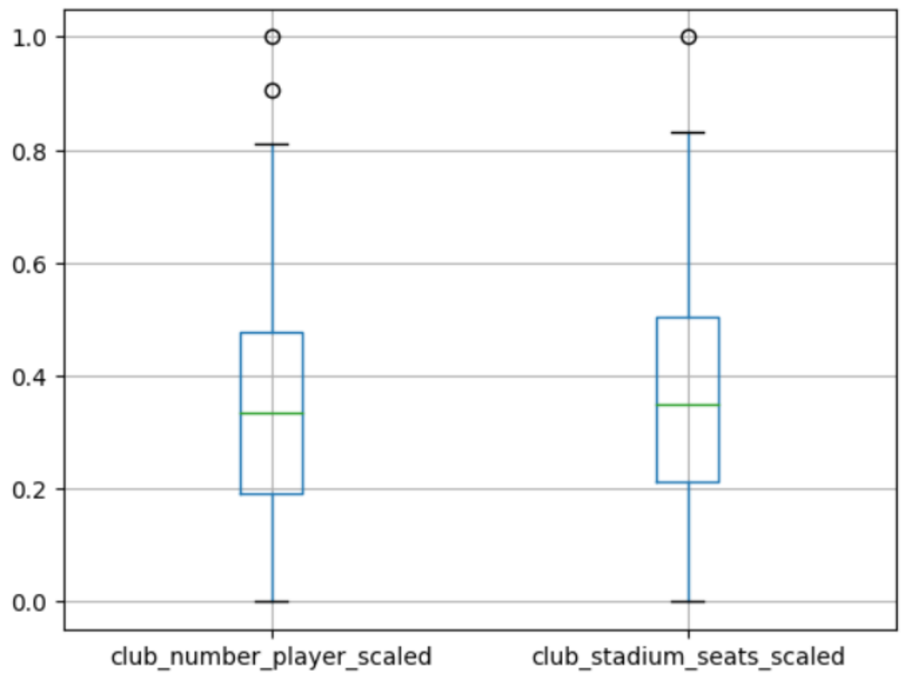
```
11329
99354
```

Data Transformation: normalization (cont.)

...but once normalized, can be visually analyzed on correlation.

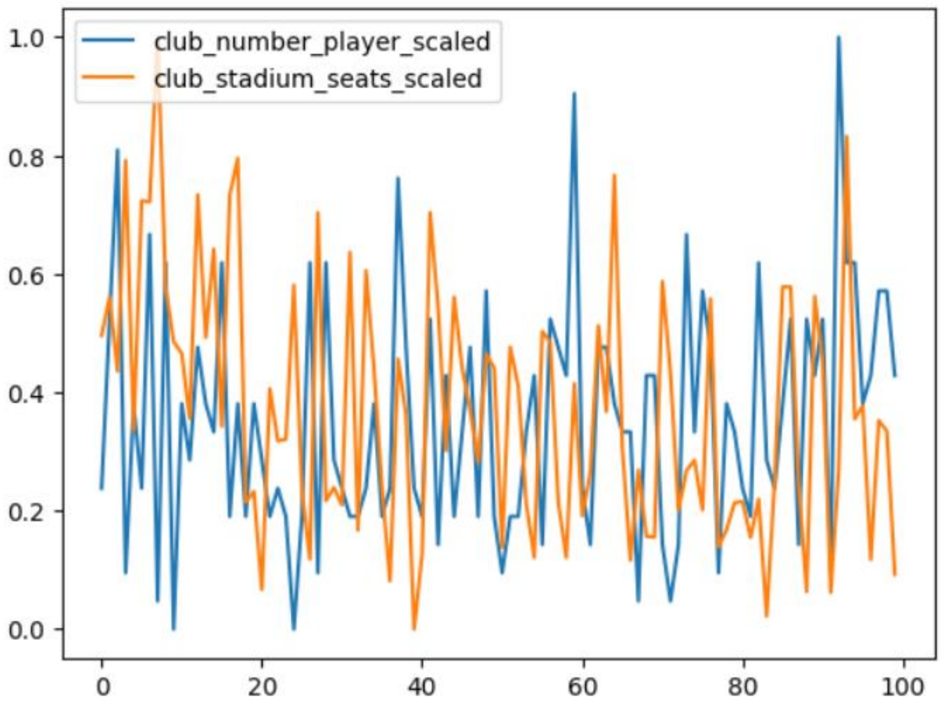
```
df_top_100_clubs[["club_number_player_scaled", "club_stadium_seats_scaled"]].boxplot()
```

```
<Axes: >
```




```
df_top_100_clubs[["club_number_player_scaled", "club_stadium_seats_scaled"]].plot()
```

```
<Axes: >
```



Data Transformation: Scaling `map`

You can use the `map()` method of Pandas for **substituting each value** in a Series with **another value**, that may be derived from a *function*, a *dictionary* or *another Series*.

- Transforming categorical values to numeric ones can give more statistical insights.
- Scaling introduces ranking and ordering bias.

```
df_bvb_player.player_position
```

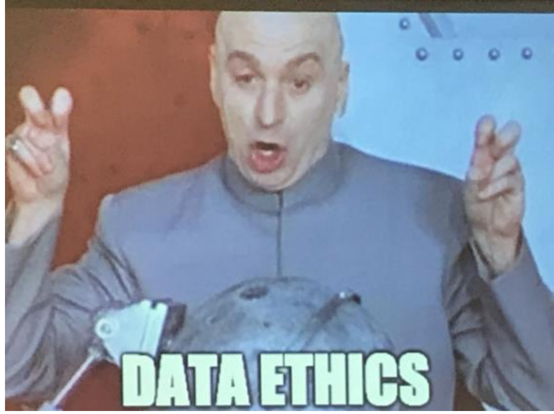
```
0        Torwart
1        Torwart
2        Torwart
3        Torwart
4         Abwehr
5         Abwehr
6         Abwehr
7         Abwehr
8         Abwehr
9         Abwehr
10        Abwehr
11        Abwehr
12        Abwehr
13        Abwehr
14    Mittelfeld
15    Mittelfeld
16    Mittelfeld
17    Mittelfeld
18    Mittelfeld
19    Mittelfeld
20    Mittelfeld
21    Mittelfeld
22    Mittelfeld
23         Sturm
24         Sturm
25         Sturm
26         Sturm
27         Sturm
28         Sturm
29         Sturm
```

```
df_bvb_player.player_position.map(
    {'Torwart': 1, 'Abwehr': 2, 'Mittelfeld': 3, 'Sturm':4, np.NaN:0}
    )
```

```
0     1
1     1
2     1
3     1
4     2
5     2
6     2
7     2
8     2
9     2
10    2
11    2
12    2
13    2
14    3
15    3
16    3
17    3
18    3
19    3
20    3
21    3
22    3
23    4
24    4
25    4
26    4
27    4
28    4
29    4
```

https://pandas.pydata.org/docs/reference/api/pandas.Series.map.html

Preprocessing Considerations



Think about likely **causes of noise** and errors when **correcting and transforming data**, e.g.,

- Do two extremely similar attributes really represent the same?
- Does a missing value have more meaning in the data context than np.NaN?
- Is this “outlier“ really an outlier, or is there a reasonable explanation for it?
- Does removing an outlier harm or help interpreting the whole data context?

Consider **ethics** when applying Data Integration and Transformations:

- Limit harmful uses
- Reflect diversity / inclusion
- Uphold human rights and values

…preprocessing changes the data and introduces new bias.

Takeaways

- NumPy’s efficient vectorization approach works also for Pandas.
- Operations on data in Pandas will always maintain the data context.
- Pandas has a profound programmatic preprocessing suite for data reduction, cleaning, integration, and transformation.
- Always consider changes in the data as they introduce new bias.

Outlook

- In two weeks we will dive deep into Plotting DataFrames.
- Python Matplotlib is the core library you are going to use for visualising and interpreting your data.

- In the weeks after we will jump into R.

See you again next week.

Questions?