Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

Unüberwachtes Lernen: Feature Engineering Praxis

Violeta Vogel

Technik und Informatik, BFH

Vorbereitung der Umgebung

Vorbereiten der Umgebung

- laden der (vorerst) benötigten Libraries

```
import pandas as pd
import numpy as np
```

- festlegen des Datenpfades (ist den jeweiligen Gegebenheiten anzupassen)

```
datapath = '../3_data'
from os import chdir; chdir(datapath)
```

- laden der Beispieldaten in einen pandas Data Frame

```
data = pd.read_csv('bank_data.csv', sep=';')
```

- für späteren Gebrauch: erstellen von Indices mit den Namen der kategorialen resp. numerischen Variablen

```
cat_vars = data.select_dtypes(include=['object']).columns
num_vars = data.select_dtypes(exclude=['object']).columns
```

Eingenschaften des Data Frame: Übersicht

- Objekttyp (Klasse) des eben erstellten Data Frame

```
print(type(data))
```

```
<class 'pandas.core.frame.DataFrame'>
```

- Dimensionen des Data Frame (rows, columns)

```
print(data.shape)
```

```
(9868, 21)
```

- Zusammenfassung, knappe Info zum Data Frame

```
print(data.info())
```

Eingenschaften des Data Frame: Übersicht

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9868 entries, 0 to 9867
Data columns (total 21 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   age              9745 non-null   float64
 1   job              9795 non-null   object
 2   marital          9849 non-null   object
:
dtypes: float64(7), int64(3), object(11)
```

- Typ, Übersicht Index, Spalten Datentypen, nicht Nullvalues und Speicherbedarf
- ausserdem hier vorliegende Datentypen
  - float64, int64: numerisch
  - object: nicht numerisch (kategorial)

Eingenschaften des Data Frame: Übersicht

- erste paar rows (und columns) ausgeben

```
#print(data.head())
print(data.iloc[0:6, 0:6])
```

```
   age           job  marital            education  default housing
0  31.0  blue collar   single            basic.9y  unknown      no
1  29.0      student   single          high.school       no     yes
2  30.0   technician   single  professional.course       no      no
3  86.0      retired  married            basic.4y       no     yes
4  29.0  blue collar  married            basic.4y  unknown      no
5  33.0       admin.  married    university.degree       no      no
```

- (alternativ können mit .tail() auch die letzten rows angezeigt werden

Eingenschaften des Data Frame: Nas (Missing Values)

- wie viele NAs insgesamt?

```
print(data.isna().sum().sum())
```

```
9284
```

- wie viele rows mit NAs?

```
print(data.shape[0] - data.dropna().shape[0])
```

```
8036
```

Eingenschaften des Data Frame: Nas (Missing Values)

- wie sind die NAs auf die einzelnen columns (Variablen) verteilt?

```
s = data.isna().sum()
print(s[s > 0])
```

```
age                123
job                 73
marital             19
education          463
duration            51
poutcome          7814
emp.var.rate       247
cons.price.idx     247
cons.conf.idx      247
```

Eingenschaften des Data Frame: Dublikate

- als Duplikate bezeichnet man Beobachtungen, welche in allen Variablen denselben Wert aufweisen
- die Anzahl allenfalls vorliegender Duplikate kann wie folgt ermittelt werden

```
data.duplicated().sum()
```

```
7
```

- tatsächlich erstellt die Methode .duplicated() eine Serie von Boolschen Werten, welche anschliessend dazu verwendet werden kann, die Duplikate
  - anzuzeigen, oder
  - zu entfernen (vgl. Kap. 1.3.1.1.3)

- ob allerdings Duplikate als Fehler zu betrachten sind, ist eine fachliche Entscheidung

Eingenschaften des Data Frame: Dublikate

- die meisten ML Methoden können nur mit numerischen Werten umgehen
- um Informationen von kategorialen Variablen (vgl. oben: "job", "marital", etc.) trotzdem berücksichtigen zu können, müssen diese numerisch dargestellt werden können
- im Folgenden sollen daher exemplarisch einige derartige Variablen untersucht, und Empfehlungen für entsprechende Umcodierungen erstellt werden

Kategoriale Variablen

- Statistische Kennzahlen einer ausgewählten Variablen

```
print(data.education.head())
```

```
0              basic.9y
1           high.school
2    professional.course
3              basic.4y
4              basic.4y
```

```
print(data.education.describe())
```

```
count                 9405
unique                   7
top       university.degree
freq                  3136
Name: education, dtype: object
```

- dabei bedeuten
  - count: Anzahl nicht Missing Values
  - unique: Anzahl unterschiedliche Werte (Kategorien / Levels)
  - top: häufigster Wert (→ Modalwert)
  - freq: Frequenz (Anzahl Vorkommen) des Modalwertes
  - ausserdem: Name und Datentyp

Kategoriale Variablen

- Übersicht (statistischer) Kennzahlen aller kategorialen Variablen (mit Hilfe des eingangs erstellten Indexes)

```
print(data[cat_vars].describe())
```

```
           job  marital          education default housing  loan   contact  \
count     9795     9849               9405    9868    9868  9868      9868
unique      11        3                  7       2       3     3         2
top     admin.  married  university.degree      no     yes    no  cellular
freq      2630     5747               3136    8268    5211  8196      6993

       month day_of_week poutcome     y
count   9868        9868     2054  9868
unique    10           5        2     2
top      may         thu  failure    no
freq    2745        2152     1101  5223
```

Kategoriale Variablen

- Übersicht zu den Frequenzen der Werte einer ausgewählten Variablen in einer sogenannten Frequenztabelle

```
print(data.education.value_counts())
```

```
university.degree      3136
high.school            2230
professional.course    1307
basic.9y               1251
basic.4y                972
basic.6y                503
illiterate                6
```

Bemerkungen

- Reihenfolge: nach abnehmenden Frequenzen, der Modalwert an erster Stelle
- NAs werden hier nicht mehr speziell ausgewiesen (Default)
- illiterate: schon mal eine vorläufige Empfehlung?

Kategoriale Variablen

- abschliessend eine Zusammenstellung der unique Values aller kategorialen Variablen:

```
print(data[cat_vars].nunique())
```

```
job            11
marital         3
education       7
default         2
housing         3
loan            3
contact         2
month          10
day_of_week     5
poutcome        2
y               2
```

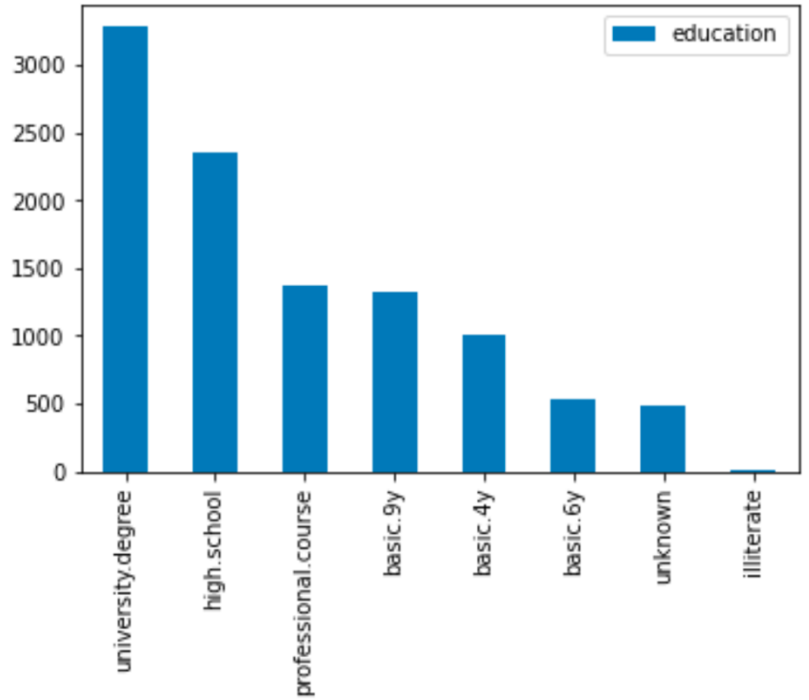
Bemerkungen

- sehr grosse Werte (in der Nähe von count) sind nicht sehr beliebt
  - deuten auf Merkmale wie Labels (IDs) hin, welche für Datenanalyse und Machine Lerning keinen Mehrwert liefern
  - können bei nominaler Umcodierung zu zu vielen Dummy Variablen führen
- im vorliegenden Dataset aber unproblematisch
- `.nunique()` kann auch auf numerische Variablen angewendet werden

Kategoriale Variablen

- Barchart: Standard-Visualisierung für kategoriale Daten: pandas.DataFrame.plot.bar (da `value_counts()` selber ein Objekt vom Typ `pandas.Series` zurückgibt)

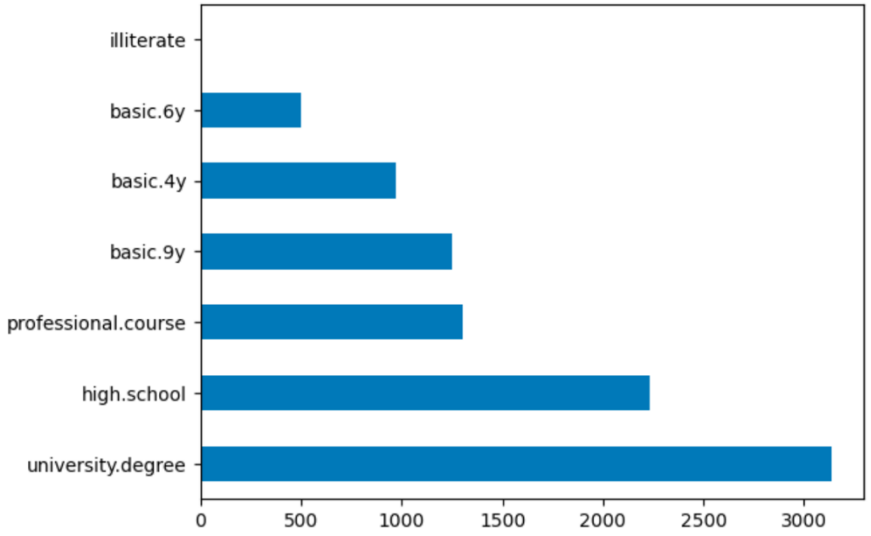
```
data.education.value_counts().plot.bar();
```

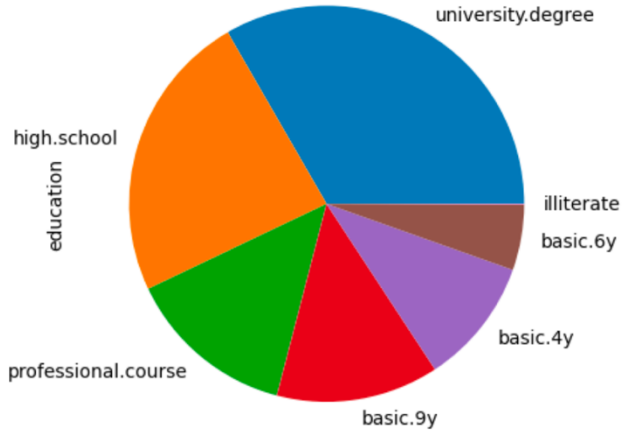


- die Kategorien sind nach abnehmenden Frequenzen angeordnet (Rückgabe von .value_counts())

Kategoriale Variablen

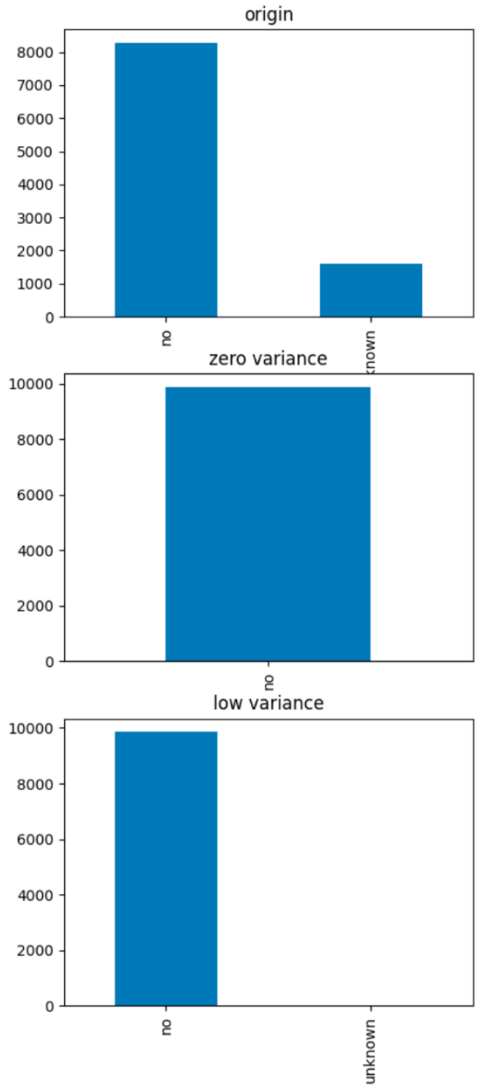
- weitere Visualisierungsmöglichkeiten (vgl. [ipynb])
  - links: horizontaler Barplot, besonders geeignet, wenn viele Kategorien vorliegen
  - rechts: Pie Plot, insbesondere bei Data Analytics nicht besonders beliebt: (Begründung)





Kategoriale Variablen

- von Zero- resp. Low Variance spricht man, wenn auf einer Variable keine oder nur eine minimale Varianz vorliegt
- dieses Konzept stammt zwar aus der Analyse numerischer Variablen, kann aber auch auf kategoriale Variablen angewendet werden
- die Visualisierung einer Simulation (vgl. [ipynb])
  - oben: das "Original" der Variable "default"
  - mitte: zero variance: alle Beobachtungen weisen für diese Variable denselben Wert auf,
    für Machine Learning kein Mehrwert
  - unten: low variance: ein Wert dominiert andere,
    für Machine Learning eher weniger geeignet,
    fachlich abzuklären



Numerische Variablen

- statistische Kennzahlen einer ausgewählten Variablen mit .describe()

```
print(data.age.describe())
```

```
count    9745.000000
mean       40.423191
std        11.915715
min        17.000000
25%        31.000000
50%        38.000000
75%        48.000000
max       116.000000
Name: age, dtype: float64
```

Numerische Variablen

count	Anzahl (nicht NAs) Werte
mean	Mittelwert (Arithmetisches Mittel)
std	Standardabweichung
min	Minimum, kleinster Wert (→ 0. Quartil)
25%	trennt kleinste 25% der Datenwerte vom Rest ab, → 1. Quartil
50%	trennt die unteren 50% der Datenwerte vom Rest ab, → 2. Quartil, Median
75%	trennt grösste 25% der Datenwerte vom Rest ab, → 3. Quartil
max	Maximum - grösster Wert (→ 4. Quartil)

- mean und std werden verwendet, um Normalverteilungsmodelle zu parametrisieren, sie werden daher auch als **parametrische** Kennzahlen bezeichnet
  ausserdem finden sie Verwendung zum Skalieren

- die übrigen bezeichnet man analog als **nichtparametrische** Kennzahlen (vgl. Boxplot)

Numerische Variablen

- dasselbe für alle numerischen Variablen des Data Frame

```
print(data.describe())
```

```
 age     duration     campaign         pdays     previous  \
count  9745.000000  9817.000000  9868.000000  9868.000000  9868.000000
mean     40.423191   378.159927     2.387617   895.442136     0.296109
std      11.915715   356.220783     2.512851   303.530529     0.678229
min      17.000000     0.000000     1.000000     0.000000     0.000000
25%      31.000000   139.000000     1.000000   999.000000     0.000000
50%      38.000000   258.000000     2.000000   999.000000     0.000000
75%      48.000000   508.000000     3.000000   999.000000     0.000000
max     116.000000  4199.000000    43.000000   999.000000     6.000000

       emp.var.rate  cons.price.idx  cons.conf.idx    euribor3m  nr.employed
count   9621.000000     9621.000000    9621.000000  9868.000000  9868.000000
:
```

Numerische Variablen

- im Gegensatz zu den kategorialen Variablen, wird hier mit .describe() die Anzahl Unique Values nicht ausgegeben
- für weiterführende Überlegungen kann es aber durchaus Sinn machen, diese Kennzahl ebenfalls zu ermitteln, insbesondere bei Integer Werten (aber nicht nur)
- dies kann z.B. mit untenstehendem Code erreicht werden

```
print(data[num_vars].nunique())
```

```
age                77
duration         1409
campaign           34
pdays              26
previous            7
emp.var.rate       10
:
```

Numerische Variablen

eine Nachbemerkung zu pandas.DataFrame Funktionen

- die bisher eingesetzten Zugriffe auf Funktionen von pandas.DataFrame nach der Form

  `<dataframe>.<variable>.<funktion()>`

  sind zwar praktisch, funktionieren aber nur, wenn der Name der Variablen den Namenskonventionen für Python-Objekte entspricht

- ist dies nicht der Fall, wie beispielsweise bei "nr.employed", muss die Variable mit dem traditionellen Zugriff (Spaltenindex) angesprochen werden

```
print(data.age.mean())              ## ok
#print(data.nr.employed.mean())     ## error
print(data['nr.employed'].mean()) ## ok
```

Numerische Variablen

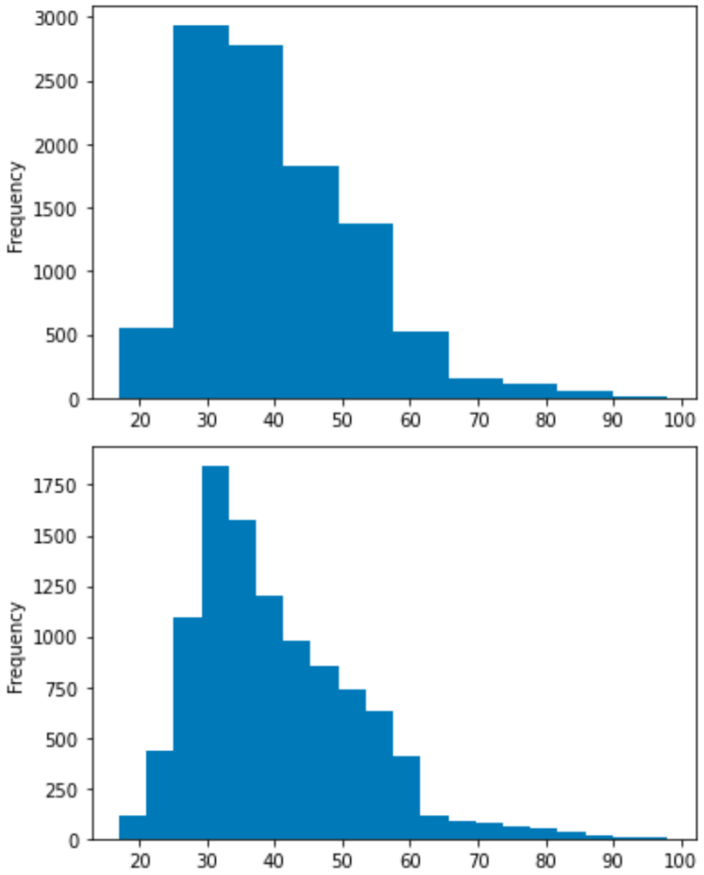
- Histogramm mit pandas.DataFrame.plot.hist(), dient der Beurteilung der Verteilung der Werte

```
data.age.plot.hist();
```

- per Vorgabe werden 10 Balken ("bins") erstellt, dies kann mit untenstehender Parametrisierung überschrieben werden

```
data.age.plot.hist(bins=20);
```

- Details zu optimaler Anzahl Bins später (Kap. 1.3.3.3 Binning)

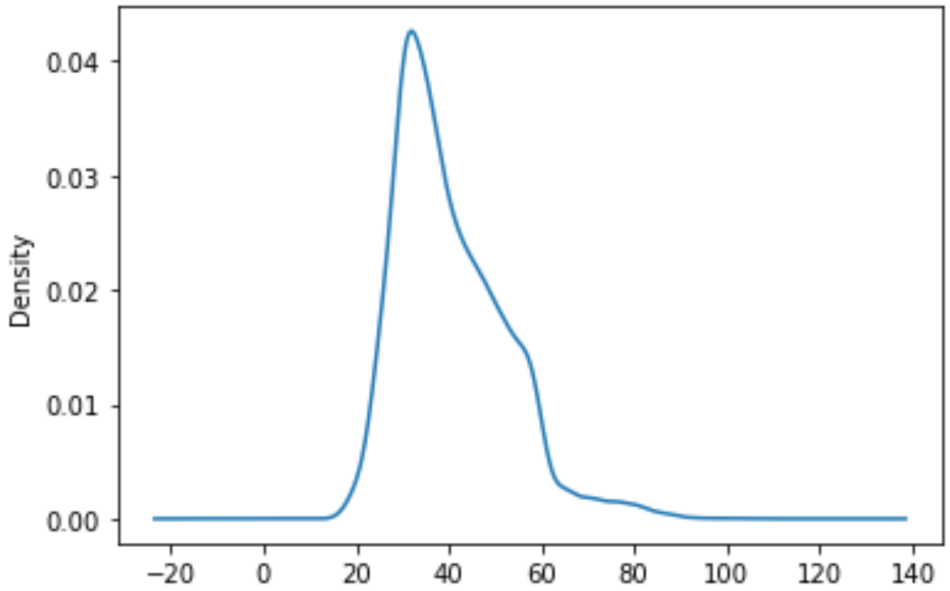


Numerische Variablen

- Densityplot mit pandas.DataFrame.plot.kde

```
data.age.plot.kde();
```

- kde steht für "Kernel Density Estimator"
  Hintergrund:
  - für jede Beobachtung wird eine Normalverteilung der Fläche 1/n erstellt und die einzelnen Verteilungen additiv überlagert
- zeigt ev. etwas bessere Details als Histogramm, insbesondere bei grösseren Datenmengen
- ist allerdings ressourcenhungrig



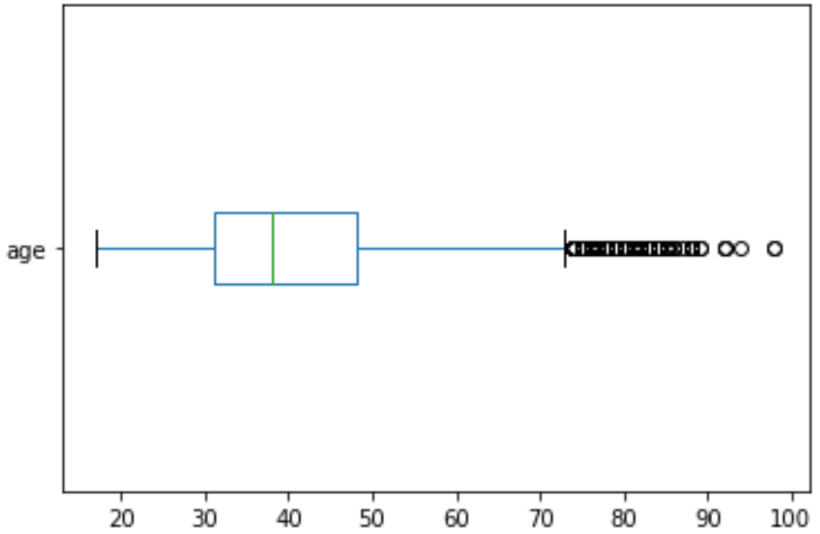
Numerische Variablen

- Boxplot mit pandas.DataFrame.boxplot()

```
data.age.plot.box(vert=False);
```

- stellt die nichtparametrischen Kennzahlen dar Minimum, 1. Quartil, Median, 3. Quartil, Maximum (vgl. 1.2.3.1)
- ausserdem Hinweise auf Ausreisser-verdächtige(!) Werte

- (die Ausrichtung ist per Default vertikal, daher wird bei der obigen Anweisung der Wert des Parameters `vert` überschrieben)

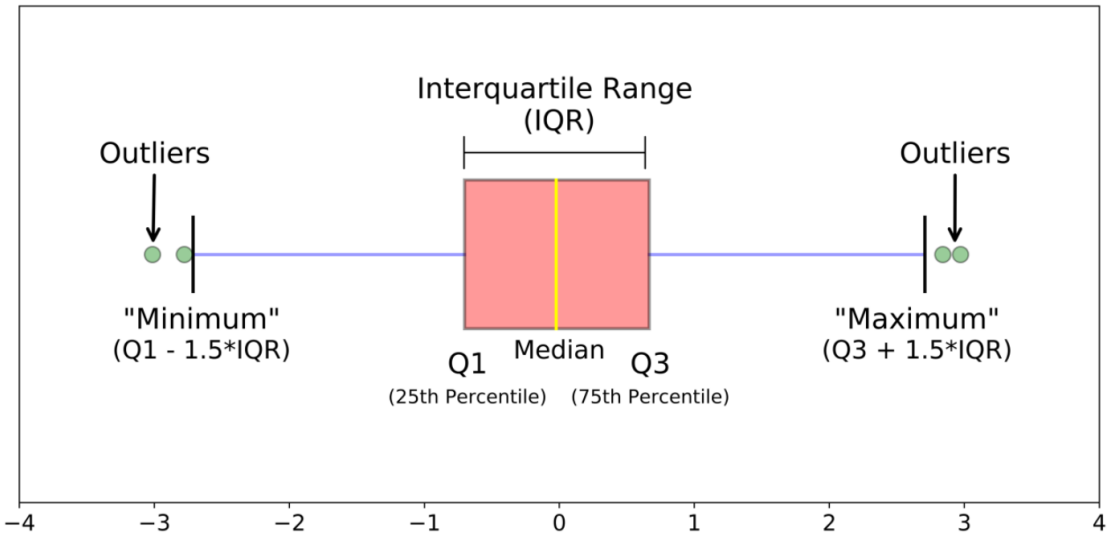


Numerische Variablen

Boxplot - einige Bemerkungen dazu:

(auch Box-Whisker-Plot genannt)

- die Box wird begrenzt durch das 1. und 3. Quartil
- innerhalb der Box wird die Position des Medians markiert
- als Whiskers (Schnauzhaare, Antennen) werden die Werte ausserhalb der Box dargestellt



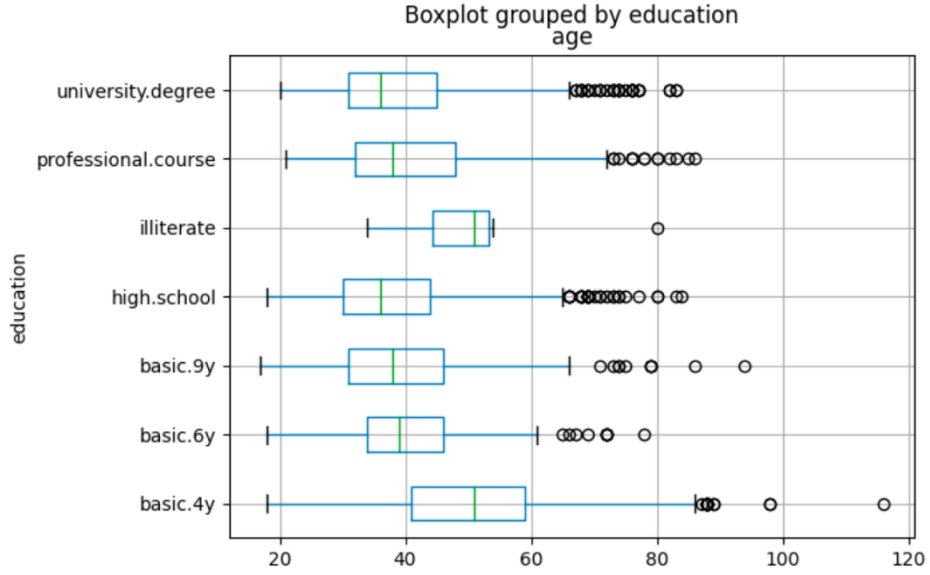
- als "Ausreisser" werden diejenigen Werte markiert, welche um
  - den 1.5 fachen IQR unterhalb des 1.
  - den 1.5 fachen IQR oberhalb des 3. Quartils liegen
  - IQR = 3. Quartil - 1. Quartil

Numerische Variablen

- Boxplots - gruppiert mit pandas.DataFrame.boxplot

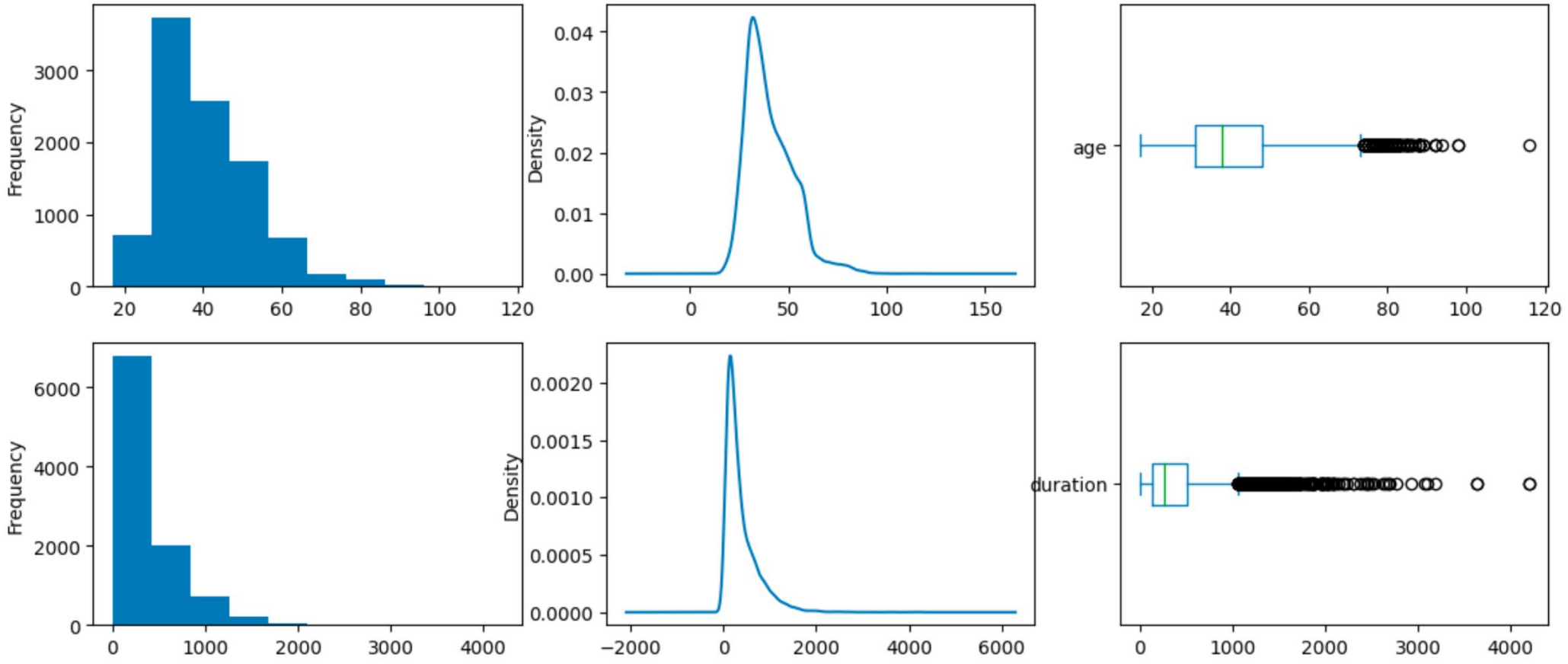
```
data.boxplot(column=['age'],
             by='education',
             vert=False);
```

- eine wichtige Anwendung von Boxplots: gruppenweise Gegenüberstellungen



Numerische Variablen

- eine Gegenüberstellung von hist - kde - box für "age" und "duration", vgl. [ipynb]



Numerische Variablen

- Verteilungen
  - beide Variablen sind rechts-schief verteilt, "age" wenig, "duration" dagegen stark
  - die Benennung (rechts, links) von schiefen Verteilungen richtet sich dabei nach der Position der Extremwerte
  - rechts-schiefe Verteilungen kommen in der Praxis oft vor, z.B. bei Einkommensstatistiken
  - sie können mittels logarithmieren oft in weniger schiefe Verteilungen transformiert werden

Numerische Variablen: Ausreiser

- ob es sich bei derartigen Randwerten effektiv um Extremwerte handelt, lässt sich auch feststellen, indem die kleinsten resp. grössten Werte sortiert ausgegeben werden:

```
print(data.age.sort_values().head(10))
```

```
7114    17.0
4083    17.0
8139    17.0
5790    18.0
433     18.0
```

```
print(data.age.sort_values(
    na_position="first").tail(10))
```

```
1138     92.0
2330     94.0
8345     98.0
3043     98.0
9867    116.0
```

- während der kleinste Wert (17) mehrmals vorkommt, setzt sich der grösste Wert (116) deutlich vom nächst kleineren ab
- ausserdem ist ein Alter von 116 auch fachlich recht unglaubwürdig

Interaktionen zwischen Variablen: kategorial

- wie bereits bei Sichtung der kategorialen Variablen, können mit einer Mehrweg-Frequenztabelle (Kreuzztabelle) allfällige Interaktionen zwischen zwei derartigen Variablen beurteilt werden
- als Beispiel der Vergleich von "job" und " duration "
  (ct als Objekt für spätere Weiterverwendung)

```
ct = pd.crosstab(
    data['job'],
    data['education'])
print(ct)
```

- welche Reihenfolge der Kategorien wird hier angezeigt?

```
education      basic.4y  basic.6y  basic.9y  high.school  illiterate  \
job
admin.               23        28       122          778           0
blue collar         442       299       689          217           2
entrepreneur         25        17        37           39           1
housemaid           122        17        19           45           0
management           11        20        31           58           0
retired             239        18        40           96           2
self-employed        24         3        46           21           1
services             25        58        82          563           0
student              11         8        43          142           0
technician           13        20        94          201           0
unemployed           27        10        44           67           0

education      professional.course  university.degree
job
admin.                          96               1513
blue collar                    106                 17
entrepreneur                    28                135
housemaid                       21                 34
management                      20                518
retired                         84                 94
self-employed                   39                194
services                        47                 45
student                         19                 53
technician                     804                441
unemployed                      41                 75
```

Interaktionen zwischen Variablen: kategorial

- mit untenstehendem Code kann die Ausgabe des resultierenden Data Frame zur besseren Interpretierbarkeit farblich formatiert werden:

```
ct.style.background_gradient(axis=None)
```

education	basic.4y	basic.6y	basic.9y	high.school	illiterate	professional.course	university.degree
job							
admin.	23	28	122	778	0	96	1513
blue collar	442	299	689	217	2	106	17
entrepreneur	25	17	37	39	1	28	135
housemaid	122	17	19	45	0	21	34
management	11	20	31	58	0	20	518
retired	239	18	40	96	2	84	94
self-employed	24	3	46	21	1	39	194
services	25	58	82	563	0	47	45
student	11	8	43	142	0	19	53
technician	13	20	94	201	0	804	441
unemployed	27	10	44	67	0	41	75

Interaktionen zwischen Variablen: kategorial

- im bisherigen Kursverlauf wurden für Visualisierungen ausschliesslich Funktionen von pandas.DataFrame.plot verwendet
  - Vorteile: einfache Aufrufe und Parametrisierungen, keine weiteren Libraries notwendig
  - Nachteile: wenig Flexibilität bei weiterer Ausgestaltung der Grafiken, keine komplexen Grafiken möglich
- eine umfassende Library zum Erstellen von mathematischen Visualisierungen aller Art ist matplotlib
  - dabei handelt es sich allerdings um ein Low-Level Interface, welches umfangreiche Parametrisierung und entsprechend viel Erfahrung erfordert (steile Lernkurve)
- seaborn bietet dagegen ein High-Level Interface, welches auf matplotlib basiert, aber wesentlich einfacher zu bedienen ist
- daher werden ab sofort Visualisierungen ausschliesslich mit seaborn erstellt
- Vorbereitung zum Arbeiten mit seaborn (falls nicht bereits geschehen)

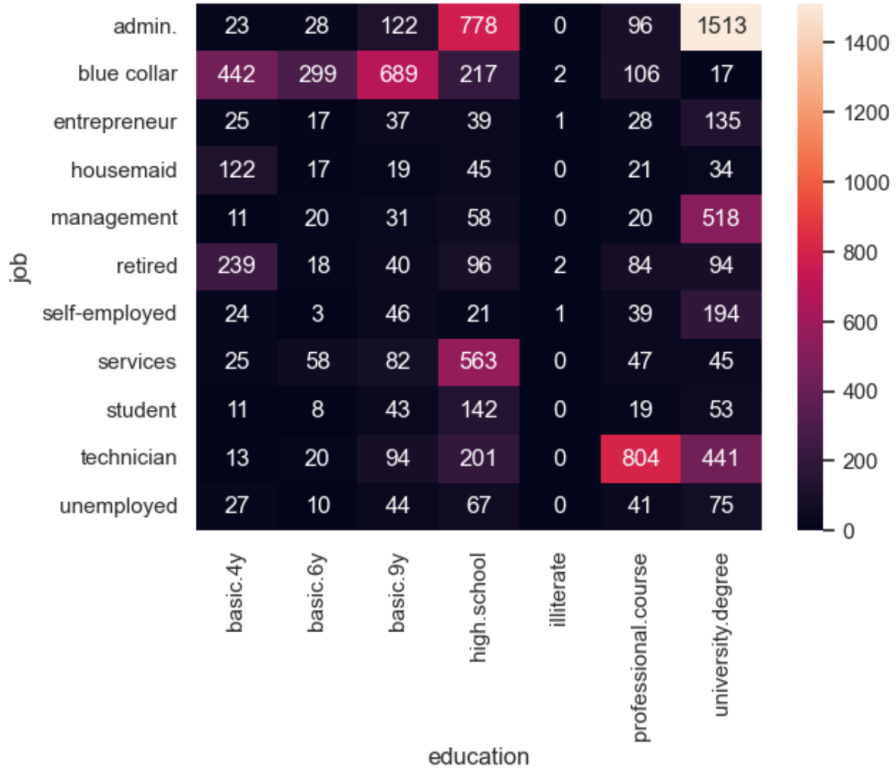
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

Interaktionen zwischen Variablen: kategorial

- im obigen Beispiel weist "job" 12 Kategorien auf, "education" deren 8
- Heatmap erleichtert die Interpretation einer derartigen Tabelle

```
sns.heatmap(
    ct,          ## crosstab
    annot=True, ## Frequenzen anzeigen
    fmt='d');    ## Zahlenformatierung
```

- (für die Parametrisierung der obigen Funktion, vgl. [ipynb])
- (wie ist wohl diese Farbgebung begründet?)
- CAS DA: mit dem Chi-quadrat Wert könnte man allenfalls Zusammenhänge zwischen Variable-Paaren quantifizieren,
  Vergleiche sind allerdings äusserst schwierig



Interaktionen zwischen Variablen: Korrelationskoeffizient

- Der Korrelationskoeffizient misst, wie stark zwei Variablen miteinander zusammenhängen – und in welche Richtung.
- Der Korrelationskoeffizient r liegt immer zwischen –1 und +1:
  - +1 → perfekter positiver Zusammenhang
  - 0 → kein linearer Zusammenhang
  - –1 → perfekter negativer Zusammenhang
- Positive Korrelation: Wenn eine Variable steigt, steigt die andere ebenfalls.
  - Beispiel: Körpergrösse und Gewicht.
- Negative Korrelation: Wenn eine Variable steigt, sinkt die andere.
  - Beispiel: Geschwindigkeit und Reisezeit.
- Keine Korrelation: Die Variablen hängen nicht linear zusammen.
  - Beispiel: Schuhgrösse und Intelligenz.

Interaktionen zwischen Variablen: Korrelationskoeffizient

- Feature Selection: Man prüft, ob Features stark mit der Target-Variable zusammenhängen.
- Explorative Datenanalyse: Man untersucht Muster und Beziehungen in Daten.
- Modellinterpretation: Man versteht, welche Variablen Einfluss haben könnten.

Interaktionen zwischen Variablen: numerisch

- zwischen zwei ausgewählten Variablen:
  (auch hier wird zwecks späterer Wiederverwendung das Ergebnis als Objekt hinterlegt)

```
corr = data[['age', 'duration']].corr()
print(corr)
```

```
                age  duration
age        1.000000 -0.017126
duration -0.017126  1.000000
```

- corr() als entsprechende Methode auf pandas.DataFrame implementiert
- die Werte auf der Diagonalen (von links oben nach rechts unten) betragen immer 1.0, da Identität = vollständige linearer Zusammenhang
- die Funktion ist kommutativ, d.h. $cor(x, y) = cor(y, x)$, d.h. die Werte werden an der Diagonale gespiegelt

Interaktionen zwischen Variablen: numerisch

- zwischen zwei ausgewählten Variablen:
  (auch hier wird zwecks späterer Wiederverwendung das Ergebnis als Objekt hinterlegt)

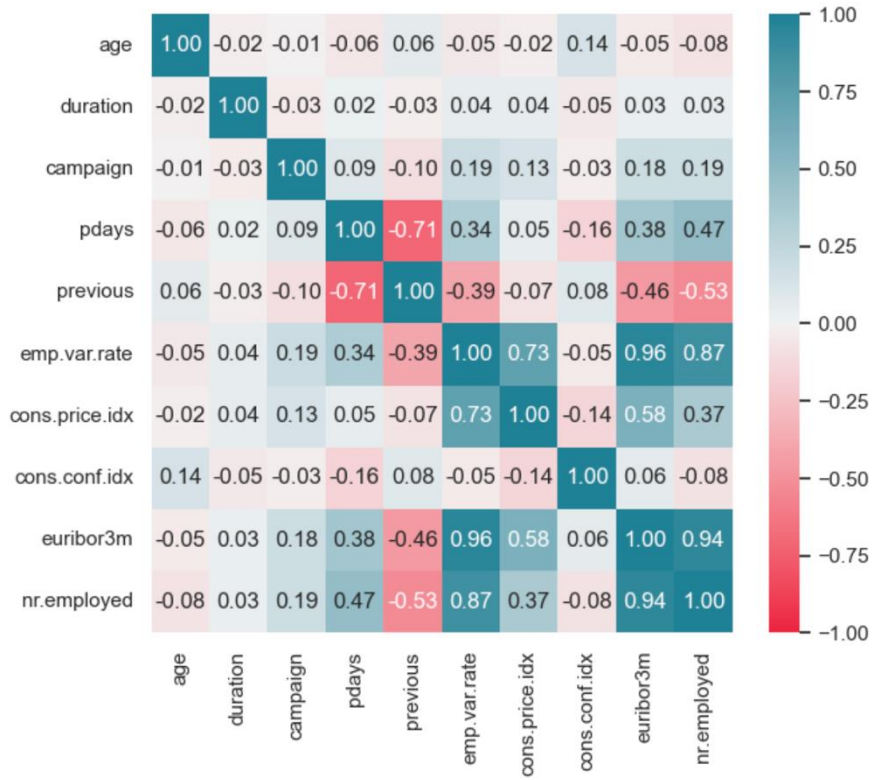
```
corr = data[['age', 'duration']].corr()
print(corr)
```

```
                age  duration
age        1.000000 -0.017126
duration  -0.017126  1.000000
```

- corr() als entsprechende Methode auf pandas.DataFrame implementiert
- die Werte auf der Diagonalen (von links oben nach rechts unten) betragen immer 1.0, da Identität = vollständige linearer Zusammenhang
- die Funktion ist kommutativ, d.h. $cor(x, y) = cor(y, x)$, d.h. die Werte werden an der Diagonale gespiegelt

Interaktionen zwischen Variablen: numerisch

```
plt.figure(figsize=(7, 6))
ax = sns.heatmap(
    corr, annot=True, fmt='.2f',
    xticklabels=corr.columns,
    yticklabels=corr.columns,
    cmap=sns.diverging_palette(
        10, 220, as_cmap=True),
    vmin=-1, vmax=1)
```



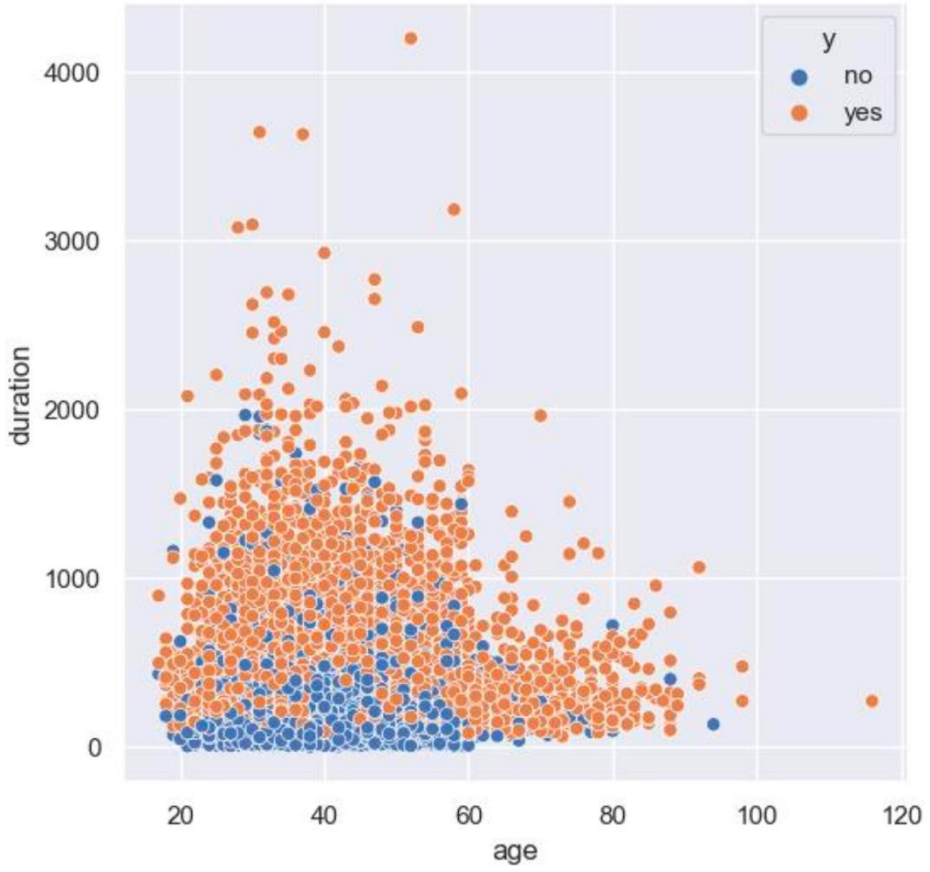
- die grössten Korrelationen findet man in der rechten unteren Ecke (makroökonomische Parameter), ist das plausibel?
- ([ipynb]: Code zum Unterdrücken der oberen Halbmatrix)

Interaktionen zwischen Variablen: numerisch

- Scatterplot, zwischen zwei ausgewählten Variablen

```
sns.scatterplot(
    x='age',
    y='duration',
    data=data,
    hue='y');
```

- in dieser Darstellung wurden (als Bereicherung) auch gleich noch eine Gruppenzugehörigkeit farblich dargestellt
- da sich hier die beiden Gruppen offensichtlich überlagern, kann es angebracht sein, für jede Gruppe einen separaten Scatterplot zu erstellen (vgl. [ipynb])



Lineare Zusammenhänge

- mit einer Simulation soll aufgezeigt werden, wie solche Zusammenhänge sichtbar gemacht werden können (vgl. [ipynb])
- Vorbereitung:
  - Auszug von vier der sozioökonimischen Variablen, der Einfachheit halber aber umbenannt auf "X1", "X2", "X3" und "X4"
  - erstellen einer neuen Variable "X5", welche linear abhängig von "X4" ist: vollständig linearer Zusammenhang
  - erstellen einer neuen Variable "X6", welche die Werte von "X5" enthält, aber mit etwas Rauschen überlagert wird: starker linearer Zusammenhang
  - berechnen einer Korrelationsmatrix auf den vorbereiteten Daten

Lineare Zusammenhänge

- da .corr() wiederum einen Pandas DataFrame erstellt, können mit der Methode .where() bestimmte Werte maskiert werden
- zwei Beispiele mit corr=1 und abs(corr)>=0.99

```
print(corr.where(abs(corr) == 1))
```

	X1	X2	X3	X4	X5	X6
X1	1.0	NaN	NaN	NaN	NaN	NaN
X2	NaN	1.0	NaN	NaN	NaN	NaN
X3	NaN	NaN	1.0	NaN	NaN	NaN
X4	NaN	NaN	NaN	1.0	1.0	NaN
X5	NaN	NaN	NaN	1.0	1.0	NaN
X6	NaN	NaN	NaN	NaN	NaN	1.0

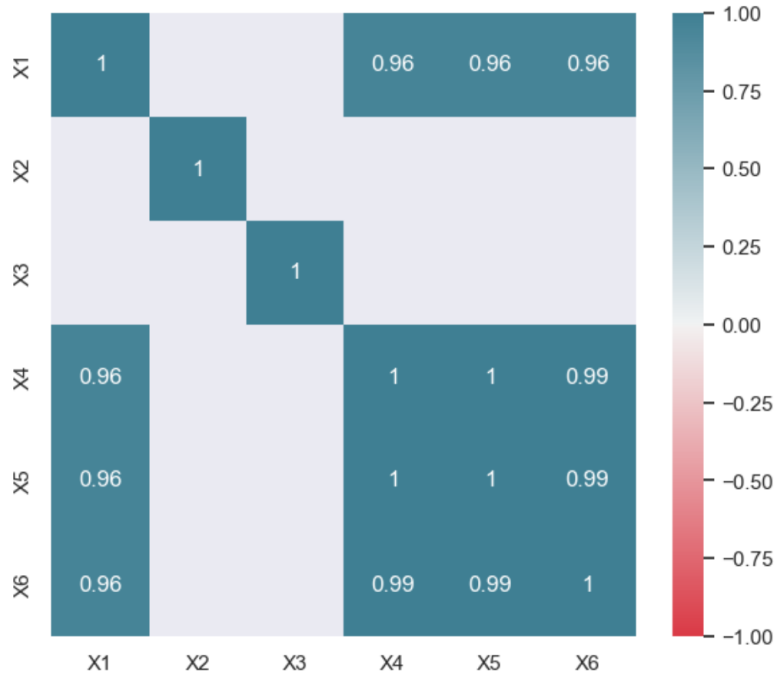
```
print(corr.where(abs(corr) >= 0.99))
```

	X1	X2	X3	X4	X5	X6
X1	1.0	NaN	NaN	NaN	NaN	NaN
X2	NaN	1.0	NaN	NaN	NaN	NaN
X3	NaN	NaN	1.0	NaN	NaN	NaN
X4	NaN	NaN	NaN	1.00	1.00	0.99
X5	NaN	NaN	NaN	1.00	1.00	0.99
X6	NaN	NaN	NaN	0.99	0.99	1.00

Lineare Zusammenhänge

- die "Maske" kann auch gleich eingesetzt werden, um bei einem Korrelogramm nicht interessierende Werte zu unterdrücken

```
mask = corr.where(abs(corr) >= 0.95)
plt.figure(figsize=(7, 6))
    ax = sns.heatmap(
        mask,
        annot=True,
        vmin=-1, vmax=1,
        cmap=sns.diverging_palette(
            10, 220, as_cmap=True));
```



Workshop 2: 1,5 Stunden

- untersuchen Sie das Melbourne Housing Dataset mit Sicht auf Machine Learning
  - achten Sie insbesondere auf folgende Punkte
  - welche Variablen sind zum vorneherein ungeeignet
  - Missing Values: welches wäre die jeweils geeignete Strategie zum Umgang damit
  - Nicht numerische Variablen: welches wäre die jeweils geeignete Strategie zum Nummerisieren
  - entdecken Sie noch andere Anomalien, welche zu bereinigen wären
  - formulieren Sie erste Empfehlungen zum Aufbereiten dieser Daten zuhanden Machine Learning
  - Hilfsmittel:
    - WS 02 Feature Engineering Exploration Overview.ipynb
      ermittelt die wichtigsten Kennwerte der Variablen in einem Data Frame und stellt diese in einer Excel-Tabelle zusammen
    - pandas_profiling_melb_data.html