Tutorial 1: Basic NLP Pipeline

Ziel des ersten Tutorial ist es, eine simple NLP Pipeline zu erstellen und sich dabei mit den Modulen nltk, numpy und scikit-learn vertraut zu machen.

Aufgabe 1: Importieren der Module

1.1: Import der wichtigsten NLP Module

Stellen Sie sicher, dass die benötigten Module "pandas", "numpy", "nltk", "sklearn" und "re" importiert sind und lassen Sie sich die Versionsnummern der Module ausgeben.

In [1]:

```
import pandas as pd
print ("pandas", pd.__version__)

import numpy as np
print ("numpy", np.__version__)

import nltk
print ("nltk", nltk.__version__)

import re
print ("re", re.__version__)

import sklearn
print ("sklearn", sklearn.__version__)
```

```
pandas 1.0.5
numpy 1.18.5
nltk 3.5
re 2.2.1
sklearn 0.21.3
```

1.2 Import von "Quality-of-Life Modulen"

Installieren sie "pandarrallel". Importieren sie die Methode "pandarrallen" von pandarrallel (für parallelization in pandas) und initialisieren sie es mit pandarallel.initialize().

In [2]:

```
# quality of life improvements
from pandarallel import pandarallel  # parallelization
pandarallel.initialize()
```

```
INFO: Pandarallel will run on 8 workers.
INFO: Pandarallel will use Memory file system to transfer data betwe
en the main process and workers.
```

2: Daten importieren

In diesem Tutorial werden Sie mit einen Dataset aus Witzen arbeiten. Diese wurden mithilfe von Crawlern von den Plattformen "stupidstuff.org", "wocka.com" sowie "reddit.com" gesammelt.

Lesen Sie die beigefügten .json-Dateien ein und führen Sie diese in einem Pandas-Dataframe zusammen. Stellen Sie dabei sicher, dass die Quelle der Daten als Key erhalten bleibt.

In [3]:

```
dfs = [pd.read_json("stupidstuff.json"), pd.read_json("reddit.json"), pd.read_
json("wocka.json")]
data = pd.concat(dfs, keys=["stupidstuff", "reddit", "wocka"])
data.loc["reddit"]
```

Out[3]:

	body	category	id	rating	score	title
0	Now I have to say "Leroy can you please paint ...	NaN	5tz52q	NaN	1.0	I hate how you cant even say black paint anymore
1	Pizza doesn't scream when you put it in the ov...	NaN	5tz4dd	NaN	0.0	What's the difference between a Jew in Nazi Ge...
2	...and being there really helped me learn abou...	NaN	5tz319	NaN	0.0	I recently went to America....
3	A Sunday school teacher is concerned that his ...	NaN	5tz2wj	NaN	1.0	Brian raises his hand and says, “He’s in Heaven.”
4	He got caught trying to sell the two books to ...	NaN	5tz1pc	NaN	0.0	You hear about the University book store worke...
...	...	...	...	...	...	...
40804	Mediocre meaty ochre.	NaN	4n0hy6	NaN	1.0	What do you call bad black paint made out of m...
40805	IHOP	NaN	4n0f4q	NaN	2.0	Where does a person with one leg work?
40806	Droid Sans	NaN	4n0ci8	NaN	0.0	What is Obi Wan's favorite font?
40807	It's the only place I can buy 400 cantaloupes ...	NaN	4n09ud	NaN	1.0	Even though I hate it, math is special.
40808	Feminem	NaN	4n08t5	NaN	2.0	What do you call a woman who raps about men be...

40809 rows × 6 columns

3: Data Preprocessing

Wie es sehr oft bei gecrawlten Daten der Fall ist, sind die Daten aus den unterschiedlichen Quellen sehr unterschiedlich beschaffen und getaggt.

3.1: Text bereinigen

Bereiten sie die Daten so auf, dass in der Spalte "body" jeweils der gesamte Text des Witzes enthalten ist und dass keine Format-Tokens (z.B. "\n") mehr enthalten sind

In [4]:

```
data.loc["reddit"]["body"] = data.loc["reddit"].parallel_apply(lambda row: row
["title"]+" "+row["body"], axis=1)
data.loc["reddit"]
```

```
<ipython-input-4-20f353993307>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py
  data.loc["reddit"]["body"] = data.loc["reddit"].parallel_apply(lam
bda row: row["title"]+" "+row["body"], axis=1)
```

Out[4]:

	body	category	id	rating	score	title
0	I hate how you cant even say black paint anymo...	NaN	5tz52q	NaN	1.0	I hate how you cant even say black paint anymore
1	What's the difference between a Jew in Nazi Ge...	NaN	5tz4dd	NaN	0.0	What's the difference between a Jew in Nazi Ge...
2	I recently went to America.... ...and being th...	NaN	5tz319	NaN	0.0	I recently went to America....
3	Brian raises his hand and says, “He’s in Heave...	NaN	5tz2wj	NaN	1.0	Brian raises his hand and says, “He’s in Heaven.”
4	You hear about the University book store worke...	NaN	5tz1pc	NaN	0.0	You hear about the University book store worke...
...	...	...	...	...	...	...
40804	What do you call bad black paint made out of m...	NaN	4n0hy6	NaN	1.0	What do you call bad black paint made out of m...
40805	Where does a person with one leg work? IHOP	NaN	4n0f4q	NaN	2.0	Where does a person with one leg work?
40806	What is Obi Wan's favorite font? Droid Sans	NaN	4n0ci8	NaN	0.0	What is Obi Wan's favorite font?
40807	Even though I hate it, math is special. It's t...	NaN	4n09ud	NaN	1.0	Even though I hate it, math is special.
40808	What do you call a woman who raps about men be...	NaN	4n08t5	NaN	2.0	What do you call a woman who raps about men be...

40809 rows × 6 columns

In [5]:

```
data.loc["wocka"]["body"] = data.loc["wocka"].parallel_apply(lambda row: re.sub("(\n)|(\r)", " ", row["body"]), axis=1)
data.loc["wocka"]
```

```
<ipython-input-5-5502645b5ae3>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data.loc["wocka"]["body"] = data.loc["wocka"].parallel_apply(lambda row: re.sub("(\n)|(\r)", " ", row["body"]), axis=1)
```

Out[5]:

	body	category	id	rating	score	title
0	What do you call a cow with no legs? Ground...	Animal	1	NaN	NaN	Cow With No Legs
1	What do you call a cow jumping over a barbed w...	Animal	2	NaN	NaN	Jumping Cow
2	What's black and white and red all over? A ...	Other / Misc	4	NaN	NaN	Black, White and Red
3	So, this guy walks into a bar. And says, "o...	Bar	5	NaN	NaN	Guy in a Bar
4	If the opposite of pro is con, isn't the oppos...	One Liners	6	NaN	NaN	Progress
...	...	...	...	...	...	...
10014	(A man comes to my register with a mint chocol...	Men / Women	18196	NaN	NaN	Hell Hath No Fury Like A Pregnant Woman Scorned
10015	(I am shelving DVDs in a library when a man co...	Children	18197	NaN	NaN	No Pranks, Just Thanks
10016	Me: "That will be 17.50, please." Customer:...	Religious	18198	NaN	NaN	Hell In A Handbag
10017	Me: "Sir, would you like to use any coupons to...	At Work	18199	NaN	NaN	A Good Ol' Fashioned A** Whoopin'
10018	Customer: "Are you Hispanic?" Me: "No." ...	Insults	18200	NaN	NaN	... And We Wonder Why Everyone Hates Us

10019 rows × 6 columns

3.2: Tokenization

Nutzen sie das nltk Modul um die Witze zu Tokenisieren. Nutzen sie dazu den RegexpTokenizer des nltk Moduls, um mit einem passenden regulären Ausdruck nur Tokens aus Wörtern und Zahlen zu übernehmen (also keine Satzzeichen). Die Tokens sollen nur kleine Buchstaben enthalten. Speichern Sie die Ergebnisse in einer Spalte "tokens".

In [6]:

```
from nltk.tokenize import RegexpTokenizer
tokenizer = RegexpTokenizer(r'\w+')

data["tokens"] = data.apply(lambda row: tokenizer.tokenize(str(row["body"].lower())), axis=1)
```

In [7]:

```
data.loc["stupidstuff"]
```

Out[7]:

	body	category	id	rating	score	title	tokens
0	A blackjack dealer and a player with a thirtee...	Children	1	2.63	NaN	NaN	[a, blackjack, dealer, and, a, player, with, a...
1	At a dinner party, several of the guests were ...	Blonde Jokes	2	2.57	NaN	NaN	[at, a, dinner, party, several, of, the, guest...
2	One day this cop pulls over a blonde for speed...	Blonde Jokes	3	3.09	NaN	NaN	[one, day, this, cop, pulls, over, a, blonde, ...
3	Three women are about to be executed for crime...	Blonde Jokes	4	4.10	NaN	NaN	[three, women, are, about, to, be, executed, f...
4	A girl came skipping home FROM school one day....	Blonde Jokes	5	4.30	NaN	NaN	[a, girl, came, skipping, home, from, school, ...
...	...	...	...	...	...	...	...
3768		Miscellaneous	3769	5.00	NaN	NaN	[]
3769	The Pope and the Queen of England are on the s...	Miscellaneous	3770	4.00	NaN	NaN	[the, pope, and, the, queen, of, england, are,...
3770		Miscellaneous	3771	1.00	NaN	NaN	[]
3771	Letter to Xerox and the Reply\n\nDear Kings of...	Miscellaneous	3772	4.00	NaN	NaN	[letter, to, xerox, and, the, reply, dear, kin...
3772	Note: Tradewars is on online role-playing game...	Miscellaneous	3773	3.00	NaN	NaN	[note, tradewars, is, on, online, role, playin...

3773 rows × 7 columns

3.3 Stopword removal

Entfernen sie alle englischen Stopwords aus den erzeugten Tokens.

In [8]:

```
def remove_stopwords_from_list(list_in, language):
    return [a for a in list_in if a not in nltk.corpus.stopwords.words(languag
e)]
```

In [9]:

```
data["tokens"] = data.parallel_apply(lambda row: remove_stopwords_from_list(ro
w["tokens"], "english"), axis=1)
```

3.4 POS Tagging

Bestimmen Sie POS-Tags für die tokenisierten Texte und speichern Sie diese in einer Spalte "pos".

In [10]:

```
data["pos"] = data.parallel_apply(lambda row: nltk.pos_tag(row["tokens"]), axi
s=1)
```

In [11]:

```
data.loc["reddit"]["pos"][1][1][1]
```

Out[11]:

```
'NN'
```

3.5 Lemmatisierung

Lemmatisieren sie die Tokens der Texte und speichern sie die bestimmten Lemmata in einer Spalte "Lemmata".

BONUS: Berücksichtigen sie bei der Lemmatisierung die Wortformen der Tokens.

In [12]:

```
# Simple solution:
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
data["lemmata"] = data.parallel_apply(lambda row: [lemmatizer.lemmatize(word)
for word in row["tokens"]], axis=1)
```

In [13]:

```
data
```

Out[13]:

		body	category	id	rating	score	title	tokens	
stupidstuff	**0**	A blackjack dealer and a player with a thirtee...	Children	1	2.63	NaN	NaN	[blackjack, dealer, player, thirteen, count, h...	[(blackj (de (pla NN
	1	At a dinner party, several of the guests were ...	Blonde Jokes	2	2.57	NaN	NaN	[dinner, party, several, guests, arguing, whet...	[(din (p (sev JJ), (
	2	One day this cop pulls over a blonde for speed...	Blonde Jokes	3	3.09	NaN	NaN	[one, day, cop, pulls, blonde, speeding, cop, ...	[(one, ((day, ((cop, (p NN
	3	Three women are about to be executed for crime...	Blonde Jokes	4	4.10	NaN	NaN	[three, women, executed, crimes, one, brunette...	[(th ((wor N (execu VBD
	4	A girl came skipping home FROM school one day....	Blonde Jokes	5	4.30	NaN	NaN	[girl, came, skipping, home, school, one, day,...	[(girl, ((ca V (skipp V (
...	**...**	...	...	...	...	...	...	...	
wocka	**10014**	(A man comes to my register with a mint chocol...	Men / Women	18196	NaN	NaN	Hell Hath No Fury Like A Pregnant Woman Scorned	[man, comes, register, mint, chocolate, candy,...	[(r (cor V (regi JJ), (n
	10015	(I am shelving DVDs in a library when a man co...	Children	18197	NaN	NaN	No Pranks, Just Thanks	[shelving, dvds, library, man, comes, boy, app...	[(shelv V (dvds, (lib JJ
	10016	Me: "That will be 17.50, please." Customer:...	Religious	18198	NaN	NaN	Hell In A Handbag	[17, 50, please, customer, christian, dear, as...	[(17, ((50, ((ple (custo
	10017	Me: "Sir, would you like to use any coupons to...	At Work	18199	NaN	NaN	A Good Ol' Fashioned A** Whoopin'	[sir, would, like, use, coupons, today, custom...	[(sir, ((wo MD), (VB), (N

	body	category	id	rating	score	title	tokens	
10018	Customer: "Are you Hispanic?" Me: "No." ...	Insults	18200	NaN	NaN	... And We Wonder Why Everyone Hates Us	[customer, hispanic, customer, middle, eastern...	[(custo (hispa (custo N

54601 rows × 9 columns

In [14]:

```
# Lemmatization with word type from https://www.machinelearningplus.com/nlp/lemmatization-examples-python/
# Lemmatize with POS Tag
from nltk.corpus import wordnet
lemmatizer = WordNetLemmatizer()

def get_wordnet_pos(word):
    """Map POS tag to first character lemmatize() accepts"""
    tag = nltk.pos_tag([word])[0][1][0].upper()
    tag_dict = {"J": wordnet.ADJ,
                "N": wordnet.NOUN,
                "V": wordnet.VERB,
                "R": wordnet.ADV}
    return tag_dict.get(tag, wordnet.NOUN)


data["lemmata_word_type"] = data.parallel_apply(lambda row: [lemmatizer.lemmatize(w, get_wordnet_pos(w)) for w in row["tokens"]], axis=1)
```

3.6 Frequencies

Fügen sie in einer neuen Spalte "frequencies" die Häufigkeitsverteilungen der lemmatisierten Tokens für jeden Text hinzu.

In [15]:

```
from nltk.probability import FreqDist
data["frequencies"] = data.parallel_apply(lambda row: FreqDist(row["lemmata_word_type"]), axis=1)
```

4 Data Analysis

4.1 Überblick über Themen

Lassen Sie sich die für die von Stupidstuff und Wocker gecrawlten Witze die Kategorien als Liste ausgeben

In [16]:

```
data.loc["stupidstuff"].category.unique().tolist()
```

Out[16]:

```
['Children',
 'Blonde Jokes',
 'Military',
 'Office Jokes',
 'Aviation',
 'Political',
 'Deep Thoughts',
 'Men',
 'Crazy Jokes',
 'Medical',
 'Food Jokes',
 'Bar Jokes',
 'Science',
 'Police Jokes',
 'Miscellaneous',
 'Sex',
 'Idiots',
 'Business',
 'Women',
 'Redneck',
 'One Liners',
 'Money',
 'School',
 'Family, Parents',
 'Sports',
 'Heaven and Hell',
 'Religious',
 'Farmers',
 'Love & Romance',
 'Blind Jokes',
 'Marriage',
 'Old Age',
 'Animals',
 'Holidays',
 'Ethnic Jokes',
 'State Jokes',
 'English',
 'Computers',
 'Lawyers',
 'Yo Mama',
 'Insults',
 'Light Bulbs',
 'Music']
```

In [17]:

```
data.loc["wocka"].category.unique().tolist()
```

Out[17]:

```
['Animal',
 'Other / Misc',
 'Bar',
 'One Liners',
 'Puns',
 'Lawyer',
 'Sports',
 'Medical',
 'News / Politics',
 'Men / Women',
 'Gross',
 'Blond',
 'Yo Momma',
 'Redneck',
 'Religious',
 'At Work',
 'College',
 'Lightbulb',
 'Children',
 'Insults',
 'Knock-Knock',
 'Tech',
 'Yo Mama',
 'Blonde']
```

In [ ]:

```

```

4.2 Überblick über numerische Werte

Lassen sie sich die durchschnittlichen Bewertungen für die Kategorien der Witze von Stupidstuff aufsteigendsortiert ausgeben. Lassen Sie sich anschließend mithilfe van Pandas einen überblick über deskriptive Statistiken der Stupidstuff Witze ausgeben, erneut nach Kategorien gruppiert.

In [18]:

```
data.loc["stupidstuff"].groupby(["category"]).mean().sort_values(by=['categor
y'])
```

Out[18]:

category	rating	score
Animals	2.902519	NaN
Aviation	3.134286	NaN
Bar Jokes	3.395632	NaN
Blind Jokes	3.249091	NaN
Blonde Jokes	3.522973	NaN
Business	3.415288	NaN
Children	3.494390	NaN
Computers	3.282333	NaN
Crazy Jokes	3.275581	NaN
Deep Thoughts	3.630000	NaN
English	3.500000	NaN
Ethnic Jokes	3.437500	NaN
Family, Parents	3.140937	NaN
Farmers	3.197586	NaN
Food Jokes	2.722308	NaN
Heaven and Hell	3.369121	NaN
Holidays	3.393623	NaN
Idiots	3.390968	NaN
Insults	3.460714	NaN
Lawyers	3.045000	NaN
Light Bulbs	3.248583	NaN
Love & Romance	3.152292	NaN
Marriage	3.397838	NaN
Medical	3.270000	NaN
Men	3.558737	NaN
Military	3.376724	NaN
Miscellaneous	3.133959	NaN
Money	3.340122	NaN
Music	2.515556	NaN
Office Jokes	3.239444	NaN
Old Age	3.047273	NaN
One Liners	3.446429	NaN
Police Jokes	3.397164	NaN
Political	3.292270	NaN
Redneck	3.691481	NaN
Religious	3.417632	NaN

category	rating	score
School	3.201613	NaN
Science	3.417778	NaN
Sex	3.610741	NaN
Sports	3.293115	NaN
State Jokes	3.433000	NaN
Women	3.329167	NaN
Yo Mama	3.152624	NaN

In [19]:

```
data.loc["stupidstuff"].groupby(["category"]).describe()
```

Out[19]:

	rating								score		
	count	mean	std	min	25%	50%	75%	max	count	mean	s
category											
Animals	131.0	2.902519	1.534843	0.00	2.0000	3.000	4.0000	5.00	0.0	NaN	N
Aviation	35.0	3.134286	1.014438	1.00	2.2650	3.000	4.0000	5.00	0.0	NaN	N
Bar Jokes	87.0	3.395632	1.269991	1.00	2.6700	3.800	4.2700	5.00	0.0	NaN	N
Blind Jokes	11.0	3.249091	1.012951	2.00	2.2150	3.500	3.8150	5.00	0.0	NaN	N
Blonde Jokes	111.0	3.522973	0.787498	1.00	3.0000	3.600	4.0000	5.00	0.0	NaN	N
Business	104.0	3.415288	1.165429	1.00	2.4575	3.000	4.4250	5.00	0.0	NaN	N
Children	82.0	3.494390	1.209082	0.00	2.8125	3.670	4.6275	5.00	0.0	NaN	N
Computers	150.0	3.282333	1.236951	0.00	2.0000	3.000	4.0000	5.00	0.0	NaN	N
Crazy Jokes	43.0	3.275581	1.261383	0.00	2.5000	3.000	4.5000	5.00	0.0	NaN	N
Deep Thoughts	14.0	3.630000	0.760455	2.00	3.2550	3.855	4.0750	4.69	0.0	NaN	N
English	1.0	3.500000	NaN	3.50	3.5000	3.500	3.5000	3.50	0.0	NaN	N
Ethnic Jokes	8.0	3.437500	1.285221	1.50	2.9475	3.835	4.1000	5.00	0.0	NaN	N
Family, Parents	96.0	3.140937	1.232906	0.00	2.0000	3.000	4.0000	5.00	0.0	NaN	N
Farmers	29.0	3.197586	1.244696	1.00	2.0000	3.750	4.0000	5.00	0.0	NaN	N
Food Jokes	13.0	2.722308	1.307473	1.00	1.5000	3.000	3.4000	5.00	0.0	NaN	N
Heaven and Hell	91.0	3.369121	1.144869	1.00	2.6700	3.500	4.0000	5.00	0.0	NaN	N
Holidays	69.0	3.393623	1.241305	1.00	2.5000	3.000	4.5000	5.00	0.0	NaN	N
Idiots	31.0	3.390968	1.077313	2.00	2.5500	3.000	4.5000	5.00	0.0	NaN	N
Insults	196.0	3.460714	1.205303	1.00	2.0000	4.000	4.2500	5.00	0.0	NaN	N
Lawyers	50.0	3.045000	1.165920	1.00	2.0000	3.000	4.0000	5.00	0.0	NaN	N
Light Bulbs	120.0	3.248583	1.283193	1.00	2.0000	3.000	4.0000	5.00	0.0	NaN	N
Love & Romance	48.0	3.152292	1.155943	1.00	2.4000	3.125	4.0000	5.00	0.0	NaN	N
Marriage	37.0	3.397838	1.186953	1.00	2.5000	4.000	4.0000	5.00	0.0	NaN	N
Medical	101.0	3.270000	1.194911	0.00	2.5000	3.250	4.0000	5.00	0.0	NaN	N
Men	190.0	3.558737	0.908827	1.00	3.0000	3.580	4.1400	5.00	0.0	NaN	N
Military	58.0	3.376724	1.005409	1.00	2.8125	3.365	4.0000	5.00	0.0	NaN	N
Miscellaneous	831.0	3.133959	1.225368	1.00	2.0000	3.000	4.0000	5.00	0.0	NaN	N
Money	82.0	3.340122	1.167725	1.00	2.3725	3.500	4.1500	5.00	0.0	NaN	N
Music	9.0	2.515556	1.249761	1.00	1.3300	2.670	3.0000	4.50	0.0	NaN	N
Office Jokes	18.0	3.239444	1.220793	1.00	2.7525	3.290	4.1800	5.00	0.0	NaN	N
Old Age	22.0	3.047273	1.002702	1.50	2.2750	3.000	3.6875	5.00	0.0	NaN	N
One Liners	28.0	3.446429	1.116702	2.00	2.3750	3.500	4.0000	5.00	0.0	NaN	N
Police Jokes	67.0	3.397164	1.312916	0.00	2.1250	3.330	5.0000	5.00	0.0	NaN	N

	rating								score		
	count	mean	std	min	25%	50%	75%	max	count	mean	s
category											
Political	141.0	3.292270	1.342550	0.00	2.0000	3.500	4.0000	5.00	0.0	NaN	N
Redneck	27.0	3.691481	0.876750	1.00	3.0000	4.000	4.1850	5.00	0.0	NaN	N
Religious	152.0	3.417632	1.114739	1.00	2.9575	3.500	4.0000	5.00	0.0	NaN	N
School	62.0	3.201613	1.342424	1.00	2.0000	3.000	4.2500	5.00	0.0	NaN	N
Science	18.0	3.417778	1.097503	1.00	3.0000	3.605	4.0000	5.00	0.0	NaN	N
Sex	54.0	3.610741	1.055910	1.00	3.0000	3.765	4.4900	5.00	0.0	NaN	N
Sports	61.0	3.293115	1.251026	0.00	2.5000	3.000	4.2500	5.00	0.0	NaN	N
State Jokes	10.0	3.433000	0.983136	1.75	3.0000	3.500	4.0000	5.00	0.0	NaN	N
Women	144.0	3.329167	1.155639	1.00	2.5000	3.585	4.0000	5.00	0.0	NaN	N
Yo Mama	141.0	3.152624	1.179632	1.00	2.3000	3.000	4.0000	5.00	0.0	NaN	N

5 Bonus

Analysieren oder bearbeiten Sie einen Aspekt ihrer Wahl des Datensatzes. Beispielsweise können Sie für jede der Plattformen den Anteil der Witze berechnen, die ein Bestimmtes Wort oder Wörter einer bestimmten Wortgruppe enthalten, Gesamtworthäufigkeiten der verschiedenen Quellen oder Genres berechnen oder eine andere beliebige Fragestellung bearbeiten, die ohne weitere Module zu importieren umsetzbar ist.