pandas_intro

April 27, 2020

1 Úvod do Pandas

Potřeba nainstalovat jako modul do Pythonu. Pokud ještě nemáte, do terminálu (ne do Pythonovské konzole) napište: * na Windows `pip install pandas`, * na Linuxu nebo Macu `pip3 install pandas`.

Pandas: * Je základní knihovna pro práci s daty v Pythonu. * Velká část datové analýzy obnáší právě práci s Pandas. * Datový soubor reprezentuje podobně jako tabulka v databázích nebo v Excelu, "DataFrame". * Stejně jako jsme se učili seznamy a slovníky, DataFrame je další datový typ pro ukládání dat.

1.1 1. Základní práce s DataFrame

1.1.1 Načítání dat

- Pandas umí načítat všechny možné formáty – CSV, JSON, Excel, HTML, SQL databáze, a spoustu dalších, stejně tak je možné do nich ukládat.
  - Toho lze využít např. pro převod dat z Excelu do CSV.
- Rovněž umožňují stáhnout data z internetu – místo cesty k souboru na disku dáme URL adresu.
- Podporují kompresi – ZIP archivy a podobně.

[1]:
```
import pandas
```

Příklad: tabulka měst provozujících tramvajovou dopravu.

[2]:
```
!cat mesta.csv
```

```
praha,PHA,1 294 513,24,496.00
```

[3]:
```
mesta = pandas.read_csv("mesta.csv", index_col="mesto", encoding="utf-8")
mesta
```

[3]:

mesto	kraj	obyvatel	linky	vymera
brno	JHM	379 527	22	230.22
liberec	LBK	103 979	6	106.09
litvinov	ULK	24 143	5	40.70
most	ULK	66 644	5	86.94
olomouc	OLK	100 494	7	103.36
ostrava	MSK	290 450	15	214.23
plzen	PLK	170 936	3	137.65
praha	PHA	1 294 513	24	496.00

1.1.2 Základní informace o tabulce

Datové typy, názvy sloupců, počet neprázdných hodnot.

[4]:
```
mesta.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 8 entries, brno to praha
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   kraj      8 non-null      object
 1   obyvatel  8 non-null      object
 2   linky     8 non-null      int64
 3   vymera    8 non-null      float64
dtypes: float64(1), int64(1), object(2)
memory usage: 320.0+ bytes
```

Velikost.

[5]:
```
mesta.shape
```

[5]:
```
(8, 4)
```

Názvy sloupců.

[6]:
```
mesta.columns
```

[6]:
```
Index(['kraj', 'obyvatel', 'linky', 'vymera'], dtype='object')
```

[7]:
```
list(mesta)
```

[7]:
```
['kraj', 'obyvatel', 'linky', 'vymera']
```

Základní statistické údaje – jen na číselných sloupcích.

[8]:
```
mesta.describe()
```

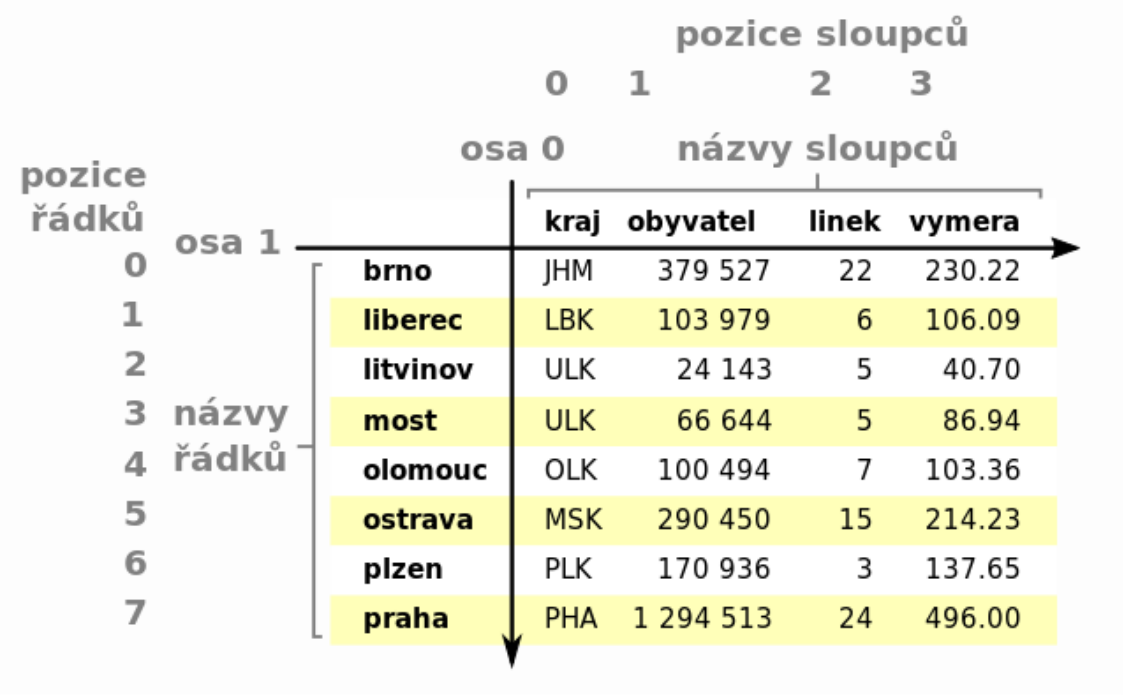
[8]:
```
            linky      vymera
count    8.000000    8.000000
mean    10.875000  176.898750
std      8.305549  143.759399
min      3.000000   40.700000
25%      5.000000   99.255000
50%      6.500000  121.870000
75%     16.750000  218.227500
max     24.000000  496.000000
```

Prvních pár řádků tabulky.

[9]:
```
mesta.head()
```

[9]:
```
          kraj obyvatel  linky  vymera
mesto
brno       JHM  379 527     22  230.22
liberec    LBK  103 979      6  106.09
litvinov   ULK   24 143      5   40.70
most       ULK   66 644      5   86.94
olomouc    OLK  100 494      7  103.36
```

1.2 2. Základní selekce



Výběr hodnot v tabulce probíhá převážně pomocí "metody" `loc`.

1.2.1 Výběr podle jmen řádků a sloupců

1. Pouze řádky

[10]:
```
mesta.loc["brno"]
```

[10]:
```
kraj            JHM
obyvatel    379 527
linky            22
vymera       230.22
Name: brno, dtype: object
```

[11]:
```
mesta.loc[["brno", "praha", "ostrava"]]
```

[11]:

mesto	kraj	obyvatel	linky	vymera
brno	JHM	379 527	22	230.22
praha	PHA	1 294 513	24	496.00
ostrava	MSK	290 450	15	214.23

V prvním případě jsme dostali výsledek orientovaný na výšku, ve druhém na šířku. Poprvé jsme chtěli údaj o 1 městu a dostali tzv. `Series`, podruhé jsme chtěli údaj o více městech a dostali `DataFrame` (podmnožinu toho původního DataFrame `mesta`).

Mohli bychom si taky explicitně říct o výsledek pro 1 město v podobě DataFrame.

[12]:
```
mesta.loc[["brno"]]
```

[12]:

mesto	kraj	obyvatel	linky	vymera
brno	JHM	379 527	22	230.22

Lze použít také rozsah (záleží samozřejmě na pořadí, v jakém máme data uložena).

[13]:
```
mesta.loc["most":"praha"]
```

[13]:

mesto	kraj	obyvatel	linky	vymera
most	ULK	66 644	5	86.94
olomouc	OLK	100 494	7	103.36
ostrava	MSK	290 450	15	214.23
plzen	PLK	170 936	3	137.65
praha	PHA	1 294 513	24	496.00

Pozor na pořadí.

[14]:
```
mesta.loc["praha":"most"]
```

[14]:
```
Empty DataFrame
Columns: [kraj, obyvatel, linky, vymera]
Index: []
```

[15]:
```
mesta.loc["most":"praha":2]
```

[15]:

mesto	kraj	obyvatel	linky	vymera
most	ULK	66 644	5	86.94
ostrava	MSK	290 450	15	214.23
praha	PHA	1 294 513	24	496.00

[16]:
```
mesta.loc[:"most"]
```

[16]:

mesto	kraj	obyvatel	linky	vymera
brno	JHM	379 527	22	230.22
liberec	LBK	103 979	6	106.09
litvinov	ULK	24 143	5	40.70
most	ULK	66 644	5	86.94

[17]:
```
mesta.loc["most":]
```

[17]:

mesto	kraj	obyvatel	linky	vymera
most	ULK	66 644	5	86.94
olomouc	OLK	100 494	7	103.36
ostrava	MSK	290 450	15	214.23
plzen	PLK	170 936	3	137.65
praha	PHA	1 294 513	24	496.00

Kdy to dává a kdy naopak nedává smysl?

2. Pouze sloupce

[18]:
```
mesta.loc[:, "kraj"]
```

[18]:
```
mesto
brno        JHM
liberec     LBK
litvinov    ULK
most        ULK
olomouc     OLK
ostrava     MSK
plzen       PLK
praha       PHA
Name: kraj, dtype: object
```

[19]:
```
mesta.loc[:, ["kraj", "linky"]]
```

[19]:

mesto	kraj	linky

```
brno      JHM     22
liberec   LBK      6
litvinov  ULK      5
most      ULK      5
olomouc   OLK      7
ostrava   MSK     15
plzen     PLK      3
praha     PHA     24
```

Zkrácený zápis.

[20]:
```
mesta["kraj"]
```

[20]:
```
mesto
brno        JHM
liberec     LBK
litvinov    ULK
most        ULK
olomouc     OLK
ostrava     MSK
plzen       PLK
praha       PHA
Name: kraj, dtype: object
```

[21]:
```
mesta[["kraj", "linky"]]
```

[21]:
```
          kraj  linky
mesto
brno       JHM     22
liberec    LBK      6
litvinov   ULK      5
most       ULK      5
olomouc    OLK      7
ostrava    MSK     15
plzen      PLK      3
praha      PHA     24
```

3. Řádky a sloupce

[22]:
```
mesta.loc["plzen", "linky"]
```

[22]:
```
3
```

[23]:
```
mesta.loc["most":"plzen", "obyvatel"]
```

[23]:
```
mesto
most        66 644
olomouc    100 494
```

```
ostrava    290 450
plzen      170 936
Name: obyvatel, dtype: object
```

[24]:
```
mesta.loc[["most", "brno", "praha"], ["obyvatel", "vymera"]]
```

[24]:
```
        obyvatel  vymera
mesto
most      66 644   86.94
brno     379 527  230.22
praha  1 294 513  496.00
```

1.2.2 Výběr podle pozic řádků a sloupců

Pro připomenutí.

[25]:
```
mesta
```

[25]:
```
         kraj   obyvatel  linky  vymera
mesto
brno      JHM    379 527     22  230.22
liberec   LBK    103 979      6  106.09
litvinov  ULK     24 143      5   40.70
most      ULK     66 644      5   86.94
olomouc   OLK    100 494      7  103.36
ostrava   MSK    290 450     15  214.23
plzen     PLK    170 936      3  137.65
praha     PHA  1 294 513     24  496.00
```

[26]:
```
mesta.iloc[2]
```

[26]:
```
kraj           ULK
obyvatel    24 143
linky            5
vymera        40.7
Name: litvinov, dtype: object
```

[27]:
```
mesta.iloc[[2]]
```

[27]:
```
         kraj obyvatel  linky  vymera
mesto
litvinov  ULK   24 143      5    40.7
```

[28]:
```
mesta.iloc[2, 1:]
```

[28]:
```
obyvatel    24 143
linky            5
```

```
vymera        40.7
Name: litvinov, dtype: object
```

[29]:
```
mesta.iloc[[2, 3, 5], [0, 1]]
```

[29]:
```
         kraj obyvatel
mesto
litvinov  ULK   24 143
most      ULK   66 644
ostrava   MSK  290 450
```

1.3 3. Ukládání dat

[30]:
```
mesta.to_csv("data.csv")
```

[31]:
```
mesta.to_html("data.html")
```

[32]:
```
mesta.to_excel("data.xls")
```

[33]:
```
mesta.to_json("data.json", indent=4)
```

1.4 A Cvičení

[ ]:
```

```

1.5 4. Index

Pokud index explicitně nevytvoříme jako v příkladu předtím, Pandas ho vytvoří automaticky číselný. Je to podobné jako číslování řádků v Excelu, každá tabulka má index. Jde jen o to, že si ho můžeme pojmenovat sami, máme-li k tomu rozumný důvod.

Připomenutí: takhle vypadal náš DataFrame doteď.

[34]:
```
mesta
```

[34]:
```
         kraj   obyvatel  linky  vymera
mesto
brno      JHM    379 527     22  230.22
liberec   LBK    103 979      6  106.09
litvinov  ULK     24 143      5   40.70
most      ULK     66 644      5   86.94
olomouc   OLK    100 494      7  103.36
ostrava   MSK    290 450     15  214.23
plzen     PLK    170 936      3  137.65
praha     PHA  1 294 513     24  496.00
```

Nespecifikujeme explicitní index.

[35]:
```
mesta = pandas.read_csv("mesta.csv", encoding="utf-8")
mesta
```

[35]:

	mesto	kraj	obyvatel	linky	vymera
0	brno	JHM	379 527	22	230.22
1	liberec	LBK	103 979	6	106.09
2	litvinov	ULK	24 143	5	40.70
3	most	ULK	66 644	5	86.94
4	olomouc	OLK	100 494	7	103.36
5	ostrava	MSK	290 450	15	214.23
6	plzen	PLK	170 936	3	137.65
7	praha	PHA	1 294 513	24	496.00

Index je teď číselný. Přístup pomocí názvu řádků (`loc`) nebo pomocí jejich čísel (`iloc`) je teď velmi podobný. Jediný rozdíl je v indexování rozsahem.

[36]:
```
mesta.loc[:4]
```

[36]:

	mesto	kraj	obyvatel	linky	vymera
0	brno	JHM	379 527	22	230.22
1	liberec	LBK	103 979	6	106.09
2	litvinov	ULK	24 143	5	40.70
3	most	ULK	66 644	5	86.94
4	olomouc	OLK	100 494	7	103.36

[37]:
```
mesta.iloc[:4]
```

[37]:

	mesto	kraj	obyvatel	linky	vymera
0	brno	JHM	379 527	22	230.22
1	liberec	LBK	103 979	6	106.09
2	litvinov	ULK	24 143	5	40.70
3	most	ULK	66 644	5	86.94

Index můžeme manuálně nastavit na jeden ze sloupců. Ten potom zmizí z datové části, a přesune se do pozice indexu.

[38]:
```
mesta.set_index("mesto")
```

[38]:

mesto	kraj	obyvatel	linky	vymera
brno	JHM	379 527	22	230.22
liberec	LBK	103 979	6	106.09
litvinov	ULK	24 143	5	40.70
most	ULK	66 644	5	86.94
olomouc	OLK	100 494	7	103.36
ostrava	MSK	290 450	15	214.23
plzen	PLK	170 936	3	137.65
praha	PHA	1 294 513	24	496.00

1.6 5. Dotazy jako v SQL

Srovnáním DataFrame s tabulkou podobnost s databázemi nekončí. Pandas umožňují dotazovat se nad daty podobným způsobem jako SQL.

Vrátíme názvy měst jako index pro lepší srozumitelnost.

[39]:
```
mesta = pandas.read_csv("mesta.csv", index_col="mesto", encoding="utf-8")
mesta
```

[39]:

mesto	kraj	obyvatel	linky	vymera
brno	JHM	379 527	22	230.22
liberec	LBK	103 979	6	106.09
litvinov	ULK	24 143	5	40.70
most	ULK	66 644	5	86.94
olomouc	OLK	100 494	7	103.36
ostrava	MSK	290 450	15	214.23
plzen	PLK	170 936	3	137.65
praha	PHA	1 294 513	24	496.00

1.6.1 Výběr sloupečků

SQL: `SELECT linky, obyvatel FROM mesta;`

[40]:
```
mesta[["linky", "obyvatel"]]
```

[40]:

mesto	linky	obyvatel
brno	22	379 527
liberec	6	103 979
litvinov	5	24 143
most	5	66 644
olomouc	7	100 494
ostrava	15	290 450
plzen	3	170 936
praha	24	1 294 513

1.6.2 Podmínky

SQL: `SELECT * FROM mesta WHERE linky > 10;`

[41]:
```
mesta[mesta["linky"] > 10]
```

[41]:

mesto	kraj	obyvatel	linky	vymera
brno	JHM	379 527	22	230.22
ostrava	MSK	290 450	15	214.23
praha	PHA	1 294 513	24	496.00

SQL: `SELECT kraj, vymera FROM mesta WHERE vymera >= 100 AND vymera <= 200;`

[42]:
```
mesta[(mesta["vymera"] >= 100) & (mesta["vymera"] <= 200)][["kraj", "vymera"]]
```

[42]:
```
        kraj  vymera
mesto
liberec  LBK  106.09
olomouc  OLK  103.36
plzen    PLK  137.65
```

[43]:
```
mesta.loc[(mesta['vymera'] >= 100) & (mesta['vymera'] <= 200), ["kraj",␣
↪"vymera"]]
```

[43]:
```
        kraj  vymera
mesto
liberec  LBK  106.09
olomouc  OLK  103.36
plzen    PLK  137.65
```

1.6.3 Logické operátory v podmínkách

SQL: `SELECT linky FROM mesta WHERE kraj = 'JHM' OR kraj = 'OLK';`

[44]:
```
mesta[(mesta['kraj'] == 'JHM') | (mesta['kraj'] == 'OLK')][['linky']]
```

[44]:
```
         linky
mesto
brno        22
olomouc      7
```

SQL: `SELECT linky FROM mesta WHERE kraj IN ('JHM', 'ULK', 'OLK');`

[45]:
```
mesta[mesta['kraj'].isin(['JHM', 'ULK', 'OLK'])][['linky']]
```

[45]:
```
          linky
mesto
brno         22
litvinov      5
most          5
olomouc       7
```

SQL: `SELECT linky FROM mesta WHERE kraj NOT IN ('JHM', 'ULK', 'OLK');`

[46]:
```
mesta[~mesta['kraj'].isin(['JHM', 'ULK', 'OLK'])][['linky']]
```

[46]:
```
         linky
mesto
liberec      6
```

```
ostrava     15
plzen        3
praha       24
```

1.7 6. Převod mezi DataFrame a seznamy

1.7.1 DataFrame -> seznam

[47]:
```
mesta.values.tolist()
```

[47]:
```
[['JHM', '379 527', 22, 230.22],
 ['LBK', '103 979', 6, 106.09],
 ['ULK', '24 143', 5, 40.7],
 ['ULK', '66 644', 5, 86.94],
 ['OLK', '100 494', 7, 103.36],
 ['MSK', '290 450', 15, 214.23],
 ['PLK', '170 936', 3, 137.65],
 ['PHA', '1 294 513', 24, 496.0]]
```

V datech chybí názvy měst. Pandas totiž nebere index jako součást dat, ale jen jako popis tabulky. Je tedy potřeba ho do tabulky nejdřív dostat.

[48]:
```
mesta.reset_index()
```

[48]:

	mesto	kraj	obyvatel	linky	vymera
0	brno	JHM	379 527	22	230.22
1	liberec	LBK	103 979	6	106.09
2	litvinov	ULK	24 143	5	40.70
3	most	ULK	66 644	5	86.94
4	olomouc	OLK	100 494	7	103.36
5	ostrava	MSK	290 450	15	214.23
6	plzen	PLK	170 936	3	137.65
7	praha	PHA	1 294 513	24	496.00

[49]:
```
mesta_seznam = mesta.reset_index().values.tolist()
mesta_seznam
```

[49]:
```
[['brno', 'JHM', '379 527', 22, 230.22],
 ['liberec', 'LBK', '103 979', 6, 106.09],
 ['litvinov', 'ULK', '24 143', 5, 40.7],
 ['most', 'ULK', '66 644', 5, 86.94],
 ['olomouc', 'OLK', '100 494', 7, 103.36],
 ['ostrava', 'MSK', '290 450', 15, 214.23],
 ['plzen', 'PLK', '170 936', 3, 137.65],
 ['praha', 'PHA', '1 294 513', 24, 496.0]]
```

Pozor, `list(mesta)` nedělá to co bychom čekali.

[50]:
```
list(mesta)
```

[50]:
```
['kraj', 'obyvatel', 'linky', 'vymera']
```

1.7.2 Seznam -> DataFrame

[51]:
```
pandas.DataFrame(mesta_seznam)
```

[51]:

	0	1	2	3	4
0	brno	JHM	379 527	22	230.22
1	liberec	LBK	103 979	6	106.09
2	litvinov	ULK	24 143	5	40.70
3	most	ULK	66 644	5	86.94
4	olomouc	OLK	100 494	7	103.36
5	ostrava	MSK	290 450	15	214.23
6	plzen	PLK	170 936	3	137.65
7	praha	PHA	1 294 513	24	496.00

To funguje, akorát Pandas neví jak pojmenovat sloupce – v seznamu žádná taková informace není. Pokud se nám nelíbí čísla která automatický dosadí, dodáme názvy sloupců sami.

[52]:
```
pandas.DataFrame(mesta_seznam, columns=["mesto", "kraj", "obyvatel", "linky",␣
↪"vymera"])
```

[52]:

	mesto	kraj	obyvatel	linky	vymera
0	brno	JHM	379 527	22	230.22
1	liberec	LBK	103 979	6	106.09
2	litvinov	ULK	24 143	5	40.70
3	most	ULK	66 644	5	86.94
4	olomouc	OLK	100 494	7	103.36
5	ostrava	MSK	290 450	15	214.23
6	plzen	PLK	170 936	3	137.65
7	praha	PHA	1 294 513	24	496.00

1.8 B Cvičení

[ ]:
```

```