Введение в Pandas

Берленко Татьяна Андреевна, МОЭВМ, 2020

Начало работы

- Установка

```
pip install pandas # Важна версия pip! Для python3.7 pip3.7
```

- Импорт

```
import pandas as pd
```

- Структуры данных
  - Series
  - DataFrame

Pandas

- Series
  - создание из списка Python

```
s = pd.Series([-5, 10, 89, -3])
s
0   -5
1   10
2   89
3   -3
dtype: int64  # тип данных значений серии
```

Первый столбец чисел - индексные метки, второй - значения.

По умолчанию индексы - числа.

Элементами могут быть не только числа, но и строки, списки, словари, и тд

Pandas

- Series
  - создание из словаря Python

```
>> pd.Series({'a':8, 'f':5})
a    8
f    5
dtype: int64
```

  - создание с помощью numpy

```
>> ar = np.arange(4, 9, 0.5)
>> ar
array([4. , 4.5, 5. , 5.5, 6. , 6.5, 7. , 7.5, 8. , 8.5])
>> pd.Series(ar)
0    4.0
1    4.5
2    5.0
3    5.5
4    6.0
5    6.5
6    7.0
7    7.5
8    8.0
9    8.5
dtype: float64
```

Pandas

- Series
  - создание из скаляра

    >> pd.Series(4)
    0 4
    dtype: int64
  - свойства .index и .values

    >> ar.values
    array([4. , 4.5, 5. , 5.5, 6. , 6.5, 7. , 7.5, 8. , 8.5]) # Возвращает массив NumPy даже если базово это был не он
    >> ar.index
    RangeIndex(start=0, stop=10, step=1) # Диапазон значений
  - размер и форма

    len(ar) # количество элементов серии
    ar.size # количество элементов серии
    ar.shape # возвращает кортеж из двух значений, для series это одно значение - количество элементов
  - установка индекса во время создания объекта

    >> labels = ['apple', 'orange', 'avocado']
    >> s = pd.Series([1, 2, 3], labels)
    >> s.index
    Index(['apple', 'orange', 'avocado'], dtype='object') # можно использовать индекс для доступа к элементам

Pandas

- Series
  - .head(), .tail()

```
>> s = pd.Series(list('qwertyuiasdfgjkl'))
>> s.head(2)
0  q
1  w
dtype: object
>> s.tail()
s.tail()
11  f
12  g
13  j
14  k
15  l
dtype: object
```

  - .take()

```
>> s.take((1, 5, 10)) # только порядковый номер элемента серии, иначе ошибка, даже если есть индекс
1  w
5  y
10  d
dtype: object
```

Pandas

- Series
  - Поиск по метке

```
# оператор []
>> s = pd.Series(list('qwerty'), index=list('abcdef'))
s
a   q
b   w
c   e
d   r
e   t
f   y
dtype: object
>> s[0]
'q'
>> s['a']
'q'
>> s[['a', 'c']] # также и для целочисленных позиций
a   q
c   e
dtype: object
```

  - Поиск по метке

```
>> s = pd.Series(list('qwerty'), index=[1, 4, 5, 10, 0, 6])
>> s
1   q
4   w
5   e
10  r
0   t
6   y
dtype: object
>> s[0] (поиск по метке)
't'
>> s[2] # вызовет ошибку
>> s = pd.Series(list('qwerty'), index=[1, 4, 5, 10, 10, 3])
>> s[10]
10  r
10  t
dtype: object
```

Pandas

- Series
  - .iloc(), .loc()

    # **явный** поиск по позиции и по метке
    >> s.iloc[[0,3]] # позиция
    1 q
    10 r
    dtype: object
    >> s.loc[10] # метка
    10 r
    10 t
    dtype: object
  - Срезы

    >> s.iloc[-1]
    39
    >> s.iloc[-1:-4:-1]
    10.000000 39
    9.578947 38
    9.157895 37
    dtype: int64
    >> s[2:5]
    ...
  - Срезы с отрицательными значениями

    >> s[-4:] # вернет последние 4 значения (аналог .tail(4))
  - Срезы с маркированными индексами

    >> a = pd.Series(['qwe', 'rty', 'asd', 'fgh'], index=['apple', 'banana', 'orange', 'avocado'])
    >> a['banana':]
    banana rty
    orange asd
    avocado fgh
    dtype: object
    # или
    >> a.loc['banana':]
    banana rty
    orange asd
    avocado fgh
    dtype: object

Pandas

- Series
  - Выравнивание данных по меткам индекса

```
>> a = pd.Series(['qwe', 'rty', 'asd', 'fgh'], index=list('abcd'))
>> b = pd.Series(['ert', 'opp', 'tum', 'pok'], index=list('abcd'))
>> a + b
a    qweert
b    rtyopp
c    asdtum
d    fghpok
dtype: object
# порядок индексов не важен
>> a * 5
a    qweqweqweqweqwe
b    rtyrtyrtyrtyrty
c    asdasdasdasdasd
d    fghfghfghfghfgh
dtype: object
# аналогично
>> t = pd.Series(5, index=a.index)
>> a * t
```

```
>> c = pd.Series(['ert', 'opp', 'tum', 'pok'], index=list('dpcj'))
>> a + c
a      NaN # такого индекса нет у c
b      NaN
c    asdtum
d    fghert
j      NaN # такого индекса нет у a
p      NaN
dtype: object
```

Pandas

- Series
  - Выполнение логического отбора

```
>> a = pd.Series(np.arange(-3, 3, 0.5))
>> cond = a > 0
>> a[cond] # можно сразу написать a[a> 0]]
7    0.5
8    1.0
9    1.5
10   2.0
11   2.5
dtype: float64
# при этом логические операторы не работают, нужно использовать & | со скобками
```

  - .all(), .any(), .sum()

```
.all() - все ли значения соответствуют выражению
>> cond.all() # или (a< 0).all()
False
.any() # хоть одно значение соответствует?
>> cond.any()
True
>> cond..sum() # количество элементов, соответствующих условию
```

Pandas

- Series
  - **Переиндексация**

```
>> s.index = [9, 8, 7, 5] # модифицирует объект на месте
>> s = pd.Series([1, 9, 0, -7])
>> s
0   1
1   9
2   0
3  -7
dtype: int64
>> s.reindex([0, 8, 2]) # возвращает новый объект
0   1.0
8   NaN
2   0.0
dtype: float64
>> b.reindex([1, 2, 88], fill_value=0)
# можно указать значение вместо NaN
```

```
>> a = pd.Series([1, 5, -8], index=[0,1,2])
>> b = pd.Series([61, 50, -80], index=list('123'))
>> a + b
0   NaN
1   NaN
2   NaN
1   NaN
2   NaN
3   NaN
dtype: float64
>> b.index = b.index.values.astype(int) # cast к int
a + b
0    NaN
1   66.0
2   42.0
3    NaN
dtype: float64
```

Полезные материалы

- Документация https://pandas.pydata.org/docs/
- Онлайн-учебник https://coderlessons.com/tutorials/python-technologies/vyuchit-python-panda/uchebnik-po-python-pandas
- Онлайн-курс https://stepik.org/course/4852/syllabus
- Онлайн-курс https://www.coursera.org/learn/data-analysis-with-python#syllabus