Module 02

Series

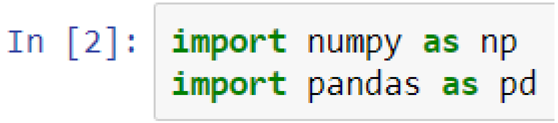
Data Science Developer

Purwadhika
Startup and Coding School

Definition

- A Series is very similar to a NumPy array (in fact it is built on top of the NumPy array object).
- What differentiates the NumPy array from a Series, is that a Series can have axis labels, meaning it can be indexed by a label, instead of just a number location.
- It also doesn't need to hold numeric data, it can hold any arbitrary Python Object.

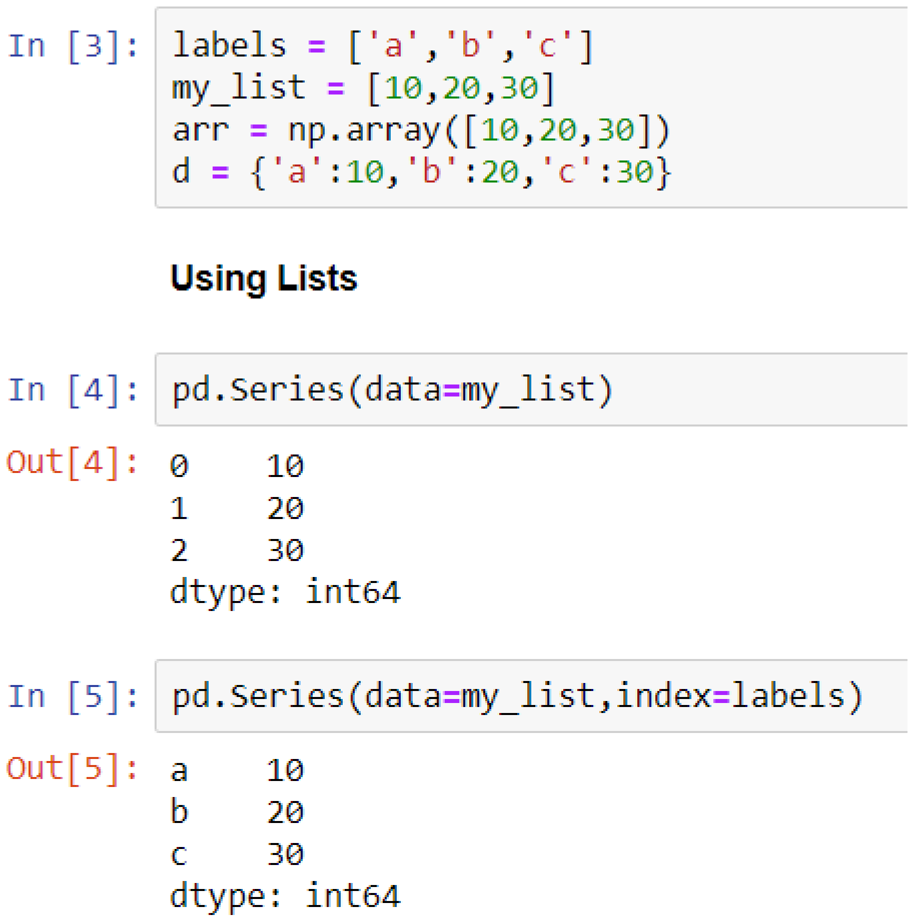
Using Numpy and Pandas



```
In [2]: import numpy as np
        import pandas as pd
```

Creating a Series

From a Python List

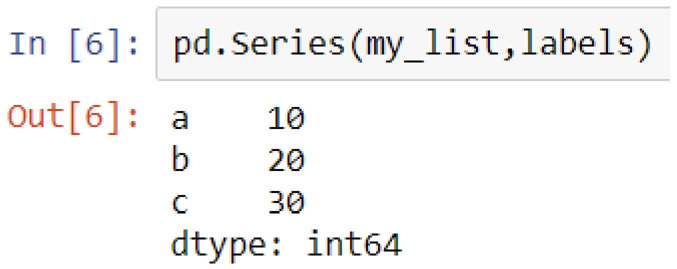


```
In [3]: labels = ['a','b','c']
        my_list = [10,20,30]
        arr = np.array([10,20,30])
        d = {'a':10,'b':20,'c':30}
```

Using Lists

```
In [4]: pd.Series(data=my_list)

Out[4]: 0    10
        1    20
        2    30
        dtype: int64
```

```
In [5]: pd.Series(data=my_list,index=labels)

Out[5]: a    10
        b    20
        c    30
        dtype: int64
```

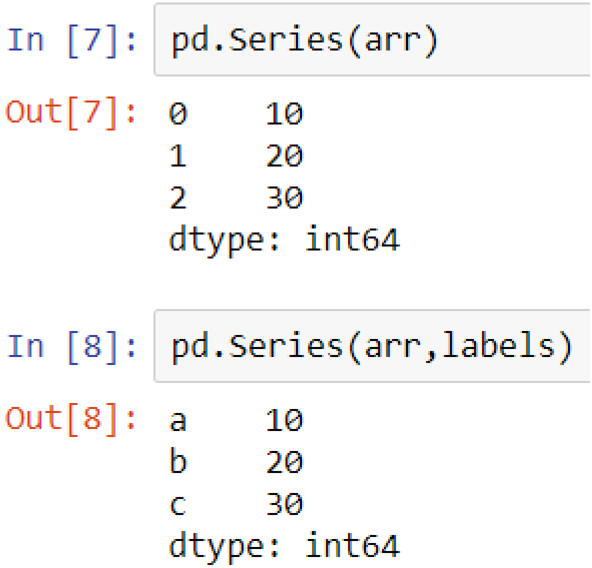


```
In [6]: pd.Series(my_list,labels)

Out[6]: a    10
        b    20
        c    30
        dtype: int64
```

Purwadhika
Startup and Coding School

Creating a Series

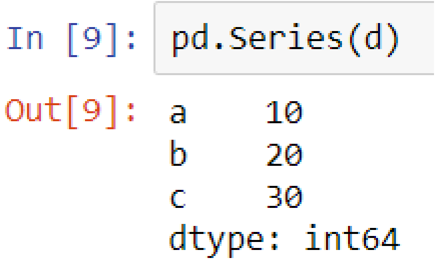
From a Numpy Array



```
In [7]:  pd.Series(arr)

Out[7]:  0    10
         1    20
         2    30
         dtype: int64

In [8]:  pd.Series(arr,labels)

Out[8]:  a    10
         b    20
         c    30
         dtype: int64
```

Creating a Series

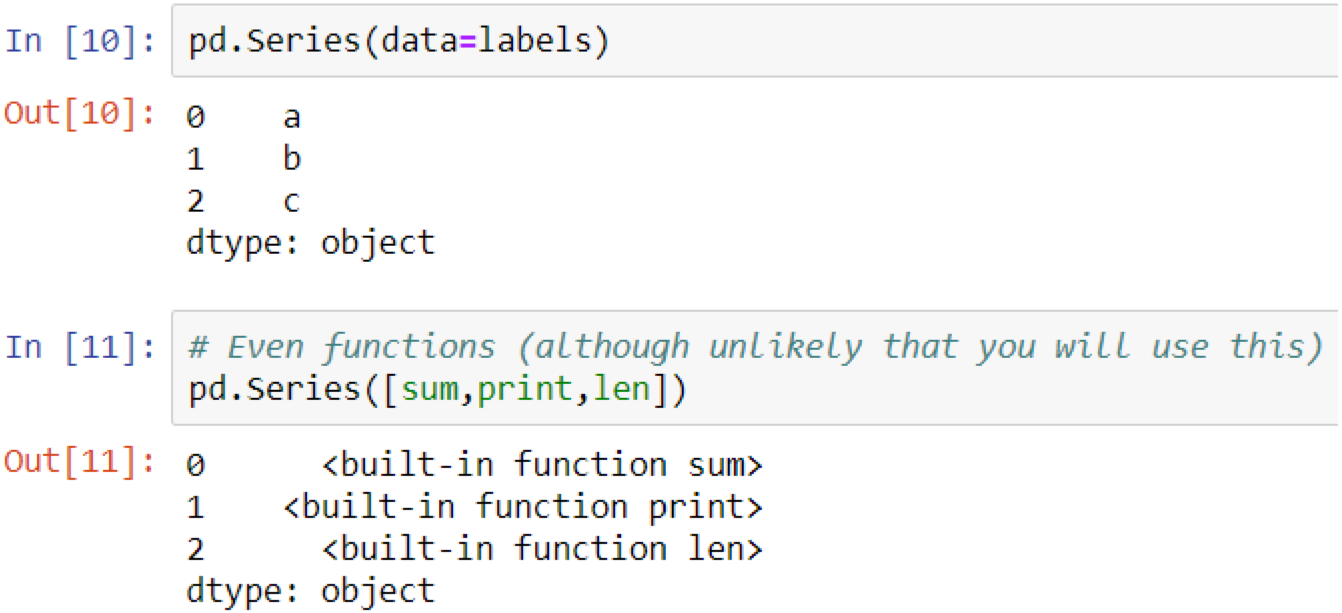
From a Dictionary



```
In [9]: pd.Series(d)

Out[9]: a     10
        b     20
        c     30
        dtype: int64
```

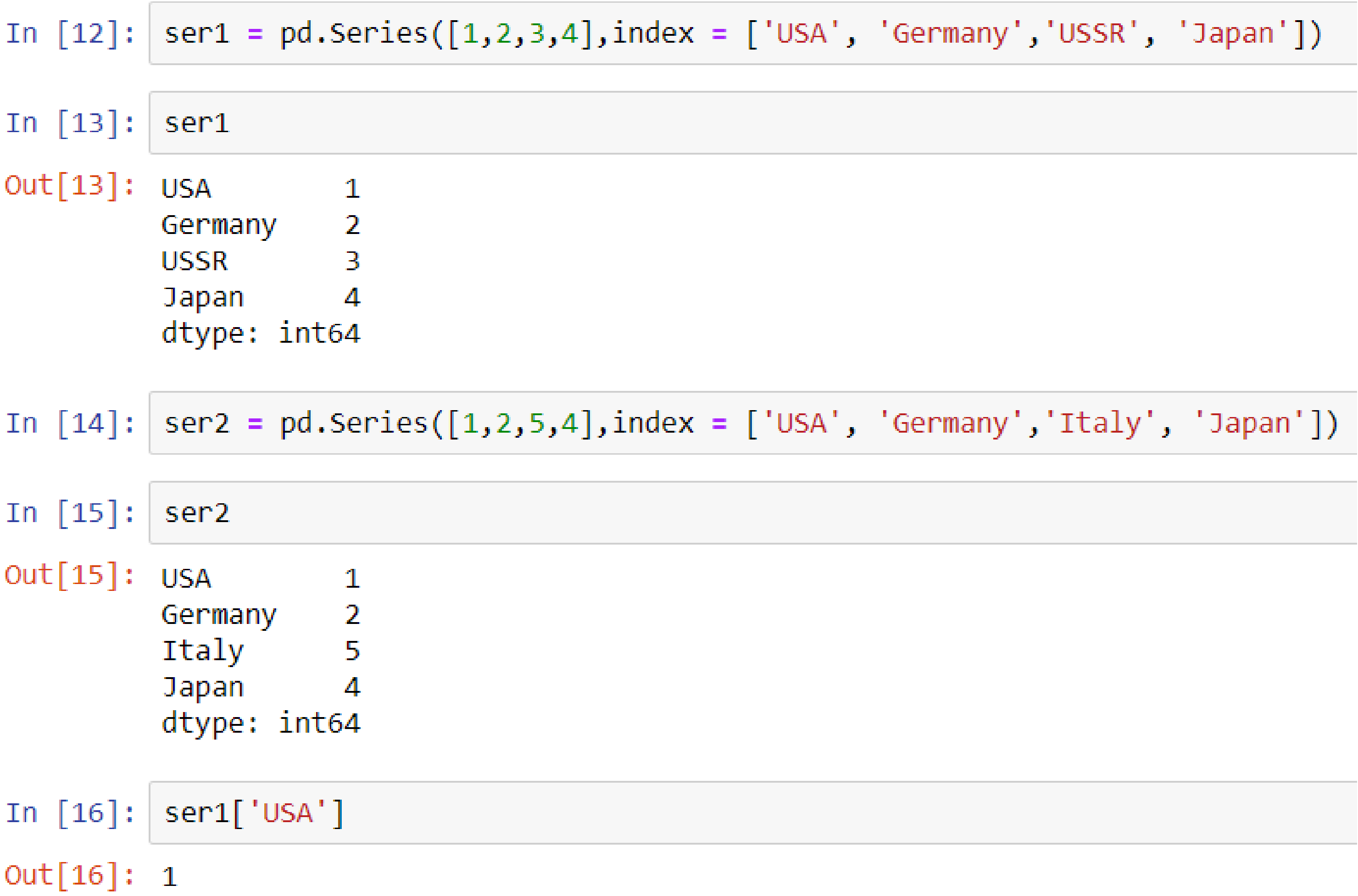
Data in a Series

A pandas Series can hold a variety of object types:



```
In [10]: pd.Series(data=labels)

Out[10]: 0    a
         1    b
         2    c
         dtype: object

In [11]: # Even functions (although unlikely that you will use this)
         pd.Series([sum,print,len])

Out[11]: 0      <built-in function sum>
         1    <built-in function print>
         2      <built-in function len>
         dtype: object
```

Using an Index

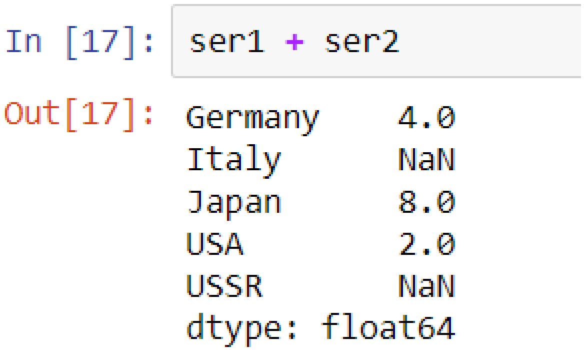


```
In [12]: ser1 = pd.Series([1,2,3,4],index = ['USA', 'Germany','USSR', 'Japan'])

In [13]: ser1

Out[13]: USA        1
         Germany    2
         USSR       3
         Japan      4
         dtype: int64

In [14]: ser2 = pd.Series([1,2,5,4],index = ['USA', 'Germany','Italy', 'Japan'])

In [15]: ser2

Out[15]: USA        1
         Germany    2
         Italy      5
         Japan      4
         dtype: int64

In [16]: ser1['USA']

Out[16]: 1
```

Purwadhika
Startup and Coding School

Using an Index

Operations are then also done based off of index:



```
In [17]: ser1 + ser2

Out[17]: Germany    4.0
         Italy      NaN
         Japan      8.0
         USA        2.0
         USSR       NaN
         dtype: float64
```