# 02_Series_II

November 28, 2023

## 0.1 Pandas: Series (II)

Recordando de la píldora anterior, una `Serie` de Pandas es un array unidimensional de datos indexados. Se puede crear a partir de una lista o un array de la siguiente manera:

[7]:
```
import numpy as np
import pandas as pd


top5 = pd.Series([2923,2799,2320,2264,2071],
                 index = ["Avatar", "Vengadores:End Game", "Avatar II",␣
 ↪"Titanic", "SW VII"])

top5.name = "Top5"

top5
```

[7]:
```
Avatar                 2923
Vengadores:End Game    2799
Avatar II              2320
Titanic                2264
SW VII                 2071
Name: Top5, dtype: int64
```

### 0.1.1 `Series` como array NumPy

Por lo que hemos visto hasta ahora, puede parecer que el objeto `Series` es básicamente intercambiable con un array unidimensional de NumPy. **La diferencia esencial es la presencia del**

**índice**: mientras que el array de Numpy tiene un índice entero *implícitamente definido* que se utiliza para acceder a los valores, las `Series` de Pandas tienen un índice ***explícitamente definido*** asociado a los valores.*OJO.*- si lo quieres hacer usando un index , o ya verermos creando a partir de un diccionario, lo que pasa es que si no lo pones recurrrira al index prefefinido al implicitamente defincido , el posicional

Esta definición explícita del índice proporciona al objeto `Series` capacidades adicionales. Por ejemplo, el índice no necesita ser un entero, sino que puede consistir en valores de cualquier tipo deseado(int, boole, etc). Por ejemplo, **si lo deseamos, podemos utilizar cadenas como índice:**, aunque lo normal es usar siempre un tipo de indicde (string, int, float etc). Se paraecen a los dicionarios

[9]:
```
# mindice implicito posicional de siempre)iloc
top5.iloc[2]
```

[9]: 2320

[13]:
```
top5.loc["Avatar II"]
```

[13]: 2320

[ ]:
```

```

Y el acceso al artículo funciona como se esperaba:

[14]:
```
top5["Avatar II"]
```

[14]: 2320

[17]:
```
serie_test = pd.Series([2300,1200,1200,1222])

print(type(serie_test))
```

```
<class 'pandas.core.series.Series'>
```

[18]:
```
serie_test.index
```

[18]: RangeIndex(start=0, stop=4, step=1)

Pero lo más interesante, como array subyacente que es, se pueden aplicar todos los métodos de los array de numpy

[19]:
```
print(top5.max())
print(top5.min())
print(top5.sum())
print(top5.mean())
```

```
2923
2071
12377
2475.4
```

[20]:
```
print(top5.argmax())
print(top5.argmin())
```

```
0
4
```

metodos propios de las Series

[21]:
```
print(top5.idxmax())
```

```
Avatar
```

[22]:
```
print(top5.idxmin())
```

```
SW VII
```

[24]:
```
print(top5.to_list())# con esto puedo crear una lista con los valores
```

```
[2923, 2799, 2320, 2264, 2071]
```

Para terminar, recordemos que, de nuevo como array que son, el slicing crea vistas y que si queremos copias que no interfieran con la Series original es neceario hacer uso del método copy

[26]:
```
top_vista = top5[3:]
top_vista_copy = top5[3:].copy
print(top_vista)
print(top_vista_copy)
```

```
Titanic    2264
SW VII     2071
Name: Top5, dtype: int64
<bound method NDFrame.copy of Titanic    2264
SW VII     2071
Name: Top5, dtype: int64>
```

[28]:
```
top5
```

[28]:
```
Avatar                 2923
Vengadores:End Game    2799
Avatar II              2320
Titanic                2264
SW VII                 2071
Name: Top5, dtype: int64
```

[31]:
```
top_vista_copy["Titanic"] = 1000
print(top_vista_copy)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
```

```
e:\Cursos\BC_Data_Science\Repositorio\ONLINE_DS_THEBRIDGE_V\SPRING 4\UNIT␣
 ↪1\02_Series_II.ipynb Celda 23 line 1

----> <a href='vscode-notebook-cell:/e%3A/Cursos/BC_Data_Science/Repositorio/
 ↪ONLINE_DS_THEBRIDGE_V/SPRING%204/UNIT%201/02_Series_II.ipynb#X32sZmlsZQ%3D%3D'
 ↪line=0'>1</a> top_vista_copy["Titanic"] = 1000

      <a href='vscode-notebook-cell:/e%3A/Cursos/BC_Data_Science/Repositorio/
 ↪ONLINE_DS_THEBRIDGE_V/SPRING%204/UNIT%201/02_Series_II.ipynb#X32sZmlsZQ%3D%3D'
 ↪line=1'>2</a> print(top_vista_copy)

TypeError: 'method' object does not support item assignment
```

[32]:
```
top5
```

[32]:
```
Avatar                  2923
Vengadores:End Game     2799
Avatar II               2320
Titanic                 2264
SW VII                  2071
Name: Top5, dtype: int64
```

[33]:
```
top_vista["Titanic"] =1000
```

[35]:
```
top5
```

[35]:
```
Avatar                  2923
Vengadores:End Game     2799
Avatar II               2320
Titanic                 1000
SW VII                  2071
Name: Top5, dtype: int64
```

[ ]:
```

```