04_DataFrame_I

November 28, 2023

0.1 Pandas: DataFrame (I)

0.2 Introducción

La siguiente estructura fundamental en Pandas es el `DataFrame`.(NOTA: los dataframe para un datasciencist) Al igual que el objeto `Series` de la sección anterior, el `DataFrame` puede ser **pensado como una generalización de un array de NumPy, o como una especialización de un diccionario de Python.** Ahora echaremos un vistazo a cada una de estas perspectivas.

0.2.1 DataFrame como matriz generalizada de NumPy

Si una `Serie` es un análogo de un array unidimensional con índices flexibles, un **`DataFrame` es un análogo de un array bidimensional con índices de fila y nombres de columna flexibles.**(UNA TABLA) Al igual que se puede pensar en una matriz bidimensional como una secuencia ordenada de columnas unidimensionales alineadas, se puede pensar en un `DataFrame` como una secuencia de objetos `Series` alineados.(lo que nos viene a decir es que otra forma de ver un dataframe es verlo como una tabla formada por series verticales o columnas que son series de pandas y todas las columnas tienen el mismo indice) Aquí, por "alineado" queremos decir que **comparten el mismo índice.**()

Para comprobarlo, reconstruyamos la serie de la población de estados de la sesión anterior y luego una nueva `Serie` que enumere el área de cada uno de los cinco estados discutidos en la sesion anterior:

[3]:
```
import pandas as pd

population_dict = {'California': 38332521,
                   'Texas': 26448193,
```

```
                   'New York': 19651127,
                   'Florida': 19552860,
                   'Illinois': 12882135}

population = pd.Series(population_dict)
```

[6]:
```
area_dict = {"California":423967,
             "Texas": 695662,
             "New York":141297,
             "Florida": 170312,
             "Illinois": 149995}
area = pd.Series(area_dict)
area
```

[6]:
```
California    423967
Texas         695662
New York      141297
Florida       170312
Illinois      149995
dtype: int64
```

Ahora que tenemos esto junto con la serie `población` de antes, podemos utilizar un diccionario para construir un único objeto bidimensional que contenga esta información:(creamos un diccionario de 2 series panda)

[8]:
```
estados = {"poblacion": population,
           "superficie": area}
estados
```

[8]:
```
{'poblacion': California    38332521
 Texas         26448193
 New York      19651127
 Florida       19552860
 Illinois      12882135
 dtype: int64,
 'superficie': California    423967
 Texas         695662
 New York      141297
 Florida       170312
 Illinois      149995
 dtype: int64}
```

Ahora creamos un dataframe a partir de ese diccionario y veamos que pinta tiene:

[11]:
```
states = pd.DataFrame(estados)
states
```

[11]:
```
            poblacion  superficie
California   38332521      423967
Texas        26448193      695662
New York     19651127      141297
Florida      19552860      170312
Illinois     12882135      149995
```

Las series anteriores son ahora las columnas del dataframe y tienen como nombre el nombre de la clave en el diccionario. Fijate que además las series comparten índice. Veamos esto un poco más.

Al igual que el objeto `Series`, el `DataFrame` tiene un atributo `index` que da acceso a las etiquetas del índice:

[12]:
```
states.index # nombre de la fila(indice de la serie que hemos usado)
```

[12]:
```
Index(['California', 'Texas', 'New York', 'Florida', 'Illinois'],
dtype='object')
```

Además, el `DataFrame` tiene un atributo `columns`, que es un objeto `Index` que contiene las etiquetas de las columnas:

[13]:
```
states.columns # nos dice el nombre de las columnas y tb es un objeto de tipo␣
↪index que veremos en un par de sesiones
```

[13]:
```
Index(['poblacion', 'superficie'], dtype='object')
```

Y un atributor `values` que nos da los valores, pero fijate de que tipo es

[14]:
```
states.values # es una array formado por una matriz bidimesional
```

[14]:
```
array([[38332521,   423967],
       [26448193,   695662],
       [19651127,   141297],
       [19552860,   170312],
       [12882135,   149995]], dtype=int64)
```

De este modo, el `DataFrame` puede considerarse como una generalización de una matriz NumPy bidimensional, en la que tanto las filas como las columnas tienen un índice generalizado para acceder a los datos. Pero en la siguiente sesion veremos que es mejor verla como un megadiccionario o incliso como una tabla formado por filas y columnas que parten de un indice