## Abrir archivos con pandas

nombre edad ciudad  
0 Juan 28 Madrid  
1 Ana 22 Barcelona

para saltar tabulaciones agrega el parametro delimiter al pd.read asi:  
 pd.read\_csv(source, delimiter='\t')

Para archivos con comentarios con el signo #,   
 se debe saltar las filas con el parametro skiprows = n

nombre edad ciudad  
0 Juan 28 Madrid  
1 Ana 22 Barcelona  
2 Luis 35 Valencia

Dependiendo el caso agregar el parametro: engine='python'

## Descripcion preliminar de los datos

Para tener una descripcion de los datos utlizar el metodo describe de pandas

edad  
count 14.000000  
mean 25.000000  
std 3.113247  
min 22.000000  
25% 22.000000  
50% 25.000000  
75% 28.000000  
max 28.000000

Si quiero saber los tipos utilizo la propiedad dtypes:

nombre object  
edad int64  
ciudad object

Si quero obtener los valores unicos de una columna utilizo el metodo unique, e  
 specificando la columna, ejemplo: df["nombre"].unique():

['Juan' 'Ana' 'Pablo' 'Carlos' 'Maria' 'Carla' 'Marcos' 'Jhon' 'Camila'  
 'Enzo' 'Julian' 'Tita' 'Pepe']

Si quiero contarlos utilizo el metodo value\_counts:

nombre  
Juan 2  
Ana 1  
Pablo 1  
Carlos 1  
Maria 1  
Carla 1  
Marcos 1  
Jhon 1  
Camila 1  
Enzo 1  
Julian 1  
Tita 1  
Pepe 1

## Manejo de datos faltantes

Para contar la cantidad de datos faltantes por columna   
 utilizar los metodos .isnull().sum():

ID 0  
Nombre 1  
Edad 2  
Ciudad 1  
Salario 2

Para eliminar los datos faltantes se utiliza el metodo .dropna()

ID Nombre Edad Ciudad Salario  
0 1 Juan 30.0 Madrid 3000.0  
1 2 Ana 22.0 Barcelona 3200.0  
2 3 Luis 35.0 Valencia 3500.0  
3 4 María 29.0 Sevilla 4000.0  
6 7 Pedro 20.0 Valencia 3700.0

Si quiero eliminar las columnas con datos faltantes se utiliza el parametro axis=1  
 en el metodo .dropna()

ID  
0 1  
1 2  
2 3  
3 4  
4 5  
5 6  
6 7  
7 8  
8 9  
9 10

Se puede rellenar los datos faltantes utilizando el metodo fillna()  
 y especificar que deseas rellenar

ID Nombre Edad Ciudad Salario  
0 1 Juan 30.0 Madrid 3000.0  
1 2 Ana 22.0 Barcelona 3200.0  
2 3 Luis 35.0 Valencia 3500.0  
3 4 María 29.0 Sevilla 4000.0  
4 5 Carlos NaN Madrid 3200.0  
5 6 Laura 30.0 Madrid NaN  
6 7 Pedro 20.0 Valencia 3700.0  
7 8 Desconocido 22.0 Valencia 3400.0  
8 9 Lucía NaN NaN 3600.0  
9 10 Hugo 25.0 Sevilla NaN

Para rellenar datos numericos como el salario se puede rellenar con la media

ID Nombre Edad Ciudad Salario  
0 1 Juan 30.0 Madrid 3000.0  
1 2 Ana 22.0 Barcelona 3200.0  
2 3 Luis 35.0 Valencia 3500.0  
3 4 María 29.0 Sevilla 4000.0  
4 5 Carlos NaN Madrid 3200.0  
5 6 Laura 30.0 Madrid 3450.0  
6 7 Pedro 20.0 Valencia 3700.0  
7 8 Desconocido 22.0 Valencia 3400.0  
8 9 Lucía NaN NaN 3600.0  
9 10 Hugo 25.0 Sevilla 3450.0

Para datos numericos como la edad, es mas util la mediana:

ID Nombre Edad Ciudad Salario  
0 1 Juan 30.0 Madrid 3000.0  
1 2 Ana 22.0 Barcelona 3200.0  
2 3 Luis 35.0 Valencia 3500.0  
3 4 María 29.0 Sevilla 4000.0  
4 5 Carlos 27.0 Madrid 3200.0  
5 6 Laura 30.0 Madrid 3450.0  
6 7 Pedro 20.0 Valencia 3700.0  
7 8 Desconocido 22.0 Valencia 3400.0  
8 9 Lucía 27.0 NaN 3600.0  
9 10 Hugo 25.0 Sevilla 3450.0

Para rellenar datos como la cuidad, seria util el uso de la moda

ID Nombre Edad Ciudad Salario  
0 1 Juan 30.0 Madrid 3000.0  
1 2 Ana 22.0 Barcelona 3200.0  
2 3 Luis 35.0 Valencia 3500.0  
3 4 María 29.0 Sevilla 4000.0  
4 5 Carlos 27.0 Madrid 3200.0  
5 6 Laura 30.0 Madrid 3450.0  
6 7 Pedro 20.0 Valencia 3700.0  
7 8 Desconocido 22.0 Valencia 3400.0  
8 9 Lucía 27.0 Madrid 3600.0  
9 10 Hugo 25.0 Sevilla 3450.0

Para especificar los datos en donde se deban eliminar las columnas  
 se utiliza la propiedad subset, ejemplo de subset= [Nombre]:

ID Nombre Edad Ciudad Salario  
0 1 Juan 30.0 Madrid 3000.0  
1 2 Ana 22.0 Barcelona 3200.0  
2 3 Luis 35.0 Valencia 3500.0  
3 4 María 29.0 Sevilla 4000.0  
4 5 Carlos NaN Madrid 3200.0  
5 6 Laura 30.0 Madrid NaN  
6 7 Pedro 20.0 Valencia 3700.0  
8 9 Lucía NaN NaN 3600.0  
9 10 Hugo 25.0 Sevilla NaN

## Reestructurar datos

Para segmentar y clasificar datos continuos en grupos o intevalos se utiliza pd.cut():

ID Nombre Edad Ciudad Salario Rango\_Edad  
0 1 Juan 30.0 Madrid 3000.0 25-30  
1 2 Ana 22.0 Barcelona 3200.0 20-25  
2 3 Luis 35.0 Valencia 3500.0 30-35  
3 4 María 29.0 Sevilla 4000.0 25-30  
4 5 Carlos 27.0 Madrid 3200.0 25-30  
5 6 Laura 30.0 Madrid 3450.0 25-30  
6 7 Pedro 20.0 Valencia 3700.0 20-25  
7 8 Desconocido 22.0 Valencia 3400.0 20-25  
8 9 Lucía 27.0 Madrid 3600.0 25-30  
9 10 Hugo 25.0 Sevilla 3450.0 20-25

Para obtener el salario promedio por edad utilizo groupby()

Rango\_Edad  
20-25 3437.5  
25-30 3450.0  
30-35 3500.0

## Manejo de duplicados

Al usar el metodo .duplicated() me indica por medio de booleans que filas son duplicadas, sin necesidad de que estos sean consecutivos

0 False  
1 False  
2 False  
3 False  
4 False  
5 True  
6 False  
7 False  
8 False  
9 False  
10 True

Se puede usar el parametro subset para especificar las columnas y saber si contiene duplicados

0 False  
1 False  
2 False  
3 False  
4 False  
5 True  
6 False  
7 False  
8 False  
9 True  
10 True

Obteniendo las filas de los duplicados con: duplicate[duplicate.duplicated()

ID Nombre Edad Salario Fecha\_Ingreso  
5 3 Carlos 30 58000 2020-03-25  
10 2 Luis 35 60000 2019-06-15

Para eliminar los duplicados se utiliza el metodo .drop\_duplicates()

ID Nombre Edad Salario Fecha\_Ingreso  
0 1 Ana 23 50000 2020-01-10  
1 2 Luis 35 60000 2019-06-15  
2 3 Carlos 30 58000 2020-03-25  
3 4 María 22 58000 2018-07-30  
4 5 Elena 28 49000 2018-07-30  
6 7 Sofía 34 72000 2016-11-22  
7 8 Andrés 45 61000 2021-04-01  
8 9 Laura 29 52000 2015-05-19  
9 10 Ana 41 67000 2014-08-25