TELEDUC
Educación a Distancia

Herramientas avanzadas de programación en *Python* para procesamiento de datos

Resumen

Módulo 4

Importante: Para todos los códigos que se entregarán en este resumen, se asume que ya se ejecutó la siguiente línea de código:

```
import pandas as pd
```

Además, que existe un Data Frame creado de nombre df

En este módulo se explicarán las herramientas para extraer información de un Data Frame y darle más valor a los datos.

Recordemos que se emplea la función *loc* para hacer filtros básicos (bajo un criterio), de la siguiente forma:

```
df.loc[df['Nombre columna'] (operación lógica)]
```

Esto permite solamente hacer un tipo de filtro, para una columna. Es posible filtrar por más de un criterio mediante los operadores lógicos binarios (*and* y *or*). En *Python*, estos se representan por medio de las mismas palabras *and* y *or*. Pero en *Pandas*, para crear filtros dentro de la función *loc*, se utiliza la siguiente sintaxis:

- ***And:*** se utiliza el carácter "&". Provoca que las operaciones lógicas a la izquierda y derecha se deban cumplir simultáneamente, para que el resultado total sea *True*.
- ***Or:*** se ocupa el carácter "|". Provoca que se deba cumplir al menos una de las operaciones lógicas a la izquierda y derecha, para que el resultado total sea *True*.

Con los operadores lógicos binarios mencionado anteriormente, es posible crear filtros más complejos (que tengan más de un criterio). De forma general, para un Data Frame de nombre df:

```
df.loc[(df['Nombre columna'] (operación lógica)) (&/|)
  (df['Nombre columna'] (operación lógica)) (&/|) … ]
```

Donde es posible incluir la cantidad de filtros (criterios) que se estime conveniente.

Asimismo, también es posible ordenar valores en base a una o más columnas mediante la función sort_values. De forma general:

```
df = df.sort_values(by=(columna o lista de columnas),
             ascending = (True o False))
```

Es importante notar que si se quiere ordenar un Data Frame por más de una columna (en el parámetro *by*), cuando hay valores repetidos en la primera, se ocupa la segunda como criterio, y así sucesivamente. En cuando al parámetro *ascending*, este indica si se quiere ordenar de menor a mayor (ascending = True), o de mayor a menor (ascending = False).

Además, es posible reordenar los valores de un Data Frame en una nueva tabla mediante la función *pivot*. Con ella, es posible elegir qué valores se quieren como filas y cuáles como columnas. Entonces,, es posible darle una nueva interpretación a los datos dependiendo de cómo usamos esta función. De forma general:

```
df.pivot(index=(columna que determinará las filas),
  columns=(columna que determinara las columnas),
           values=(columna a analizar))
```

De esta manera, podemos especificar qué información queremos en las filas, qué información queremos en las columnas, y cuáles son los valores para analizar bajo esta nueva interpretación de los datos (que estén dentro de esta tabla generada).

Finalmente, tenemos la pivot_table, que es muy similar a las tablas dinámicas de Excel. Es un caso particular de la función *pivot*, y permite aplicar funciones matemáticas sobre la columna que especifiquemos en el parámetro *values*. Esto se hace mediante la librería *numpy*. De forma general:

```
df.pivot_table(index=(columna que determinará las filas)
         ,columns=(columna que determinara las
columnas),values=(columna a analizar),aggfunc=(funciones
                de la librería numpy))
```

Las funciones de la librería *numpy* que se emplean son:

- np.mean: para determinar la media en la columna del parámetro *values* bajo la interpretación que se configure
- np.min: para determinar el valor mínimo en la columna del parámetro *values* bajo la interpretación que se configure
- np.max: para determinar el valor máximo en la columna del parámetro *values* bajo la interpretación que se configure
- np.std: para determinar la desviación estándar en la columna del parámetro *values* bajo la interpretación que se configure

Se observa que en el parámetro *aggfunc*, se puede especificar más de una función de la librería *numpy* mediante los caracteres “{}” y separando por “,”. Por ejemplo:

```
{np.mean,np.min,np.max}
```