Autor: María José
Gómez Silva

CAPÍTULO 2. INGESTA DE DATOS.

2. 4 Relaciones entre tablas

En ocasiones, los datos necesarios para alimentar un algoritmo de Machine Learning están distribuidos en varios *DataFrames.* En esos casos es necesario combinar los *DataFrames* en uno nuevo que contenga toda la información necesaria.

Pandas ofrece la función *pd.merge* que permite combinar las filas de dos *DataFrames* de acuerdo a la comparación del valor de una o varias columnas. Esta operación es similar al operador *join* del lenguaje SQL empleado sobre bases de datos relacionales. A la función *pd.merge* hay que indicarle los dos *DataFrames* que tiene que relacionar y el resultado será un nuevo *DataFrame*. Para definir la relación a aplicar se dispone de dos argumentos, *on* y *how*. Las columnas de combinación se indican mediante una lista en el argumento *on*. Y el tipo de combinación se indica en el argumento *how*. Los tipos de combinaciones posibles son *left, right, outer* e *inner*.

Vamos a estudiar el funcionamiento de las combinaciones mencionadas, empleando dos tablas, una con datos sobre las calificaciones obtenidas por algunos alumnos en algunas materias, y otra con el listado de los profesores que imparten algunas de las materias. En este caso, la columna en común y que relaciona ambas tablas es '*materia*', como puede observarse en la Figura 1.

```
In [70]: tabla1 = pd.DataFrame({'Alumno':['Pedro', 'Pedro', 'Irene', 'Samuel'],
                                'Materia':['Matemáticas', 'Literatura', 'Matemáticas', 'Tecnología'],
                                'Nota':[8.2, 3.4, 5.8, 6.7]})
         tabla1
```

Out[70]:

	Alumno	Materia	Nota
0	Pedro	Matemáticas	8.2
1	Pedro	Literatura	3.4
2	Irene	Matemáticas	5.8
3	Samuel	Tecnología	6.7

```
In [72]: tabla2 = pd.DataFrame({'Materia':['Matemáticas', 'Literatura', 'Biología'],
                                'Profesor':['Rodríguez', 'Díaz', 'García']})
         tabla2
```

Out[72]:

	Materia	Profesor
0	Matemáticas	Rodríguez
1	Literatura	Díaz
2	Biología	García

Figura 1. DataFrames relacionados por la columna 'materia'.

Emplearemos la columna '*Materia*' como columna de comparación.

La operación de combinación de tipo *inner* devuelve únicamente aquellas filas que tienen valores idénticos en la columna que se compara. Si no se indica nada, la función *pd.merge* realiza por defecto una combinación de tipo *inner*.

La combinación de tipo *left* devuelve un *DataFrame* con las filas que tienen valores idénticos en la columna comparada y todas las filas del *DataFrame* de la izquierda (con o sin correspondencia con el *DataFrame* de la derecha). Las celdas sin correspondencia se rellenan con NaN.

De forma análoga, la combinación de tipo *right* devuelve un *DataFrame* con las filas que tienen valores idénticos en la columna comparada y todas las filas del *DataFrame* de la derecha (con o sin correspondencia con el *DataFrame* de la izquierda).

Por último, la combinación de tipo *outer*, devuelve la unión de las filas devueltas por las opciones *left* y *right*. La figura 2 muestra el resultado de los cuatro tipos de combinaciones empleando las tablas de la Figura 1.

```
In [74]: resultado=pd.merge(tabla1, tabla2, on=['Materia'], how='inner')
         resultado
```

Out[74]:

	Alumno	Materia	Nota	Profesor
0	Pedro	Matemáticas	8.2	Rodríguez
1	Irene	Matemáticas	5.8	Rodríguez
2	Pedro	Literatura	3.4	Díaz

```
In [75]: resultado=pd.merge(tabla1, tabla2, on=['Materia'], how='left')
         resultado
```

Out[75]:

	Alumno	Materia	Nota	Profesor
0	Pedro	Matemáticas	8.2	Rodríguez
1	Pedro	Literatura	3.4	Díaz
2	Irene	Matemáticas	5.8	Rodríguez
3	Samuel	Tecnología	6.7	NaN

```
In [76]: resultado=pd.merge(tabla1, tabla2, on=['Materia'], how='right')
         resultado
```

Out[76]:

	Alumno	Materia	Nota	Profesor
0	Pedro	Matemáticas	8.2	Rodríguez
1	Irene	Matemáticas	5.8	Rodríguez
2	Pedro	Literatura	3.4	Díaz
3	NaN	Biología	NaN	García

```
In [77]: resultado=pd.merge(tabla1, tabla2, on=['Materia'], how='outer')
         resultado
```

Out[77]:

	Alumno	Materia	Nota	Profesor
0	Pedro	Matemáticas	8.2	Rodríguez
1	Irene	Matemáticas	5.8	Rodríguez
2	Pedro	Literatura	3.4	Díaz
3	Samuel	Tecnología	6.7	NaN
4	NaN	Biología	NaN	García

Figura 2. Resultado de aplicar los cuatro tipos de combinaciones de DataFrames disponibles con la función pd.merge.

En el ejemplo anterior, ambos *DataFrames* contenían una columna con el mismo nombre, *Materia*, que ha sido usada para realizar las comparaciones. Si esta columna recibiese nombres distintos en cada *DataFrame*, por ejemplo, *Materia* en la tabla 1 y *Asignatura* en la tabla 2, habría que indicar los nombres de estas dos columnas a comparar en la función *pd.merge*, tal y como se muestra en la Figura 3, con los comandos *left_on* y *right_on*.

```
In [79]: tabla1 = pd.DataFrame({'Alumno':['Pedro', 'Pedro', 'Irene', 'Samuel'],
                                'Materia':['Matemáticas', 'Literatura', 'Matemáticas', 'Tecnología'],
                                'Nota':[8.2, 3.4, 5.8, 6.7]})
         tabla1
```

Out[79]:

	Alumno	Materia	Nota
0	Pedro	Matemáticas	8.2
1	Pedro	Literatura	3.4
2	Irene	Matemáticas	5.8
3	Samuel	Tecnología	6.7

```
In [78]: tabla2 = pd.DataFrame({'Asignatura':['Matemáticas', 'Literatura', 'Biología'],
                                'Profesor':['Rodríguez', 'Díaz', 'García']})
         tabla2
```

Out[78]:

	Asignatura	Profesor
0	Matemáticas	Rodríguez
1	Literatura	Díaz
2	Biología	García

```
In [80]: resultado=pd.merge(tabla1, tabla2, left_on=['Materia'], right_on=['Asignatura'], how='inner')
         resultado
```

Out[80]:

	Alumno	Materia	Nota	Asignatura	Profesor
0	Pedro	Matemáticas	8.2	Matemáticas	Rodríguez
1	Irene	Matemáticas	5.8	Matemáticas	Rodríguez
2	Pedro	Literatura	3.4	Literatura	Díaz

Figura 3. Comparación de columnas con distintos nombres en cada DataFrame.

En algunos casos, los valores a comparar no están en las columnas en común, si no en los índices de los *DataFrames*. En ese caso, habrá que emplear los argumentos *left_index* y *right_index* de la función *pd.merge* (en lugar de *left_on* y *right_on*). Los argumentos *left_index* y *right_index* solo admiten los valores True o False, dependiendo, respectivamente, de si se deben usar o no los índices como valores de comparación. También es posible realizar la comparación entre las columnas de un *DataFrame* y los índices del otro, combinando correctamente los argumentos anteriores.

Cuando se trata de recopilar información distribuida en varios *DataFrames*, además de la función *pd.merge* para combinarlos, Pandas ofrece la función *pd.concat* para concatenarlos. Esta función forma un nuevo *DataFrame* uniendo la información contenida en otros dos, apilándola horizontal (axis=1) o verticalmente (axis=0), o lo que es lo mismo, uniendo sus columnas o sus filas, como muestra la Figura 4.

```
In [83]: tabla1 = pd.DataFrame({'Materia':['Matemáticas', 'Matemáticas', 'Tecnología'],
                                'Nota':[8.2, 3.4, 6.7]},
                                index = ['Pedro', 'Irene', 'Samuel'])
         tabla1
```

Out[83]:

	Materia	Nota
Pedro	Matemáticas	8.2
Irene	Matemáticas	3.4
Samuel	Tecnología	6.7

```
In [84]: tabla2 = pd.DataFrame({'Curso':[2, 1],
                                'Tutor':['Rodríguez', 'Díaz']},
                                index = ['Pedro', 'Irene'] )
         tabla2
```

Out[84]:

	Curso	Tutor
Pedro	2	Rodríguez
Irene	1	Díaz

```
In [86]: resultado = pd.concat([tabla1, tabla2], axis=1)
         resultado
```

Out[86]:

	Materia	Nota	Curso	Tutor
Pedro	Matemáticas	8.2	2.0	Rodríguez
Irene	Matemáticas	3.4	1.0	Díaz
Samuel	Tecnología	6.7	NaN	NaN

```
In [87]: resultado = pd.concat([tabla1, tabla2], axis=0)
         resultado
```

Out[87]:

	Materia	Nota	Curso	Tutor
Pedro	Matemáticas	8.2	NaN	NaN
Irene	Matemáticas	3.4	NaN	NaN
Samuel	Tecnología	6.7	NaN	NaN
Pedro	NaN	NaN	2.0	Rodríguez
Irene	NaN	NaN	1.0	Díaz

Figura 4. Ejemplo de concatenación de dos DataFrames en horizontal y en vertical

REFERENCIAS

https://docs.python.org/3/tutorial/index.html

https://www.anaconda.com/products/individual

https://docs.jupyter.org/en/latest/

https://numpy.org/doc/stable/

https://pandas.pydata.org/docs

https://matplotlib.org/stable/index.html

Machine learning con Python y Scikit-learn by Joaquín Amat Rodrigo, available under a Attribution 4.0 International (CC BY 4.0) at https://www.cienciadedatos.net/documentos/py06_machine_learning_python_scikitlearn.html

https://www.drivendata.org/