INSTITUTO POLITÉCNICO NACIONAL

EL CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN DEPARTAMENTO DE DIPLOMADOS Y EXTENSIÓN PROFESIONAL DEL CENTRO DE INVESTIGACIÓN EN CÓMPUTO DEL INSTITUTO POLITÉCNICO NACIONAL

PRÁCTICA 104
UNIR DOS DATAFRAMES DE PRODUCTOS Y VENTAS RELACIONADOS MEDIANTE UN JOIN

PRESENTA:
CARRETO JIMÉNEZ DULCE KAROL

CORREO:
carretojimenezdulcekarol@gmail.com

CURSO DE PYTHON CIENTÍFICO CIC IPN C101

CURSO IMPARTIDO POR EL PROFESOR
ALAN BADILLO SALAS

FECHA DE ENTREGA
CIUDAD DE MÉXICO A 15 DE JULIO DE 2023

Contenido

Introducción

En muchas ocasiones nos podemos encontrar con que los conjuntos de datos no se encuentran agregados en una única tabla. Por ejemplo, los datos personales de los clientes y las transacciones estos han realizado. Pero esto no es necesario, la consolidación también se puede realizar directamente en pandas.

En el análisis de datos y la manipulación de información, a menudo es necesario unir dos conjuntos de datos relacionados para obtener una visión más completa y significativa de los datos. En Python, la biblioteca pandas proporciona herramientas poderosas para realizar esta tarea, y una de las operaciones comunes es unir o combinar dos DataFrames a través de un "join".

Supongamos que contamos con dos DataFrames: uno contiene información sobre los productos, como su código, nombre y categoría, y el otro DataFrame contiene datos de ventas relacionados con esos productos, incluyendo el código del producto, fecha de venta y cantidad vendida. El objetivo es combinar estos dos conjuntos de datos para obtener una visión integrada que muestre la información de ventas junto con los detalles del producto.

Para lograr esto, utilizaremos la operación de "join" en pandas. El "join" se basa en columnas comunes entre los DataFrames para combinar la información correspondiente. En este caso, la columna clave sería el código del producto, ya que es la columna que relaciona los datos de productos y ventas.

El resultado de la operación de join será un DataFrame que incluye todas las filas de ambos DataFrames que tienen un código de producto en común. Las columnas se conservan en el DataFrame resultante, y si hay columnas con el mismo nombre en ambos DataFrames, se les agregará un sufijo automáticamente para distinguirlas.

Unir DataFrames a través de un join en pandas es una forma eficiente y conveniente de combinar y relacionar conjuntos de datos relacionados, permitiendo un análisis más completo y enriquecido de la información.

Justificación

La ventaja de unir dos DataFrames mediante un join en Python utilizando la biblioteca pandas se basa en varias razones importantes:

- **Completitud de datos:** Al unir dos DataFrames relacionados, se puede obtener una visión más completa y significativa de los datos al combinar información relevante de diferentes fuentes. En el ejemplo mencionado, la unión de los DataFrames de productos y ventas permite tener información sobre los productos y al mismo tiempo conocer las ventas asociadas a cada producto. Esto proporciona una visión integral y facilita el análisis y la toma de decisiones basadas en los datos.
- **Análisis y relaciones entre los datos:** Unir los DataFrames relacionados a través de un join permite establecer relaciones entre diferentes conjuntos de datos. Por ejemplo, se pueden realizar análisis sobre cómo las ventas de ciertos productos están relacionadas con su categoría o cómo las ventas varían a lo largo del tiempo. Al combinar los datos, se pueden realizar consultas y análisis más complejos para obtener información valiosa.
- **Eficiencia y simplicidad del proceso:** La biblioteca pandas en Python proporciona una funcionalidad robusta y eficiente para realizar operaciones de join entre DataFrames. Utilizar esta herramienta simplifica el proceso de unión y ahorra tiempo y esfuerzo en comparación con otros métodos más manuales o complejos.
- **Consistencia y coherencia de los datos:** Al unir los DataFrames relacionados, se asegura que los datos estén alineados correctamente y se evita la duplicación de información. El join garantiza que solo se muestren las filas que tienen una relación válida entre los DataFrames, evitando posibles errores o inconsistencias en el análisis.

Finalmente, podemos decir que unir dos DataFrames mediante un join en pandas se basa en la necesidad de obtener una visión completa y enriquecida de los datos, establecer relaciones entre diferentes conjuntos de datos, simplificar el proceso de combinación, garantizar la coherencia de los datos y permitir un análisis más profundo y eficiente.

Contenido

Planteamiento del problema

El problema nos dice: Construye dos DataFrames de productos, ventas, donde el primer DataFrame tendrá las columnas necesarias para guardar cada producto y el segundo para guardar cada venta como se muestra en las siguientes tablas:

DataFrame de Productos

PRODUCTO_ID	NOMBRE	PRECIO
1	Coca Cola	17.5
2	Galletas Marías	18.9
3	Gansito	21.3
4	Pepsi	16.0
5	Chetos	11.5

DataFrame de Ventas

VENTA_ID	PRODUCTO_ID	FECHA
101	1	2023-07-05 11:55:00
101	3	2023-07-05 11:55:00
101	1	2023-07-05 11:55:00
102	2	2023-07-05 11:57:00
102	2	2023-07-05 11:57:00
102	3	2023-07-05 11:57:00
103	5	2023-07-05 12:31:00
103	4	2023-07-05 12:31:00

Ejecución

Creación del conjunto de datos

1. Crea un DataFrame que extienda el DataFrame de Ventas con los productos usando el operado JOIN.

En primer lugar, antes de explicar el procedimiento para combinar dataframes se ha de crear el conjunto de datos como lo indica el problema. Esto se puede realizar escribiendo directamente los datos. Por lo que, vamos a crear un dataframe con los datos específicos de los productos y otro dataframe con los datos específicos correspondientes a ventas, utilizando la librería “pandas”, como se muestra a continuación.

```
#Importamos librería pandas
import pandas as pd

#DataFrame de Productos
dataProductos=pd.DataFrame ({
    "PRODUCTO_ID":[1, 2, 3, 4, 5],
    "NOMBRE": ["Coca Cola", "Galletas Maria", "Gansito", "Pepsi", "Chetos"],
    "PRECIO": [17.5, 18.9, 21.3, 16.0, 11.5]
})

#DataFrame de Ventas
dataVentas=pd.DataFrame ({
    "VENTA_ID":[101, 101, 101, 102, 102, 102, 103, 103],
    "PRODUCTO_ID": [1, 3, 1, 2, 2, 3, 5, 4],
    "FECHA": ["2023-07-05 11:55:00", "2023-07-05 11:55:00", "2023-07-05 11:55:00",
              "2023-07-05 11:57:00", "2023-07-05 11:57:00", "2023-07-05 11:57:00",
              "2023-07-05 12:31:00", "2023-07-05 12:31:00"]

})

display (dataProductos)
display (dataVentas)
```

Al ejecutar la compilación de datos, obtenemos:

	PRODUCTO_ID	NOMBRE	PRECIO
0	1	Coca Cola	17.5
1	2	Galletas Marìa	18.9
2	3	Gansito	21.3
3	4	Pepsi	16.0
4	5	Chetos	11.5

	VENTA_ID	PRODUCTO_ID	FECHA
0	101	1	2023-07-05 11:55:00
1	101	3	2023-07-05 11:55:00
2	101	1	2023-07-05 11:55:00
3	102	2	2023-07-05 11:57:00
4	102	2	2023-07-05 11:57:00
5	102	3	2023-07-05 11:57:00
6	103	5	2023-07-05 12:31:00
7	103	4	2023-07-05 12:31:00

Agrupación de datos

2. Agrupa el DataFrame extendido para calcular el monto total por venta (suma la columna precio)

Después de haber completado los datasframes con los datos específicos para productos y ventas, se puede unir esta con la de las transacciones. Para esto se puede utilizar el método merge de pandas. A este método se le ha de indicar las dos variables y la columna utilizará para la unión. La unión de las dos tablas se puede realizar con la siguiente estructura:

“nuevo_dataframe = pd.merge (dataframe_1, dataframe_2, on='columna_comun')”

Si en los dataframes existen columnas con el mismo nombre se puede indicar un sufijo para identificar el origen de cada una. De esta manera se puede identificar el origen de cada una de ellas.

“nuevo_dataframe = pd.merge (dataframe_1, dataframe_2, how=’left’, on='columna_comun')”

En este caso como en los dataframes tenemos la misma columna llamada “PRODUCTO_ID” utilizaremos la segunda estructura, como se muestra a continuación:

```
#Crea un DataFrame que extienda el DataFrame de Ventas con los productos usando el operado JOIN
dataExtendido=pd.merge(dataProductos, dataVentas, how='left', on='PRODUCTO_ID')
display (dataExtendido)
```

Al ejecutar la compilación de datos, podemos observar que obtenemos por defecto la combinación de los registros, que están tanto en la lista de productos y la lista de ventas, omitiendo la columna con el mismo nombre. El tipo de unión utilizada se puede indicar utilizando el parámetro “how”.

	PRODUCTO_ID	NOMBRE	PRECIO	VENTA_ID	FECHA
0	1	Coca Cola	17.5	101	2023-07-05 11:55:00
1	1	Coca Cola	17.5	101	2023-07-05 11:55:00
2	2	Galletas Marìa	18.9	102	2023-07-05 11:57:00
3	2	Galletas Marìa	18.9	102	2023-07-05 11:57:00
4	3	Gansito	21.3	101	2023-07-05 11:55:00
5	3	Gansito	21.3	102	2023-07-05 11:57:00
6	4	Pepsi	16.0	103	2023-07-05 12:31:00
7	5	Chetos	11.5	103	2023-07-05 12:31:00

- En los puntos siguientes, necesitamos sumar alguna columna en cuestión, para esto, utilizaremos la función sum() en el DataFrame resultante. Podemos seleccionar las columnas relevantes y aplicar sum() utilizando la opción axis=1 para realizar la suma a lo largo de las filas, como se muestra para los siguientes apartados:
  - Punto adicional: Agrupa el DataFrame extendido para calcular el monto **total por venta** (suma la columna precio).

```
#Agrupa el DataFrame extendido para calcular el monto total por venta (suma la columna precio)
dataExtendido.groupby(by="VENTA_ID").sum()
```

Al compilar, tenemos:

VENTA_ID	PRODUCTO_ID	PRECIO
101	5	56.3
102	7	59.1
103	9	27.5

Datos Extendidos

3. Agrupa el DataFrame extendido para calcular el monto total por producto (suma la columna precio).

En esta ocasión necesitamos sumar una columna específica de un DataFrame resultante de un merge en pandas, para esto utilizaremos el método sum() de pandas en la columna deseada, como ya mencionamos.

```
#Agrupa el DataFrame extendido para calcular el monto total por producto (suma la columna precio)
dataGrupoObtenido=dataExtendido.groupby(by="PRODUCTO_ID").sum().drop(["VENTA_ID"], axis=1)
display (dataGrupoObtenido)
```

Al compilar, tenemos:

PRODUCTO_ID	PRECIO
1	35.0
2	37.8
3	42.6
4	16.0
5	11.5

Impresión estándar

4. Reporta los valores de los montos obtenidos en la impresión estándar.

- Para este punto, primero obtenemos un número total de productos con el método groupby(), en pandas se utiliza para agrupar datos en un DataFrame según una o varias columnas. Es una función muy útil para realizar operaciones de agregación y análisis estadístico en conjuntos de datos.
- También haremos uso del método count(), en pandas se utiliza para contar el número de elementos no nulos en cada columna de un DataFrame. Proporciona el recuento de valores no nulos en cada columna por defecto.
- Finalmente, el método drop(), en pandas se utiliza para eliminar filas o columnas de un DataFrame. Proporciona la capacidad de eliminar etiquetas o índices específicos de filas o columnas.
  - Obtenemos el número total de productos

```
#Número total de productos
dataGrupoTotal=dataExtendido.groupby(by="PRODUCTO_ID").count().drop(["PRECIO", "VENTA_ID", "FECHA"], axis=1)
dataGrupoTotal.rename(columns={"NOMBRE":"NÚMERO DE PRODUCTOS"}, inplace=True)
display (dataGrupoTotal)
```

  - Al compilar, tenemos:

PRODUCTO_ID	NÚMERO DE PRODUCTOS
1	2
2	2
3	2
4	1
5	1

  - Reportamos los valores de los montos obtenidos en la impresión estándar:

```
#Reporta los valores de los montos obtenidos en la impresión estándar
dataReporte=pd.merge(dataGrupoObtenido, dataProductos,  how='left', on='PRODUCTO_ID')
display (dataReporte)
```

  - Al compilar, tenemos:

	PRODUCTO_ID	PRECIO_x	NOMBRE	PRECIO_y
0	1	35.0	Coca Cola	17.5
1	2	37.8	Galletas Maria	18.9
2	3	42.6	Gansito	21.3
3	4	16.0	Pepsi	16.0
4	5	11.5	Chetos	11.5

- Podemos observar, que no existe un orden en la clasificación de precios, es decir, no nos indica el número de productos para la suma en precio_y, lo cual puede ser confuso para aquel que no sabe de donde provienen los datos, para esto, nos sirvió crear “dataGrupoTotal” y “dataReposrte”, porque simplemente las vamos a llamar.

```
dataFinal=pd.merge (dataGrupoTotal, dataReporte,how='left', on='PRODUCTO_ID')
display (dataFinal)
```

- De esta forma, obtenemos “dataFinal”, mostrándonos un orden con una clasificación adecuada, como se muestra a continuación.

	PRODUCTO_ID	NÚMERO DE PRODUCTOS	PRECIO_x	NOMBRE	PRECIO_y
0	1	2	35.0	Coca Cola	17.5
1	2	2	37.8	Galletas Maria	18.9
2	3	2	42.6	Gansito	21.3
3	4	1	16.0	Pepsi	16.0
4	5	1	11.5	Chetos	11.5

Conclusiones

Unir dos DataFrames mediante un join en Python utilizando la biblioteca pandas es una práctica fundamental en el análisis de datos y la manipulación de información. A través de esta operación, se pueden combinar conjuntos de datos relacionados para obtener una visión completa, establecer relaciones y realizar análisis más profundos.

La justificación de utilizar el join radica en la necesidad de tener datos completos, aprovechar la información de múltiples fuentes, simplificar el proceso de combinación, garantizar la coherencia de los datos y permitir un análisis más eficiente. Al unir los DataFrames, se obtiene una perspectiva más integral y significativa de los datos, lo que facilita la toma de decisiones basadas en la información obtenida.

La biblioteca pandas en Python ofrece herramientas poderosas para realizar la operación de join, lo que agiliza el proceso y brinda una funcionalidad robusta para la manipulación de los datos. Esto se traduce en ahorro de tiempo y esfuerzo en comparación con métodos más manuales o complejos.

En conclusión, unir DataFrames mediante un join en pandas es una práctica esencial para el análisis de datos, ya que permite combinar información relacionada, establecer relaciones, obtener una visión completa de los datos y facilitar el análisis y la toma de decisiones. Es una técnica que proporciona coherencia, eficiencia y mayor comprensión de los datos, lo que resulta en una mejor comprensión y aprovechamiento de la información disponible.