Transformaciones Avanzadas y Enriquecimiento - Día 3

Change Status

pending 40 min

Learning Objectives

1. Entender operaciones avanzadas de transformación
2. Aprender joins y merges entre datasets
3. Comprender agregaciones y cálculos derivados
4. Conocer validaciones de integridad

Activities and Learning

Task 1: Joins y Merges (10 minutos)

¿Cómo combinar datos de múltiples fuentes?

Los joins permiten combinar datos relacionados de diferentes tablas o datasets.

Tipos de joins:

Inner join: Solo filas que existen en ambas tablas
Left join: Todas las filas de la tabla izquierda + matches de la derecha
Right join: Todas las filas de la tabla derecha + matches de la izquierda
Outer join: Todas las filas de ambas tablas

Ejemplo de join:

```
import pandas as pd

# Dataset de clientes
clientes = pd.DataFrame({
    'cliente_id': [1, 2, 3, 4],
    'nombre': ['Ana', 'Juan', 'María', 'Pedro']
})

# Dataset de pedidos
pedidos = pd.DataFrame({
    'cliente_id': [1, 1, 3, 5],
    'producto': ['A', 'B', 'C', 'D'],
    'cantidad': [2, 1, 3, 1]
})

# Join para combinar información
pedidos_clientes = pd.merge(
    pedidos,
    clientes,
    on='cliente_id',
    how='left'
)

print(pedidos_clientes)
```

Task 2: Agregaciones y Cálculos Derivados (10 minutos)

¿Cómo crear métricas calculadas?

Las agregaciones resumen datos y los cálculos derivados crean nuevas métricas basadas en datos existentes.

Operaciones comunes:

Sumar: Totales por categoría
Contar: Número de elementos
Promediar: Valores medios
Agrupar: Análisis por segmentos

Ejemplo de agregación:

```
# Agregaciones por cliente
resumen_cliente = pedidos_clientes.groupby('cliente_id').agg({
    'cantidad': 'sum',
    'producto': 'count'
}).rename(columns={
```

```
    'cantidad': 'total_cantidad',
    'producto': 'numero_pedidos'
})

print(resumen_cliente)
```

Task 3: Validaciones de Integridad (10 minutos)

¿Cómo asegurar consistencia de datos?

Las validaciones de integridad verifican que los datos sean coherentes y cumplan reglas de negocio.

Tipos de validación:

Referencial: Claves foráneas existen
Dominio: Valores dentro de rangos válidos
Consistencia: Datos coherentes entre campos
Negocio: Reglas específicas del dominio

Ejemplo de validación:

```
def validar_integridad(df):
    errores = []

    # Validar que clientes referenciados existen
    clientes_validos = set(clientes['cliente_id'])
    pedidos_invalidos = df[~df['cliente_id'].isin(clientes_validos)]

    if len(pedidos_invalidos) > 0:
        errores.append(f"Pedidos con clientes inexistentes: {len(pedidos_invalidos)}")

    # Validar cantidades positivas
    if (df['cantidad'] <= 0).any():
        errores.append("Cantidades no positivas encontradas")

    return errores

errores = validar_integridad(pedidos_clientes)
print("Errores de integridad:", errores)
```