Transformaciones Básicas con Pandas - Día 2

Change Status

pending 40 min

Learning Objectives

1. Entender operaciones básicas de limpieza de datos
2. Aprender manejo de valores faltantes
3. Comprender normalización de datos
4. Conocer técnicas de validación básica

Activities and Learning

Task 1: Limpieza de Datos (10 minutos)

¿Qué significa limpiar datos?

La limpieza de datos corrige errores, inconsistencias y problemas que impiden el análisis efectivo.

Problemas comunes:

Valores faltantes: Datos no disponibles
Duplicados: Registros repetidos
Formatos inconsistentes: Fechas, números en diferentes formatos
Outliers: Valores extremos que pueden ser errores

Ejemplo de limpieza básica:

```
import pandas as pd

# Crear DataFrame con datos sucios
datos = pd.DataFrame({
    'nombre': ['Ana', 'Juan', None, 'María'],
    'edad': [25, None, 30, 25],
    'ciudad': ['Madrid', 'Barcelona', 'Madrid', 'madrid']
})

# Limpiar datos
datos_limpios = datos.copy()
datos_limpios = datos_limpios.dropna()  # Eliminar filas con NaN
datos_limpios['ciudad'] = datos_limpios['ciudad'].str.lower()  # Normalizar texto
```

Task 2: Manejo de Valores Faltantes (10 minutos)

¿Cómo manejar datos faltantes?

Los valores faltantes (NaN, None, null) son inevitables. Hay varias estrategias para manejarlos.

Estrategias principales:

Eliminar: Quitar filas/columnas con muchos faltantes
Imputar: Rellenar con valores calculados (media, mediana, moda)
Flag: Crear columna indicando si faltaba el dato original
Preservar: Mantener como está si es informativo

Ejemplo de imputación:

```
# Imputar valores faltantes
datos['edad'] = datos['edad'].fillna(datos['edad'].mean())  # Media
datos['ciudad'] = datos['ciudad'].fillna('desconocido')  # Valor fijo
```

Task 3: Normalización y Validación (10 minutos)

¿Por qué normalizar datos?

La normalización asegura que los datos tengan formatos consistentes y valores válidos.

Técnicas de normalización:

Texto: Convertir a minúsculas, quitar espacios extra

Fechas: Formato consistente (YYYY-MM-DD)
Números: Escala común, quitar separadores de miles
Categorías: Valores estandarizados

Ejemplo de validación:

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

    # Validar edades razonables
    if (df['edad'] < 0).any() or (df['edad'] > 120).any():
        errores.append("Edades fuera de rango")

    # Validar emails
    import re
    patron_email = r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'
    if not df['email'].str.match(patron_email).all():
        errores.append("Emails inválidos")

    return errores
```