Limpieza y Transformación de Datos - Día 3

Change Status

pending · 40 min

Learning Objectives

1. Identificar y resolver problemas comunes de calidad de datos en datasets reales
2. Aplicar técnicas de limpieza sistemática: duplicados, formatos, normalización
3. Implementar transformaciones básicas que preparan datos para análisis

Ejercicio: Pipeline completo de limpieza de datos realista

Ejercicio práctico para aplicar los conceptos aprendidos.

Crear dataset con problemas comunes:

```
import pandas as pd
import numpy as np

# Crear datos con problemas típicos
datos = {
    'id': [1, 2, 3, 4, 5, 1, 6],  # Duplicado en id 1
    'nombre': ['Ana García', 'Carlos López', 'María Rodríguez', 'Juan Pérez', 'Ana García', 'ana garcia', 'Luis Martín'],
    'edad': ['25', '30', '28', '35', '25', '25', '40'],  # String en lugar de int
    'email': ['ana@email.com', 'carlos@email.com', 'maria@email.com', 'juan@email.com', 'ana@email.com', 'ana@email.com', 'luis@email.com'],
    'salario': [45000, 55000, 48000, 60000, 45000, 45000, 52000],
    'departamento': ['Ventas', 'IT', 'Marketing', 'IT', 'ventas', 'VENTAS', 'Recursos Humanos']  # Inconsistente capitalización
}

df = pd.DataFrame(datos)
print("Datos originales con problemas:")
print(df)
```

Inspeccionar y diagnosticar problemas:

```
print(f"\nTipos de datos: {df.dtypes}")
print(f"\nDuplicados por id: {df['id'].duplicated().sum()}")
```

```
print(f"Duplicados completos: {df.duplicated().sum()}")
print(f"\nValores únicos en departamento: {df['departamento'].unique()}")
```

Limpiar duplicados:

```
# Eliminar duplicados basados en id y email
df_limpio = df.drop_duplicates(subset=['id', 'email'], keep='first')
print(f"\nDespués de eliminar duplicados: {len(df_limpio)} filas")
```

Corregir tipos de datos y formatos:

```
# Convertir edad a numérico
df_limpio['edad'] = pd.to_numeric(df_limpio['edad'], errors='coerce')

# Normalizar departamento
df_limpio['departamento'] = df_limpio['departamento'].str.title()

# Normalizar nombres
df_limpio['nombre'] = df_limpio['nombre'].str.title()

print("
```

Después de correcciones:") print(df_limpio) print(f"\nTipos corregidos: {df_limpio.dtypes}")

````
5. **Crear columnas calculadas**:
```python
# Calcular salario mensual y anual
df_limpio['salario_mensual'] = df_limpio['salario'] / 12
df_limpio['categoria_edad'] = pd.cut(df_limpio['edad'],
                                     bins=[0, 25, 35, 100],
                                     labels=['Joven', 'Adulto', 'Senior'])

print("
Con columnas calculadas:")
print(df_limpio[['nombre', 'edad', 'categoria_edad', 'salario', 'salario_mensual']])
````

Verificación: Compara el dataset original con el limpio y confirma que todos los problemas identificados han sido resueltos.

Requerimientos:

Python con Pandas instalado (del día 1)
NumPy disponible
Dataset de ejemplo o archivo CSV para practicar
Conocimiento básico de DataFrames (del día 2)
````