

Análisis Exploratorio de Datos - DeepSolation

Este notebook explora el dataset de señales de aisladores sísmicos para entender:

- Balance de clases (niveles de daño)
- Distribución por tipo de aislador
- Características de las señales

In [52]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
from glob import glob
import warnings
warnings.filterwarnings('ignore')

plt.style.use('seaborn-v0_8')
plt.rcParams['figure.figsize'] = [12, 6]

# Rutas
LABELS_CSV = '../../../../../data/nivel_damage.csv'
SIGNALS_DIR = '../../../../../data/Signals_Raw/'

print("Setup completado")
```

Setup completado

1. Carga de Etiquetas

In [53]:

```
# Cargar etiquetas
df_labels = pd.read_csv(LABELS_CSV)
df_labels.columns = ['specimen_id', 'tipo', 'nivel_dano']

print(f"Total especímenes: {len(df_labels)}")
print(f"\nColumnas: {list(df_labels.columns)}")
print(f"\nPrimeras filas:")
df_labels
```

Total especímenes: 71

Columnas: ['specimen_id', 'tipo', 'nivel_dano']

Primeras filas:

Out[53]:

	specimen_id	tipo	nivel_dano
0	A1	B	N1
1	A1-2	B	N1
2	A1-3	B	N1
3	A2	B	N1
4	A2-2	B	N1
...
66	A47	NaN	N1
67	A48	NaN	N1
68	A49	NaN	N1
69	A50	NaN	N2
70	A51	NaN	N1

71 rows × 3 columns

2. Balance de Clases

Analizamos la distribución de aisladores por nivel de daño y tipo

In [60]:

```
# Distribución por nivel de daño
fig, ax = plt.subplots(figsize=(10, 5))

nivel_counts = df_labels['nivel_dano'].value_counts().sort_index()
colors_nivel = {'N1': '#2ecc71', 'N2': '#f39c12', 'N3': '#e74c3c'}

bars = ax.bar(nivel_counts.index, nivel_counts.values,
              color=[colors_nivel[n] for n in nivel_counts.index],
              edgecolor='black', linewidth=1.5)
ax.set_title('Distribución por Nivel de Daño (Mediciones multiples por ai')
ax.set_xlabel('Nivel de Daño')
ax.set_ylabel('Cantidad de Aisladores')

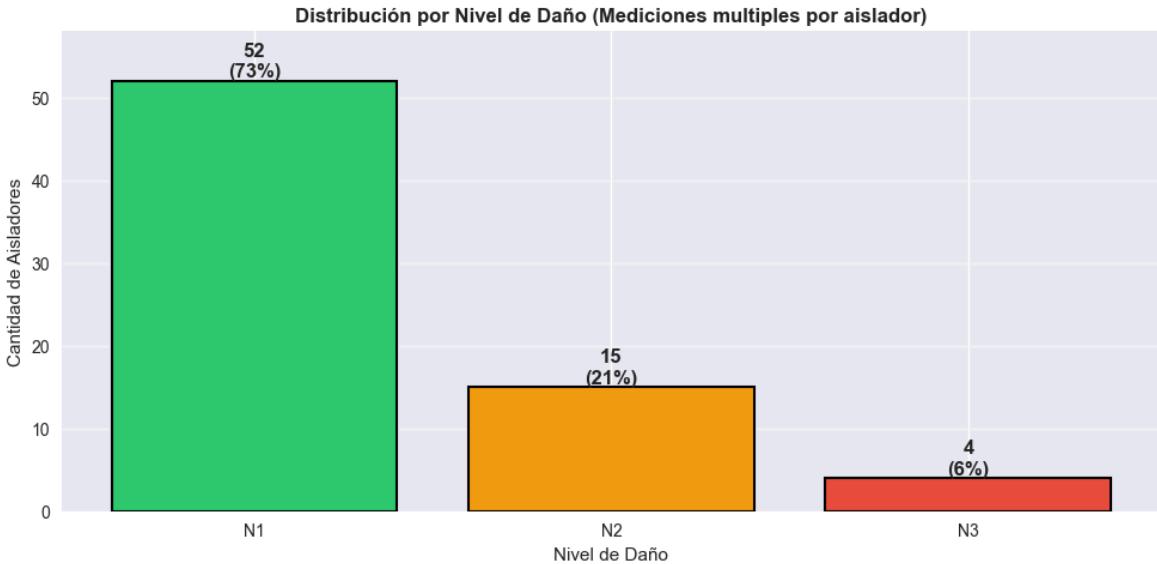
for bar, count in zip(bars, nivel_counts.values):
    ax.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.5,
            f'{count}\n({count/len(df_labels)*100:.0f}%)',
            ha='center', fontweight='bold')

ax.set_ylim(0, max(nivel_counts.values) + 6)
ax.grid(True, alpha=0.3, axis='y')

plt.tight_layout()
plt.show()

# Resumen textual
print("=*50")
print("RESUMEN DE BALANCE")
print("=*50")
print(f"\nPor Nivel de Daño:")
for nivel in ['N1', 'N2', 'N3']:
```

```
count = nivel_counts.get(nivel, 0)
pct = count / len(df_labels) * 100
print(f" {nivel}: {count} aisladores ({pct:.1f}%)")
```



RESUMEN DE BALANCE

Por Nivel de Daño:

N1: 52 aisladores (73.2%)
 N2: 15 aisladores (21.1%)
 N3: 4 aisladores (5.6%)

```
In [55]: # Distribución por nivel de daño - Solo aisladores únicos (sin variantes
df_base = df_labels[~df_labels['specimen_id'].str.contains('-')]

fig, ax = plt.subplots(figsize=(10, 5))

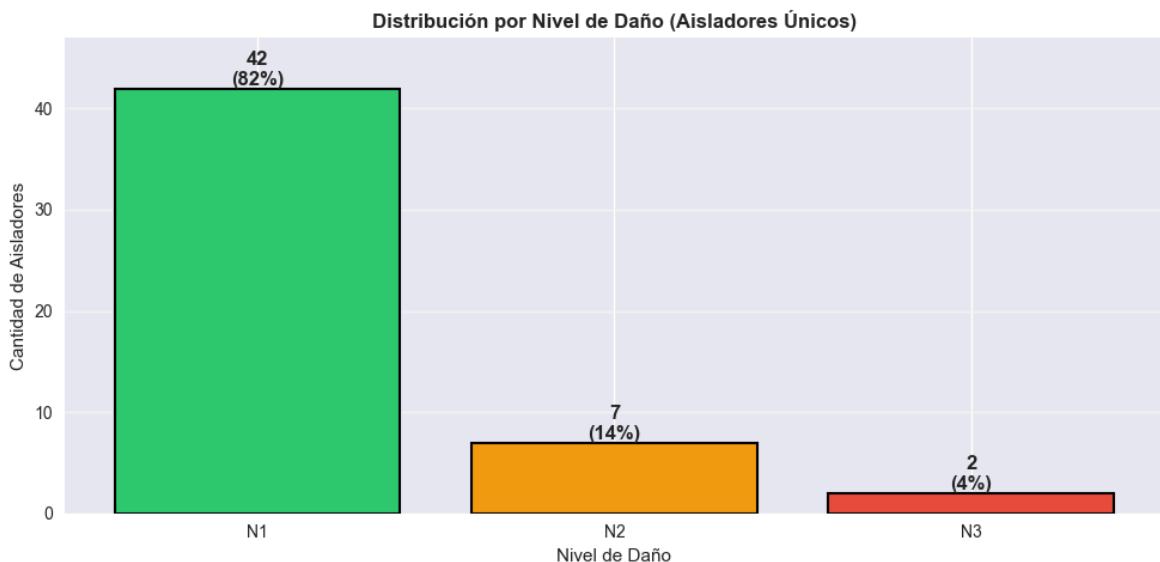
nivel_counts_base = df_base['nivel_dano'].value_counts().sort_index()
bars = ax.bar(nivel_counts_base.index, nivel_counts_base.values,
              color=[colors_nivel[n] for n in nivel_counts_base.index],
              edgecolor='black', linewidth=1.5)
ax.set_title('Distribución por Nivel de Daño (Aisladores Únicos)', fontweight='bold')
ax.set_xlabel('Nivel de Daño')
ax.set_ylabel('Cantidad de Aisladores')

for bar, count in zip(bars, nivel_counts_base.values):
    ax.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.3,
            f'{count}\n({count/len(df_base)*100:.0f}%)',
            ha='center', fontweight='bold')

ax.set_yticks([0, max(nivel_counts_base.values) + 5])
ax.grid(True, alpha=0.3, axis='y')

plt.tight_layout()
plt.show()

print(f'Aisladores únicos: {len(df_base)}')
print(f'\nPor nivel de daño:')
for nivel in ['N1', 'N2', 'N3']:
    count = nivel_counts_base.get(nivel, 0)
    print(f" {nivel}: {count} aisladores ({count/len(df_base)*100:.1f}%)")
```



Aisladores únicos: 51

Por nivel de daño:

N1: 42 aisladores (82.4%)
 N2: 7 aisladores (13.7%)
 N3: 2 aisladores (3.9%)

3. Exploración de Archivos de Señales

Verificamos la estructura de los archivos de señales disponibles.

```
In [56]: # Explorar estructura de directorios
specimen_dirs = sorted([d for d in os.listdir(SIGNALS_DIR)
                       if os.path.isdir(os.path.join(SIGNALS_DIR, d))])

print(f"Total carpetas de aisladores (especímenes): {len(specimen_dirs)}")
print(f"\nAisladores encontrados: {specimen_dirs[:10]}...")

# Verificar archivos por espécimen
signal_info = []
for specimen in specimen_dirs:
    specimen_path = os.path.join(SIGNALS_DIR, specimen)
    files = os.listdir(specimen_path)
    txt_files = [f for f in files if f.endswith('.txt')]

    for f in txt_files:
        sensor = 'S1' if 'S1' in f else 'S2' if 'S2' in f else 'Unknown'
        file_path = os.path.join(specimen_path, f)
        # Leer archivo txt separado por espacios
        df_temp = pd.read_csv(file_path, sep='\s+')
        signal_info.append({
            'specimen': specimen,
            'sensor': sensor,
            'n_samples': len(df_temp),
            'file': f
        })

df_signals = pd.DataFrame(signal_info)
print(f"\nTotal archivos de señales: {len(df_signals)}")
print(f"Sensores por espécimen: {df_signals.groupby('specimen')['sensor']}
```

Total carpetas de aisladores (especímenes): 71

Aisladores encontrados: ['A1', 'A1-2', 'A1-3', 'A10', 'A10-2', 'A10-3', 'A11', 'A11-2', 'A11-3', 'A12']...

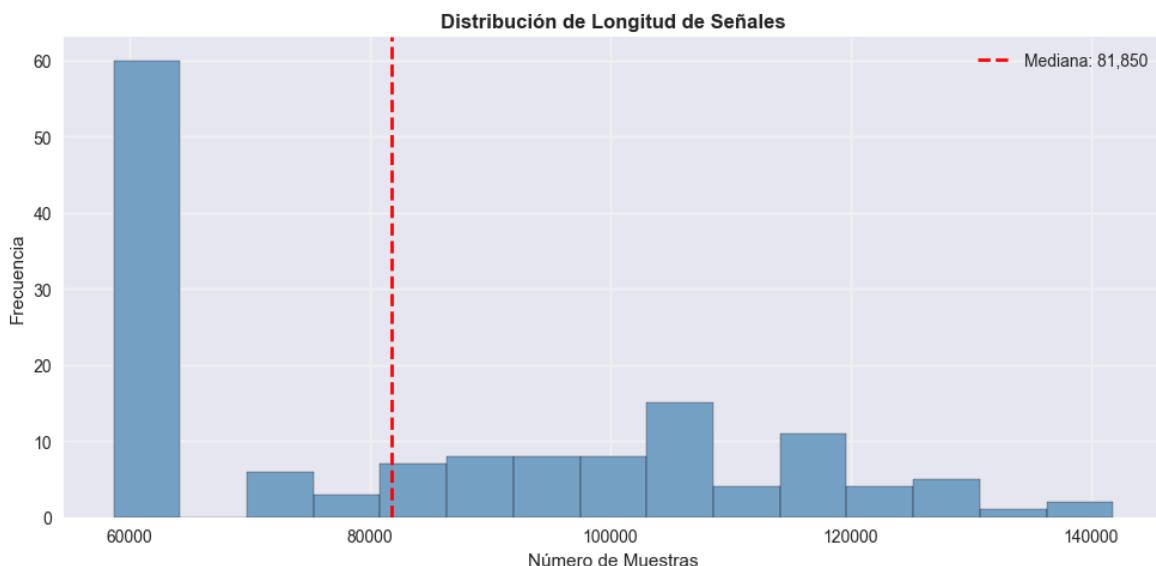
Total archivos de señales: 142
Sensores por espécimen: 2.0

```
In [57]: # Visualizar distribución de longitudes de señales
fig, ax = plt.subplots(figsize=(10, 5))

ax.hist(df_signals['n_samples'], bins=15, color='steelblue',
        edgecolor='black', alpha=0.7)
ax.axvline(df_signals['n_samples'].median(), color='red',
            linestyle='--', linewidth=2, label=f"Mediana: {df_signals['n_s
ax.set_title('Distribución de Longitud de Señales', fontweight='bold')
ax.set_xlabel('Número de Muestras')
ax.set_ylabel('Frecuencia')
ax.legend()
ax.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()

print(f"Estadísticas de longitud de señales:")
print(f"  Mínimo: {df_signals['n_samples'].min():,.0f} muestras")
print(f"  Máximo: {df_signals['n_samples'].max():,.0f} muestras")
print(f"  Mediana: {df_signals['n_samples'].median():,.0f} muestras")
```



Estadísticas de longitud de señales:

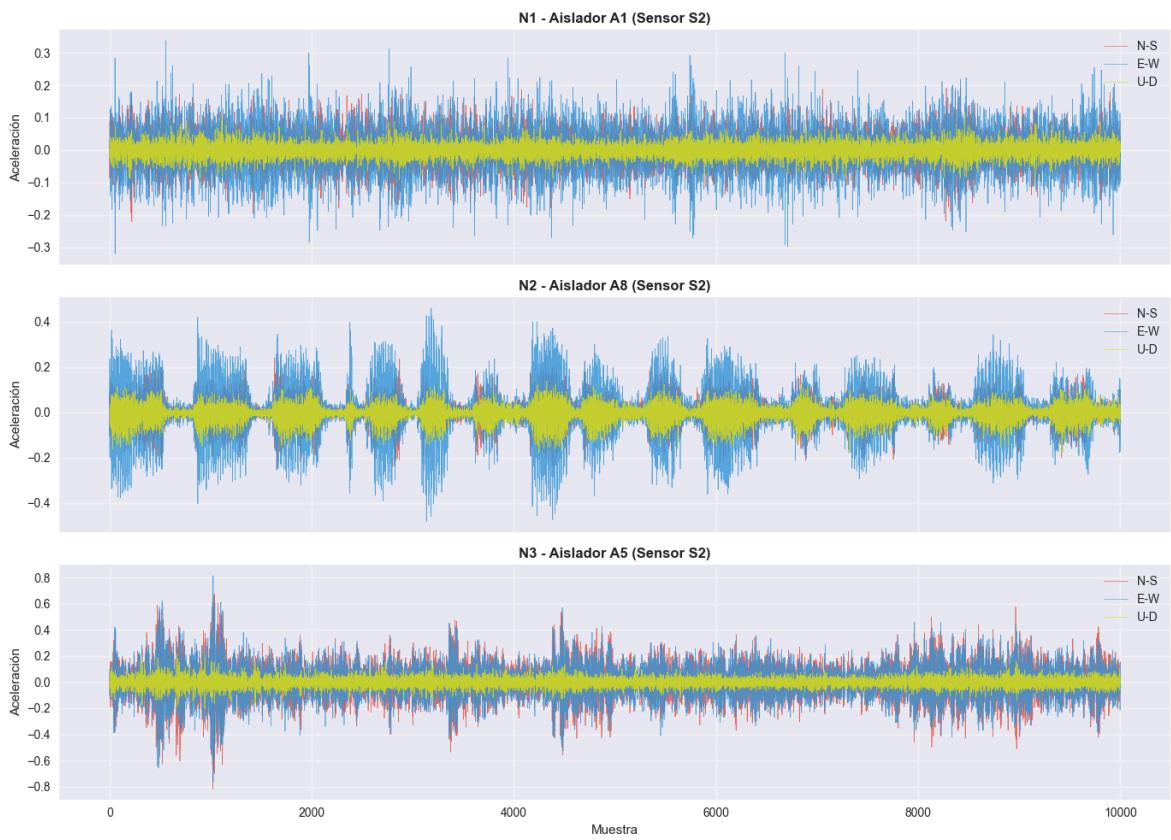
Mínimo: 58,700 muestras
Máximo: 141,800 muestras
Mediana: 81,850 muestras

4. Visualización de Señales de Ejemplo

Mostramos una señal representativa de cada nivel de daño.

```
In [58]: # Seleccionar un espécimen/aislador de cada nivel
sample_aisladores = {
    'N1': df_labels[df_labels['nivel_dano'] == 'N1']['specimen_id'].iloc[0]
```

```
'N2': df_labels[df_labels['nivel_dano'] == 'N2']['specimen_id'].iloc[  
'N3': df_labels[df_labels['nivel_dano'] == 'N3']['specimen_id'].iloc[  
}  
  
fig, axes = plt.subplots(3, 1, figsize=(14, 10), sharex=True)  
colors_axis = {'N-S': '#e74c3c', 'E-W': '#3498db', 'U-D': "#dfe309"}  
  
for idx, (nivel, specimen_id) in enumerate(sample_aisladores.items()):  
    # Cargar señal S2 (respuesta del aislador)  
    specimen_path = os.path.join(SIGNALS_DIR, specimen_id)  
    s2_file = [f for f in os.listdir(specimen_path) if 'S2' in f][0]  
    # Leer archivo txt separado por espacios  
    df_signal = pd.read_csv(os.path.join(specimen_path, s2_file), sep='\s+'  
  
    # Tomar primeras 10000 muestras para visualización  
    df_plot = df_signal.head(10000)  
  
    # Plotear cada eje  
    for col, color in colors_axis.items():  
        if col in df_plot.columns:  
            axes[idx].plot(df_plot[col].values, label=col, color=color, a  
  
    axes[idx].set_title(f'{nivel} - Aislador {specimen_id} (Sensor S2)',  
    axes[idx].set_ylabel('Aceleración')  
    axes[idx].legend(loc='upper right')  
    axes[idx].grid(True, alpha=0.3)  
  
    axes[2].set_xlabel('Muestra')  
plt.tight_layout()  
plt.show()  
  
print(f"\nAisladores mostrados:")  
for nivel, specimen in sample_aisladores.items():  
    tipo = df_labels[df_labels['specimen_id'] == specimen]['tipo'].values  
    print(f"  {nivel}: {specimen} (Tipo {tipo})")
```



\Aisladores mostrados:

- N1: A1 (Tipo B)
- N2: A8 (Tipo B)
- N3: A5 (Tipo A)

5. Conclusiones del EDA

Hallazgos principales:

1. **Desbalance de clases:** Dataset desbalanceado, especialmente en N3 (daño severo) con muy pocos ejemplos.
2. **Mediciones vs Aisladores:** Hay más mediciones que aisladores únicos debido a variantes (-2, -3) del mismo aislador.
3. **Longitud de señales:** Varían entre ~58,700 y ~170,000 muestras. Requieren estandarización.

Implicaciones para el modelado:

- Considerar técnicas de balanceo para modelos supervisados
- Decidir si usar todas las mediciones o solo aisladores únicos según el objetivo del análisis