

# Herramientas de Visualización Alternativas - Día 5

[Change Status](#)

in-progress

 40 min

 Learning Objectives

- 1
- Explorar bibliotecas de visualización más allá de Matplotlib básico

2

 Aprender Seaborn para visualizaciones estadísticas elegantes

3

 Comprender Plotly para gráficos interactivos web

4

 Conocer las fortalezas de cada herramienta para diferentes casos

Theory

Practice

Quiz

Evidence

 Actividades y Aprendizajes

Aprende todo sobre funciones y módulos en Python con ejemplos prácticos.

## Task 1: Seaborn para Visualización Estadística (10 minutos)

### ¿Por qué Seaborn sobre Matplotlib?

Seaborn es una librería de alto nivel construida sobre Matplotlib que facilita la creación de visualizaciones estadísticas elegantes con menos código y mejores valores por defecto.

### Ventajas principales:

Sintaxis más simple y expresiva  
Paletas de colores profesionales integradas  
Funciones especializadas para estadística  
Integración nativa con Pandas DataFrames

### Ejemplo básico de comparación:





```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np

# Datos de ejemplo
np.random.seed(42)
data = pd.DataFrame({
    'x': np.random.randn(100),
    'y': np.random.randn(100),
    'categoria': np.random.choice(['A', 'B', 'C'], 100)
})

# Matplotlib básico (más código)
plt.figure(figsize=(10, 6))
for cat in data['categoria'].unique():
    subset = data[data['categoria'] == cat]
    plt.scatter(subset['x'], subset['y'], label=cat, alpha=0.7)
plt.xlabel('Variable X')
plt.ylabel('Variable Y')
plt.title('Scatter Plot con Matplotlib')
plt.legend()
plt.grid(True, alpha=0.3)
```

## Task 2: Plotly para Visualizaciones Interactivas (10 minutos)

### ¿Cuándo usar visualizaciones interactivas?

Plotly crea gráficos web interactivos que permiten explorar datos dinámicamente, ideal para dashboards y presentaciones interactivas.

### Características clave:

- Interactividad nativa (zoom, hover, filtros)
- Gráficos 3D y animados
- Exportación a HTML/JavaScript
- Integración con Jupyter y web apps

### Ejemplo de gráfico interactivo:

```
import plotly.express as px
import plotly.graph_objects as go
```



```
# Datos de ventas mensuales
ventas_mensuales = pd.DataFrame({
    'mes': pd.date_range('2024-01-01', periods=12, freq='M'),
    'ventas': np.random.randint(10000, 50000, 12),
    'region': np.random.choice(['Norte', 'Sur', 'Este', 'Oeste'], 12)
})

# Gráfico interactivo con Plotly
fig = px.line(ventas_mensuales, x='mes', y='ventas',
              color='region', title='Ventas Mensuales por Región')
fig.update_layout(hovermode='x unified')
fig.show()
```

### Task 3: Pandas Plotting Integrado (10 minutos)

#### ¿Por qué usar plotting integrado?

Pandas incluye métodos de visualización que facilitan gráficos rápidos directamente desde DataFrames sin importar otras librerías.

#### Casos de uso:

- Prototipado rápido de visualizaciones
- Exploración inicial de datos
- Gráficos simples en análisis preliminar
- Integración con workflows de Pandas





#### Ejemplo de plotting integrado:

```
# Dataset de ejemplo
df_ventas = pd.DataFrame({
    'fecha': pd.date_range('2024-01-01', periods=100),
    'ventas': np.random.normal(1000, 200, 100),
    'categoria': np.random.choice(['Electrónicos', 'Ropa', 'Hogar'], 100)
})

# Gráficos rápidos con pandas
# Histograma de ventas
df_ventas['ventas'].plot.hist(bins=20, figsize=(10, 6),
                              title='Distribución de Ventas')

# Serie temporal
```



-  Dashboard
-  Career Path
-  Forms
-  Profile

```
df_ventas.set_index('fecha')['ventas'].plot(figsize=(12, 6),
                                              title='Tendencia de Ventas')

# Box plot por categoría
df_ventas.boxplot(column='ventas', by='categoria', figsize=(10, 6))
```