Documentación librería Seaborn

Básicos de Seaborn

Seaborn es una librería de visualización basada en matplotlib que facilita la creación de gráficos estadísticos con alto nivel estético y menor código. Está integrada con pandas, por lo que funciona directamente sobre DataFrames.

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

# Cargar un fichero externo en un DataFrame (opcional)
fichero_computer = pd.read_csv('ComputerSales.csv')

# Mostrar nombres de datasets internos disponibles en Seaborn
print(sns.get_dataset_names())

# Cargar un dataset interno de Seaborn
dataset_interno = sns.load_dataset('car_crashes')
dataset_interno

```

Seaborn incluye varios datasets integrados que se pueden usar directamente para practicar. En este caso, se utiliza el dataset car_crashes, que contiene estadísticas de accidentes de tráfico por estado.

Gráficos básicos

```
# Histograma (distplot sin KDE)
sns.distplot(dataset_interno['not_distracted'], bins=25, kde=False)

# Joint plot con regresión
sns.jointplot(x='speeding', y='alcohol', data=dataset_interno, kind='reg')

# KDE Plot (curva de densidad)
sns.kdeplot(dataset_interno['not_distracted'])

# Pair plot (relación entre todas las variables)
sns.pairplot(dataset_interno)

# Pair plot con paleta personalizada
sns.pairplot(dataset_interno, palette='Reds')

# Rug plot (marcas individuales sobre un eje)
sns.rugplot(dataset_interno['not_distracted'])

```

Explicación:

- sns.distplot: muestra la distribución de una variable (histograma + curva de densidad). Usar kde=False para ocultar la curva.
- sns.jointplot: gráfico combinado de dispersión + histogramas marginales. El parámetro kind='reg' añade una línea de regresión.
- sns.kdeplot: representa una estimación de densidad kernel (curva suave de distribución).
- sns.pairplot: muestra todas las relaciones posibles entre variables numéricas en forma de dispersión y distribución.

- sns.rugplot: dibuja marcas pequeñas sobre un eje para mostrar la distribución de los datos de forma más básica.

Estos son algunos de los gráficos más utilizados en exploración de datos y análisis estadístico con Seaborn.

Estilo en Seaborn

Seaborn permite modificar fácilmente el estilo visual de los gráficos para adaptarlos a distintos contextos (presentaciones, publicaciones, pantallas pequeñas, etc.). Esto se logra mediante funciones como set_style, set_context y despine.

```
1. sns.set_style('white')
2. plt.figure(figsize=(8,4))
3.
4. sns.set_context('poster', font_scale=1.4)
5. sns.jointplot(x='speeding', y='alcohol', data=dataset_interno, kind='reg')
6. sns.despine(left=False, right=False)
7.
```

Explicación:

- sns.set_style('white'): define el estilo general del gráfico. Otras opciones incluyen 'darkgrid', 'whitegrid', 'dark', y 'ticks'. En este caso, se elimina el fondo cuadriculado y se utiliza un fondo blanco limpio.
- plt.figure(figsize=(8,4)): configura el tamaño de la figura en pulgadas (ancho x alto).
- sns.set_context('poster', font_scale=1.4): ajusta el **tamaño de fuente y elementos gráficos** según el contexto:
    - 'paper': para publicaciones.
    - 'notebook': para uso general.
    - 'talk': para presentaciones.
    - 'poster': para gráficos grandes. El parámetro font_scale aumenta proporcionalmente el tamaño de las fuentes.
- sns.jointplot(...): genera un gráfico combinado de dispersión y regresión como se ha visto anteriormente.
- sns.despine(left=False, right=False): elimina los bordes (spines) de los ejes. Si se desea eliminar los bordes izquierdo y derecho, se puede poner True.

Estas configuraciones permiten crear visualizaciones más limpias, legibles y adaptadas al medio en que se presentarán.

Gráficos categóricos en Seaborn

Seaborn ofrece una amplia gama de gráficos para analizar datos categóricos. Estos permiten visualizar distribuciones, comparaciones y relaciones entre categorías y variables numéricas, facilitando la exploración de patrones en los datos.

```
1. ## Gráfico de barras
2. propinas = sns.load_dataset('tips')
```

```
3. sns.barplot(x='sex', y='total_bill', data=propinas, estimator=np.median)
4.
```

- sns.barplot(...): gráfico de barras que resume valores numéricos por categoría.
    - estimator=np.median: usa la **mediana** como medida de agregación (por defecto es la media).

```
1. # Contar categorías
2. sns.countplot(x='sex', data=propinas)
3.
```

- sns.countplot(...): muestra el **conteo de observaciones** por categoría.

```
1. # Gráfico de cajas y bigotes
2. sns.boxplot(x='day', y='total_bill', data=propinas, hue='sex')
3.
```

- sns.boxplot(...): representa la **distribución estadística** de una variable numérica por categoría.
    - Muestra la mediana, cuartiles y posibles valores atípicos.
    - hue='sex': separa por subcategorías.

```
1. # Gráfico de violín
2. sns.violinplot(x='day', y='total_bill', data=propinas, hue='sex', split=True)
3.
```

- sns.violinplot(...): combina un boxplot con una estimación de densidad para mostrar la distribución.
    - split=True: divide el gráfico por hue en la misma figura.

```
1. # Gráfico de bandas (strip plot)
2. sns.stripplot(x='day', y='total_bill', data=propinas, jitter=True, hue='sex',
dodge=True)
3.
```

- sns.stripplot(...): muestra cada punto individual con **dispersión (jitter)** para evitar superposición.
    - dodge=True: separa los puntos por subcategoría.

Gráfico de enjambre (swarm plot)

```
1. sns.violinplot(x='day', y='total_bill', data=propinas)
2. sns.swarmplot(x='day', y='total_bill', data=propinas, color='white')
3.
```

- sns.swarmplot(...): similar a stripplot, pero evita completamente la superposición de puntos.
    - Combinado con violinplot para mostrar **la distribución general** y **los puntos individuales**.

Estos gráficos son esenciales para análisis exploratorio de datos categóricos y permiten detectar diferencias, patrones y distribuciones en función de variables cualitativas.

Gráficos matriciales en Seaborn

Los gráficos matriciales permiten visualizar relaciones complejas entre múltiples variables. Se utilizan para representar correlaciones, agrupaciones o combinaciones múltiples de variables numéricas y categóricas en una sola figura.

```
## Mapas de calor
dataset_interno = sns.load_dataset('car_crashes')
plt.figure(figsize=(8,6))

sns.set_context('paper', font_scale=1.4)
dataset_interno.drop(columns=['abbrev'], inplace=True)
corr_dataset = dataset_interno.corr()

sns.heatmap(corr_dataset, annot=True, cmap='Blues')
```

- sns.heatmap(...): crea un **mapa de calor** que muestra la matriz de correlación entre las variables del dataset.
    - annot=True: muestra los valores numéricos.
    - cmap='Blues': define la paleta de colores.

```
vuelos = sns.load_dataset("flights")
plt.figure(figsize=(8,6))
sns.set_context('paper', font_scale=1.4)
vuelos = vuelos.pivot_table(index='month', columns='year', values='passengers')
sns.heatmap(vuelos, cmap='Blues', linecolor='white', linewidth=2)

```

- Se convierte un DataFrame en **tabla dinámica** para visualizar el número de pasajeros por mes y año.
- El mapa de calor muestra la evolución temporal de los datos.

Mapas de clúster

```
iris = sns.load_dataset('iris')
species = iris.pop('species')

sns.clustermap(iris)

sns.clustermap(vuelos, cmap="Blues", standard_scale=1)

```

- sns.clustermap(...): realiza un **análisis jerárquico de clúster** y lo muestra como un mapa de calor con agrupaciones automáticas por similitud.
    - standard_scale=1: normaliza las columnas.

Matriz de gráficos personalizados

```
iris = sns.load_dataset("iris")

matriz = sns.PairGrid(iris, hue='species')
matriz.map_upper(sns.scatterplot)
matriz.map_lower(sns.kdeplot, fill=True)
matriz.map_diag(sns.histplot, kde=True)

matriz.add_legend()
plt.show()

```

- sns.PairGrid(...): permite construir una **matriz de gráficos personalizada**.

- map_upper(...): define los gráficos de la parte superior.
- map_lower(...): define los gráficos de la parte inferior.
- map_diag(...): define los gráficos de la diagonal.
- hue='species': colorea según la especie de flor.

Gráficos de regresión

```
1. propinas = sns.load_dataset("tips")
2. plt.figure(figsize=(8,6))
3. sns.set_context('paper', font_scale=1.4)
4. sns.lmplot(x='total_bill', y='tip', hue='sex', data=propinas, markers=["o", "^"])
5.
```

- sns.lmplot(...): gráfico de dispersión con línea de **regresión lineal** ajustada.
  - hue='sex': separa por grupo (hombre/mujer).
  - markers: define símbolos diferentes para cada grupo.

Estos gráficos permiten descubrir patrones, relaciones y agrupaciones complejas en los datos de manera visual y efectiva.