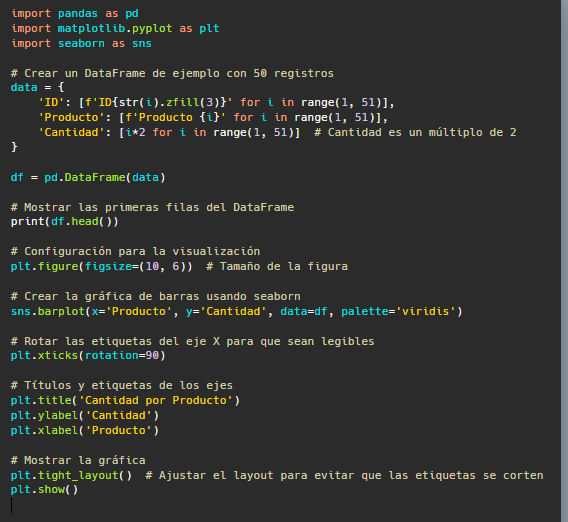
1. Grafica de barras



import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Crear un DataFrame de ejemplo con 50 registros

data = {

'ID': [f'ID{str(i).zfill(3)}' for i in range(1, 51)],

'Producto': [f'Producto {i}' for i in range(1, 51)],

'Cantidad': [i\*2 for i in range(1, 51)] # Cantidad es un múltiplo de 2

}

df = pd.DataFrame(data)

# Mostrar las primeras filas del DataFrame

print(df.head())

# Configuración para la visualización

plt.figure(figsize=(10, 6)) # Tamaño de la figura

# Crear la gráfica de barras usando seaborn

sns.barplot(x='Producto', y='Cantidad', data=df, palette='viridis')

# Rotar las etiquetas del eje X para que sean legibles

plt.xticks(rotation=90)

# Títulos y etiquetas de los ejes

plt.title('Cantidad por Producto')

plt.ylabel('Cantidad')

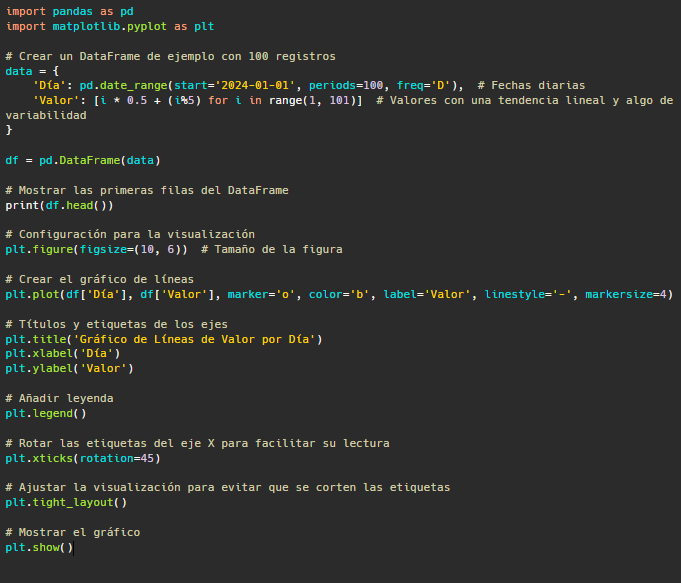
plt.xlabel('Producto')

# Mostrar la gráfica

plt.tight\_layout() # Ajustar el layout para evitar que las etiquetas se corten

plt.show()

1. Grafico de líneas



import pandas as pd

import matplotlib.pyplot as plt

# Crear un DataFrame de ejemplo con 100 registros

data = {

'Día': pd.date\_range(start='2024-01-01', periods=100, freq='D'), # Fechas diarias

'Valor': [i \* 0.5 + (i%5) for i in range(1, 101)] # Valores con una tendencia lineal y algo de variabilidad

}

df = pd.DataFrame(data)

# Mostrar las primeras filas del DataFrame

print(df.head())

# Configuración para la visualización

plt.figure(figsize=(10, 6)) # Tamaño de la figura

# Crear el gráfico de líneas

plt.plot(df['Día'], df['Valor'], marker='o', color='b', label='Valor', linestyle='-', markersize=4)

# Títulos y etiquetas de los ejes

plt.title('Gráfico de Líneas de Valor por Día')

plt.xlabel('Día')

plt.ylabel('Valor')

# Añadir leyenda

plt.legend()

# Rotar las etiquetas del eje X para facilitar su lectura

plt.xticks(rotation=45)

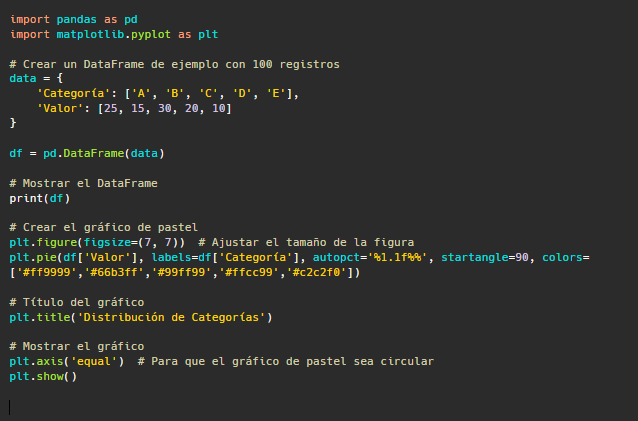
# Ajustar la visualización para evitar que se corten las etiquetas

plt.tight\_layout()

# Mostrar el gráfico

plt.show()

3.grafico de pie



import pandas as pd

import matplotlib.pyplot as plt

# Crear un DataFrame de ejemplo con 100 registros

data = {

'Categoría': ['A', 'B', 'C', 'D', 'E'],

'Valor': [25, 15, 30, 20, 10]

}

df = pd.DataFrame(data)

# Mostrar el DataFrame

print(df)

# Crear el gráfico de pastel

plt.figure(figsize=(7, 7)) # Ajustar el tamaño de la figura

plt.pie(df['Valor'], labels=df['Categoría'], autopct='%1.1f%%', startangle=90, colors=['#ff9999','#66b3ff','#99ff99','#ffcc99','#c2c2f0'])

# Título del gráfico

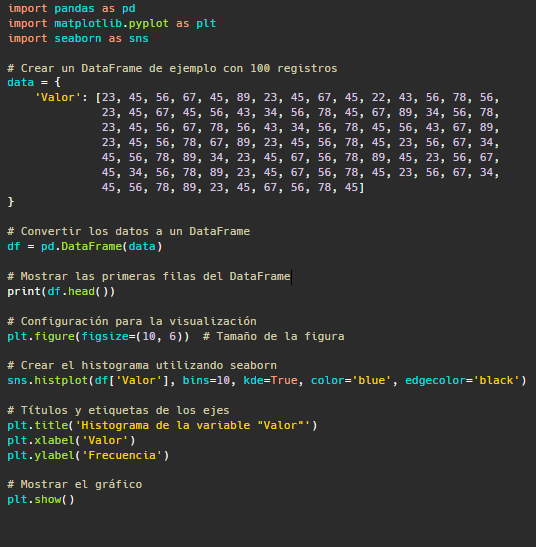
plt.title('Distribución de Categorías')

# Mostrar el gráfico

plt.axis('equal') # Para que el gráfico de pastel sea circular

plt.show()

4. Un **histograma** es una herramienta gráfica utilizada en estadística para representar la distribución de un conjunto de datos numéricos. Se usa principalmente para visualizar la frecuencia con la que ocurren ciertos rangos de valores dentro de un conjunto de datos.



import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Crear un DataFrame de ejemplo con 100 registros

data = {

'Valor': [23, 45, 56, 67, 45, 89, 23, 45, 67, 45, 22, 43, 56, 78, 56,

23, 45, 67, 45, 56, 43, 34, 56, 78, 45, 67, 89, 34, 56, 78,

23, 45, 56, 67, 78, 56, 43, 34, 56, 78, 45, 56, 43, 67, 89,

23, 45, 56, 78, 67, 89, 23, 45, 56, 78, 45, 23, 56, 67, 34,

45, 56, 78, 89, 34, 23, 45, 67, 56, 78, 89, 45, 23, 56, 67,

45, 34, 56, 78, 89, 23, 45, 67, 56, 78, 45, 23, 56, 67, 34,

45, 56, 78, 89, 23, 45, 67, 56, 78, 45]

}

# Convertir los datos a un DataFrame

df = pd.DataFrame(data)

# Mostrar las primeras filas del DataFrame

print(df.head())

# Configuración para la visualización

plt.figure(figsize=(10, 6)) # Tamaño de la figura

# Crear el histograma utilizando seaborn

sns.histplot(df['Valor'], bins=10, kde=True, color='blue', edgecolor='black')

# Títulos y etiquetas de los ejes

plt.title('Histograma de la variable "Valor"')

plt.xlabel('Valor')

plt.ylabel('Frecuencia')

# Mostrar el gráfico

plt.show()

 Usamos sns.histplot() de la biblioteca seaborn para generar el histograma.

 df['Valor']: Especificamos la columna de datos que queremos graficar.

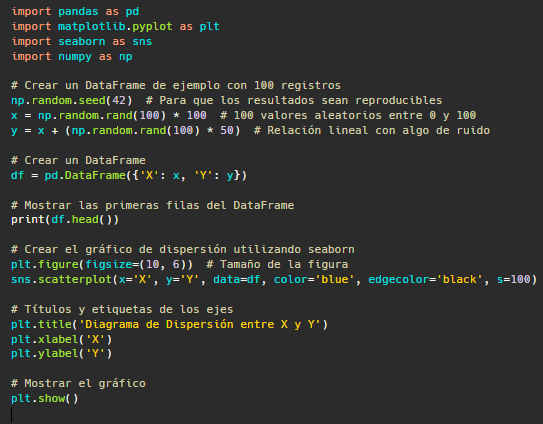
 bins=10: Especificamos cuántas "barras" (intervalos) tendrá el histograma. En este caso, se elige 10, pero puedes ajustarlo según tus necesidades.

 kde=True: Añade una curva de densidad sobre el histograma para mostrar la distribución de los datos.

 color='blue': Define el color de las barras del histograma.

 edgecolor='black': Define el color del borde de las barras.

5.grafica de dispersion :



import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

# Crear un DataFrame de ejemplo con 100 registros

np.random.seed(42) # Para que los resultados sean reproducibles

x = np.random.rand(100) \* 100 # 100 valores aleatorios entre 0 y 100

y = x + (np.random.rand(100) \* 50) # Relación lineal con algo de ruido

# Crear un DataFrame

df = pd.DataFrame({'X': x, 'Y': y})

# Mostrar las primeras filas del DataFrame

print(df.head())

# Crear el gráfico de dispersión utilizando seaborn

plt.figure(figsize=(10, 6)) # Tamaño de la figura

sns.scatterplot(x='X', y='Y', data=df, color='blue', edgecolor='black', s=100)

# Títulos y etiquetas de los ejes

plt.title('Diagrama de Dispersión entre X y Y')

plt.xlabel('X')

plt.ylabel('Y')

# Mostrar el gráfico

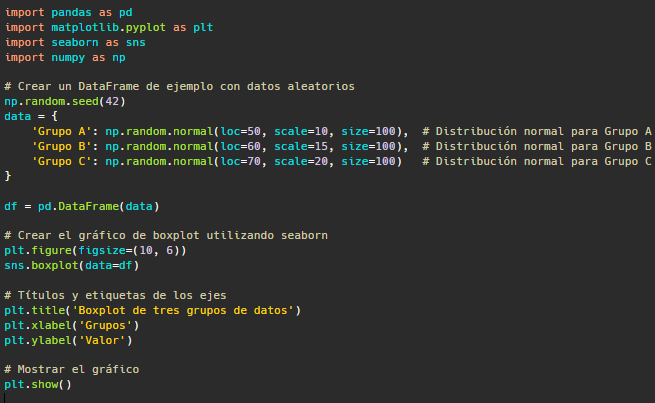
plt.show()

6.Boxplots :

Un **boxplot** (o diagrama de caja) es una herramienta estadística gráfica que resume un conjunto de datos numéricos mostrándolos a través de sus cuartiles, incluyendo valores atípicos (outliers). Es muy útil para entender la distribución, la simetría y la dispersión de los datos.

**Usos principales de los boxplots:**

1. **Identificación de la mediana, cuartiles y rango intercuartil (IQR):**
   * Muestra la mediana, los cuartiles (Q1, Q3), el rango intercuartil (IQR), y los valores atípicos.
2. **Detección de outliers (valores atípicos):**
   * Los valores fuera de los bigotes (líneas horizontales) del boxplot se consideran outliers. Ayuda a identificar si hay valores anómalos en el conjunto de datos.
3. **Comparación de distribuciones:**
   * Cuando se tienen múltiples conjuntos de datos, un boxplot permite comparar sus distribuciones fácilmente. Es útil para comparar varios grupos o variables.
4. **Visualización de la simetría y asimetría de los datos:**
   * Ayuda a ver si la distribución de los datos es simétrica, sesgada a la derecha o a la izquierda.



import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

# Crear un DataFrame de ejemplo con datos aleatorios

np.random.seed(42)

data = {

'Grupo A': np.random.normal(loc=50, scale=10, size=100), # Distribución normal para Grupo A

'Grupo B': np.random.normal(loc=60, scale=15, size=100), # Distribución normal para Grupo B

'Grupo C': np.random.normal(loc=70, scale=20, size=100) # Distribución normal para Grupo C

}

df = pd.DataFrame(data)

# Crear el gráfico de boxplot utilizando seaborn

plt.figure(figsize=(10, 6))

sns.boxplot(data=df)

# Títulos y etiquetas de los ejes

plt.title('Boxplot de tres grupos de datos')

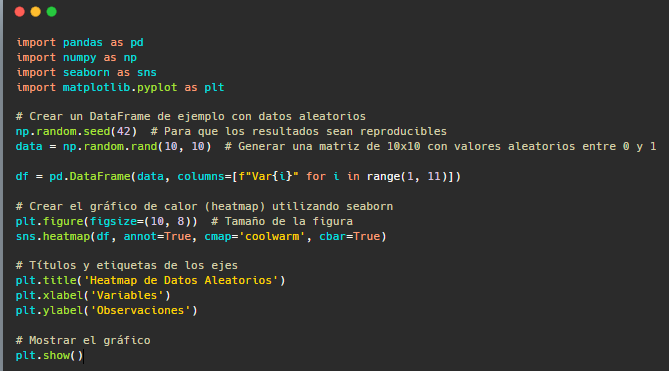
plt.xlabel('Grupos')

plt.ylabel('Valor')

# Mostrar el gráfico

plt.show()

7.Mapa de Calor



import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

# Crear un DataFrame de ejemplo con datos aleatorios

np.random.seed(42)  # Para que los resultados sean reproducibles

data = np.random.rand(10, 10)  # Generar una matriz de 10x10 con valores aleatorios entre 0 y 1

df = pd.DataFrame(data, columns=[f"Var{i}" for i in range(1, 11)])

# Crear el gráfico de calor (heatmap) utilizando seaborn

plt.figure(figsize=(10, 8))  # Tamaño de la figura

sns.heatmap(df, annot=True, cmap='coolwarm', cbar=True)

# Títulos y etiquetas de los ejes

plt.title('Heatmap de Datos Aleatorios')

plt.xlabel('Variables')

plt.ylabel('Observaciones')

# Mostrar el gráfico

plt.show()