Sesión 2: Estadística descriptiva con Matplotlib (2 horas)

2.1 Introducción a la estadística descriptiva

2.1.1 Medidas de tendencia central: media, mediana, moda

La **media** es el promedio aritmético de los valores en un conjunto de datos. La **mediana** es el valor central que separa los datos en dos partes iguales. La **moda** es el valor que ocurre con mayor frecuencia en el conjunto de datos.

```
import numpy as np

data = np.array([4, 5, 6, 4, 6, 7, 8, 4, 9, 10])

media = np.mean(data)
mediana = np.median(data)
moda = np.bincount(data).argmax()

print("Media:", media)
print("Mediana:", mediana)
print("Moda:", moda)
```

2.1.2 Medidas de dispersión: rango, varianza, desviación estándar

El **rango** es la diferencia entre el valor máximo y el valor mínimo del conjunto de datos. La **varianza** es el promedio de las diferencias al cuadrado entre los valores y la media. La **desviación estándar** es la raíz cuadrada de la varianza.

```
rango = np.max(data) - np.min(data)
varianza = np.var(data)
desviacion_estandar = np.std(data)

print("Rango:", rango)
print("Varianza:", varianza)
print("Desviación estándar:", desviacion_estandar)
```

2.1.3 Cuartiles y diagrama de caja y bigotes (boxplot)

Los **cuartiles** son valores que dividen un conjunto de datos en cuatro partes iguales. El primer cuartil (Q1) es el valor que separa el 25% inferior de los datos, el segundo cuartil (Q2) es la mediana, y el tercer cuartil (Q3) es el valor que separa el 75% inferior de los datos.

```
Q1, Q2, Q3 = np.percentile(data, [25, 50, 75])

print("Primer cuartil (Q1):", Q1)
print("Segundo cuartil (Q2 - Mediana):", Q2)
print("Tercer cuartil (Q3):", Q3)
```

2.2 Introducción a Matplotlib

2.2.1 Instalación y configuración

```
pip install matplotlib
```

```
import matplotlib.pyplot as plt
plt.style.use("ggplot")
```

2.2.2 Creación de gráficos básicos: histogramas, barras, pastel

```
# Histograma
data_hist = np.random.randn(1000)
plt.hist(data_hist, bins=30)
plt.title("Histograma")
plt.xlabel("Valores")
plt.ylabel("Frecuencia")
plt.show()

# Gráfico de barras
categorias = ["A", "B", "C", "D", "E"]
valores = [3, 7, 2, 5, 8]
plt.bar(categorias, valores)
plt.title("Gráfico de barras")
plt.xlabel("Categorías")
plt.ylabel("Valores")
plt.show()

# Gráfico de pastel
labels = ["Python", "Java", "C++", "JavaScript", "Ruby"]
sizes = [45, 30, 15, 8, 2]
colors = ["blue", "green", "red", "yellow", "purple"]
explode = (0.1, 0, 0, 0, 0)

plt.pie(sizes, labels=labels, colors=colors, explode=explode,
autopct="%1.1f%%", shadow=True, startangle=90)
plt.title("Lenguajes de programación")
plt.axis("equal")
plt.show()
```

Con este bloque de código, hemos creado un gráfico de pastel que muestra la proporción de distintos lenguajes de programación. Las variables `labels` y `sizes` representan las etiquetas y los tamaños de las porciones del gráfico de pastel, respectivamente. La variable `colors` contiene los colores de las porciones y `explode` indica qué porción debe sobresalir del gráfico. La función `plt.pie()` crea el gráfico de pastel y la función `plt.show()` lo muestra en pantalla.

2.3 Visualización de estadísticas descriptivas con Matplotlib

2.3.1 Histogramas y gráficos de densidad

Los **histogramas** y **gráficos de densidad** son útiles para visualizar la distribución de los datos. Utilice `plt.hist` para crear histogramas y `plt.plot` con un objeto de densidad de Kernel para crear gráficos de densidad.

```
import seaborn as sns

# Histograma
data_hist = np.random.randn(1000)
plt.hist(data_hist, bins=30, density=True, alpha=0.6, color='g')
plt.title("Histograma")
plt.xlabel("Valores")
plt.ylabel("Densidad")
plt.show()

# Gráfico de densidad
sns.kdeplot(data_hist, color="r")
plt.title("Gráfico de densidad")
plt.xlabel("Valores")
plt.ylabel("Densidad")
plt.show()
```

2.3.2 Gráficos de caja y bigotes (boxplot)

Utilice `plt.boxplot` para crear gráficos de caja y bigotes que muestren la dispersión de los datos y los posibles valores atípicos.

```
data_box = [np.random.normal(0, std, 100) for std in range(1, 4)]

plt.boxplot(data_box, vert=True, patch_artist=True)
plt.title("Gráfico de caja y bigotes")
plt.xlabel("Categorías")
plt.ylabel("Valores")
plt.show()
```

2.3.3 Gráficos de dispersión y correlación

Los **gráficos de dispersión** son útiles para visualizar la relación entre dos variables numéricas. Los gráficos de correlación muestran la fuerza y la dirección de una relación lineal entre dos variables.

```python
x = np.random.randn(100)
y = 2 * x + np.random.randn(100)

plt.scatter(x, y, marker="o", color="b")
plt.title("Gráfico de dispersión")
plt.xlabel("Variable X")
plt.ylabel("Variable Y")
plt.show()

correlacion = np.corrcoef(x, y)[0, 1]
print("Correlación:", correlacion)
```

2.4 Ejercicios prácticos con Python y Matplotlib

1. Utilice un conjunto de datos de su elección y calcule las medidas de tendencia central y de dispersión. Muestre los resultados en un gráfico de barras.
2. Cree un gráfico de pastel que muestre la proporción de diferentes categorías en su conjunto de datos. Por ejemplo, si su conjunto de datos contiene información sobre las ventas de productos, muestre la proporción de ventas de cada producto.
3. Dado un conjunto de datos con dos variables numéricas, cree un gráfico de dispersión para visualizar la relación entre las dos variables. Calcule la correlación entre las variables y muestre el coeficiente de correlación en el gráfico.