
# Nivel 4 – Laboratorio 2
# Exploración de análisis de datos con Pandas y Matplotlib

## Objetivos

1. Explorar las funcionalidades básicas de la librería Pandas para análisis de datos.
2. Comprender qué son las Series y DataFrames y cómo se utilizan.
3. Investigar y aplicar métodos de conteo, agrupación y ordenamiento de datos.
4. Familiarizarse con diferentes tipos de visualizaciones usando Pandas y Matplotlib.
5. Desarrollar habilidades de documentación técnica y experimentación con librerías.

## Preparación del ambiente de trabajo

Este laboratorio es de **exploración**, lo que significa que usted investigará y experimentará con diferentes funcionalidades de estas librerías para comprender cómo funcionan. Deberá crear un archivo `exploracion_datos.py` donde implementará sus experimentos.

**Recursos de documentación oficial:**

- Pandas: `https://pandas.pydata.org/docs/`
- Matplotlib: `https://matplotlib.org/stable/contents.html`

# Introducción a Pandas

Pandas es una librería de Python especializada en el análisis y manipulación de datos. Se importa convencionalmente como `pd`:

```
import pandas as pd
```

Pandas trabaja principalmente con dos estructuras de datos:

- **Series**: Una estructura unidimensional similar a una lista que puede contener diferentes tipos de datos (números, texto, etc.). Cada elemento tiene un índice asociado.
- **DataFrame**: Una estructura bidimensional similar a una tabla o matriz, donde los datos se organizan en filas y columnas. Cada columna es una Serie.

**Ejemplo de creación de una Serie:**

```
edades = pd.Series([23, 45, 18, 67, 34, 23, 45, 23])
```

**Ejemplo de creación de un DataFrame:**

```
datos = {
    "nombre": ["Ana", "Luis", "María", "Pedro"],
    "edad": [23, 45, 18, 67],
    "ciudad": ["Bogotá", "Cali", "Bogotá", "Medellín"]
}
df = pd.DataFrame(datos)
```

# Actividad 1: Exploración de conteo en Series

**Investigación:** Pandas proporciona métodos para contar elementos en una Serie. Investigue en la documentación oficial qué hacen los siguientes métodos y cuál es la diferencia entre ellos:

- `pd.Series.value_counts()`
- `pd.Series.count()`

**Experimentación:** Cree la siguiente Serie y aplique ambos métodos para observar sus diferencias:

```
calificaciones = pd.Series([4.5, 3.8, 4.5, 2.9, 4.5, 3.8, 5.0,
                            4.5, None, 3.8, 4.5])
```

**Preguntas para responder:**

1. ¿Qué información retorna `value_counts()`? ¿En qué orden aparecen los resultados?
2. ¿Qué tipo de valor retorna `count()`?
3. ¿Cómo manejan estos métodos los valores `None` (valores nulos)?
4. ¿En qué situaciones usaría cada uno de estos métodos?

# Actividad 2: Exploración de operaciones con DataFrames

**Investigación:** Los DataFrames tienen métodos útiles para visualizar y ordenar datos. Investigue qué hacen los siguientes métodos:

- `pd.DataFrame.head(n)`
- `pd.DataFrame.tail(n)`
- `pd.DataFrame.sort_values(ascending=flag)`

**Experimentación:** Cree el siguiente DataFrame y experimente con estos métodos:

```
estudiantes = pd.DataFrame({
    "nombre": ["Ana", "Luis", "María", "Pedro", "Sofía",
               "Carlos", "Laura", "Diego", "Valentina", "Andrés"],
    "edad": [20, 22, 19, 21, 20, 23, 19, 22, 20, 21],
    "nota": [4.5, 3.2, 4.8, 3.9, 4.1, 3.5, 4.7, 3.8, 4.2, 3.6]
})
```

**Preguntas para responder:**

1. ¿Qué filas muestra `head(3)` y qué filas muestra `tail(3)`?
2. ¿Para qué sirve el parámetro `ascending` en `sort_values()`?
3. Ordene el DataFrame por la columna `nota`. ¿Qué pasa cuando `ascending=True`? ¿Y cuando `ascending=False`?
4. ¿Cómo podría combinar estos métodos para ver solo los 5 estudiantes con mejores notas?

# Actividad 3: Exploración de agrupación y cálculo de promedios

**Investigación:** Pandas permite agrupar datos y calcular estadísticas sobre esos grupos. Investigue qué hacen:

- `pd.DataFrame.groupby()`
- `pd.Series.mean()`

**Experimentación:** Cree el siguiente DataFrame con información de ventas:

```
ventas = pd.DataFrame({
    "producto": ["Laptop", "Mouse", "Laptop", "Teclado", "Mouse",
                 "Laptop", "Mouse", "Teclado", "Laptop", "Mouse"],
    "precio": [2500000, 45000, 2800000, 120000, 50000,
               2600000, 48000, 125000, 2700000, 47000],
    "cantidad": [1, 3, 1, 2, 4, 1, 2, 1, 1, 5]
})
```

**Preguntas para responder:**

1. ¿Qué sucede cuando ejecuta `ventas.groupby("producto")`? ¿Qué tipo de resultado obtiene?
2. Ejecute `ventas.groupby("producto")["precio"].mean()`. ¿Qué información le proporciona?
3. ¿Cómo calcularía el promedio de cantidad vendida para cada producto?
4. Experimente agrupando por producto y calculando el promedio de múltiples columnas a la vez.

# Actividad 4: Exploración de gráficas de barras

**Investigación:** Pandas permite crear visualizaciones directamente desde Series y DataFrames usando el método `plot()`. Matplotlib es la librería que Pandas usa internamente para mostrar estas gráficas. Investigue:

- `pd.Series.plot(kind="bar")`
- `pd.Series.plot(kind="barh")`
- Los parámetros: `figsize`, `title`, `xlabel`, `ylabel`
- `plt.show()` de Matplotlib

Matplotlib se importa convencionalmente como:

```
import matplotlib.pyplot as plt
```

**Experimentación:** Use los datos de la Actividad 3 y cree visualizaciones:

```
precios_promedio = ventas.groupby("producto")["precio"].mean()
```

**Preguntas para responder:**

1. ¿Cuál es la diferencia visual entre `kind="bar"` y `kind="barh"`?
2. ¿Qué hace el parámetro `figsize=(10, 6)`? Experimente con diferentes valores.
3. ¿Para qué sirve `plt.show()`? ¿Qué pasa si no lo incluye?
4. Cree una gráfica de barras horizontales con título `"Precio promedio por producto"`, etiqueta del eje X `"Precio (COP)"` y etiqueta del eje Y `"Producto"`.
5. ¿En qué situaciones preferiría usar barras verticales vs. barras horizontales?

# Actividad 5: Exploración de histogramas

**Investigación:** Los histogramas son útiles para visualizar la distribución de datos numéricos. Investigue:

- `pd.Series.plot(kind="hist")`
- ¿Qué diferencia hay entre un histograma y una gráfica de barras?

**Experimentación:** Cree una Serie con calificaciones de estudiantes:

```
import random
random.seed(42)
calificaciones = pd.Series([random.uniform(2.0, 5.0)
                            for _ in range(100)])
```

**Preguntas para responder:**

1. Cree un histograma de las calificaciones. ¿Qué información muestra?
2. ¿Cómo se agrupan los datos en el eje X? ¿Qué representa el eje Y?
3. Experimente con el parámetro `bins=20` dentro de `plot()`. ¿Qué cambia?
4. ¿En qué situaciones es más útil un histograma que una gráfica de barras?
5. Compare visualmente un histograma con 5 bins versus uno con 50 bins. ¿Cuál ofrece mejor información?

# Actividad 6: Exploración de diagramas de caja (Boxplots)

**Investigación:** Los boxplots o diagramas de caja son útiles para visualizar la distribución de datos y detectar valores atípicos. Investigue:

- `pd.Series.plot(kind="box")`
- El parámetro `vert`
- ¿Qué información muestra un boxplot? (cuartiles, mediana, valores atípicos)

**Experimentación:** Use los datos de estudiantes de la Actividad 2:

```
estudiantes = pd.DataFrame({
    "nombre": ["Ana", "Luis", "María", "Pedro", "Sofía",
               "Carlos", "Laura", "Diego", "Valentina", "Andrés"],
    "edad": [20, 22, 19, 21, 20, 23, 19, 22, 20, 21],
    "nota": [4.5, 3.2, 4.8, 3.9, 4.1, 3.5, 4.7, 3.8, 4.2, 3.6]
})
```

**Preguntas para responder:**

1. Cree un boxplot de la columna `nota`. ¿Qué elementos visuales observa?
2. ¿Qué representa la línea en el medio de la caja?
3. ¿Cuál es la diferencia entre `vert=True` y `vert=False`?
4. Investigue qué significan los "bigotes"(whiskers) y los puntos aislados en un boxplot.
5. ¿Por qué sería útil un boxplot para comparar las notas de diferentes cursos?

# Actividad 7: Exploración de manipulación de ejes

**Investigación:** Matplotlib permite manipular los ejes de las gráficas. Investigue:

- `ax.invert_yaxis()` (donde `ax` es el objeto Axes de Matplotlib)
- ¿Cómo obtener el objeto `ax` al crear una gráfica con Pandas?

**Experimentación:** Use los precios promedio de la Actividad 4:

```
ventas = pd.DataFrame({
    "producto": ["Laptop", "Mouse", "Laptop", "Teclado", "Mouse",
                 "Laptop", "Mouse", "Teclado", "Laptop", "Mouse"],
    "precio": [2500000, 45000, 2800000, 120000, 50000,
               2600000, 48000, 125000, 2700000, 47000]
})

precios_promedio = ventas.groupby("producto")["precio"].mean()
precios_promedio = precios_promedio.sort_values()
```

Para obtener el objeto `ax`, puede usar:

```
ax = precios_promedio.plot(kind="barh", figsize=(10, 6))
```

**Preguntas para responder:**

1. Cree una gráfica de barras horizontales y luego aplique `ax.invert_yaxis()`. ¿Qué cambia?
2. ¿Para qué sería útil invertir el eje Y en una gráfica?
3. Experimente invirtiendo el eje Y en diferentes tipos de gráficas (bar, barh). ¿En cuál tiene más sentido?
4. Si ordena los datos de menor a mayor antes de graficar e invierte el eje Y, ¿cómo quedan visualmente ordenados los elementos?

## Entrega

Entregue un archivo comprimido (ZIP) que contenga:

1. **exploracion_datos.py**: Archivo con todo el código de sus experimentaciones, organizado por actividades. Use comentarios para separar cada actividad claramente.
2. **respuestas.txt** o **respuestas.pdf**: Documento con las respuestas a todas las preguntas planteadas en las actividades. Incluya capturas de pantalla de las gráficas generadas cuando sea relevante.

Suba el archivo comprimido a través de Brightspace en el laboratorio del Nivel 4 designado como "N4-L2: Exploración de análisis de datos con Pandas y Matplotlib".

**Recomendaciones:**

- No se limite a responder las preguntas mínimas. Experimente con variaciones de los ejemplos.
- Consulte la documentación oficial cuando tenga dudas. Aprender a leer documentación es una habilidad fundamental.
- Pruebe combinaciones de métodos que no se mencionan explícitamente en el laboratorio.
- Si encuentra métodos o funcionalidades interesantes en la documentación, inclúyalos en su exploración.
- Comente su código para explicar qué está probando en cada sección.