Deber2

November 13, 2024

Alumno: Ascencio Cruz Cristopher Eduardo

1 Entregables

- Código completo que resuelve los pasos indicados
- Gráficas a criterio personal de cada uno
- Código comentado línea por línea

2 Objetivo de la práctica:

El objetivo de este ejercicio es aplicar funciones estadísticas sobre una matriz de calificaciones utilizando la librería NumPy de Python. Analizarás datos de estudiantes y sus calificacionesen diferentes asignaturas, y calcularás diversas estadísticas

[156]:
```
# Importación de las librerias necesarias
import numpy as np
import matplotlib.pyplot as plt
```

[157]:
```
# Inicialización de la figura a graficar
plt.figure(figsize=(12, 6))
```

[157]:
```
<Figure size 1200x600 with 0 Axes>
```


```
<Figure size 1200x600 with 0 Axes>
```


3 Crear la matriz de calificaciones

Utiliza NumPy para crear una matriz 5x4 que contenga las calificaciones de 5 estudiantes en 4 asignaturas. Imprime la matriz para asegurarte de que se haya creado correctamente.

- Estudiante 1: [7, 8, 6, 5]
- Estudiante 2: [9, 7, 8, 6]
- Estudiante 3: [6, 5, 9, 7]
- Estudiante 4: [8, 6, 7, 8]
- Estudiante 5: [5, 4, 6, 5]

[158]:
```
'''
Matriz de 5x4 que representa las calificaciones de 5 estudiantes
```

```
Donde:
 - Cada fila representa las calificaciones de un estudiante
 - Cada fila columna representa las calificaciones obtenidas por asignatura
'''
calificaciones = np.array([
    [7, 8, 6, 5],
    [9, 7, 8, 6],
    [6, 5, 9, 7],
    [8, 6, 7, 8],
    [5, 4, 6, 5]
])

# Formatear e imprimir la matriz
print("Matriz de calificaciones:")
"""
Formatea e imprime la matriz de calificaciones.

La función `np.array2string` convierte la matriz `calificaciones` en una cadena␣
 ↪de texto con un formato específico.

Parámetros:
- calificaciones: numpy.ndarray
    Matriz de 5x4 que contiene las calificaciones de 5 estudiantes en 4␣
 ↪asignaturas.
- formatter: dict
    Un diccionario que especifica cómo formatear los elementos de la matriz. En␣
 ↪este caso, se utiliza el formato `'{:2d}'.format` para los enteros, lo que␣
 ↪asegura que cada número entero se imprima con al menos dos dígitos de ancho,␣
 ↪alineando así las calificaciones en columnas.

Salida:
- Una cadena de texto que representa la matriz de calificaciones con el formato␣
 ↪especificado.
"""
print(np.array2string(calificaciones, formatter={'int': '{:2d}'.format}))
```

```
Matriz de calificaciones:
[[ 7  8  6  5]
 [ 9  7  8  6]
 [ 6  5  9  7]
 [ 8  6  7  8]
 [ 5  4  6  5]]
```

Definición de una función utilizada para darle formato a las salidas que puedan ser repetitivas.

[159]:
```
def formatear_salida(matriz, texto: str) -> None:
    """
    Formatea y muestra los elementos de una matriz con un texto descriptivo.
```

```
    Args:
        matriz (list of numpy.ndarray): Lista de arrays de numpy a formatear y␣
↪mostrar.
        texto (str): Texto descriptivo que se mostrará antes del índice y el␣
↪elemento.

    Returns:
        None

    Ejemplo de uso:
        matriz = np.array([1.234, 5.678])
        formatear_salida(matriz, "Elemento")
         Elemento 1: 1.23
         Elemento 2: 5.67
    """
    for contador, elemento in enumerate(matriz):
        print(f" {texto} {contador + 1}: {elemento.round(2)}")
```

4 Calcular estadísticas básicas

Funciones a utilizar:

`np.mean()` `np.var()` `np.std()`

4.1 Media por Asignatura

Calcula la media de las calificaciones por cada asignatura (es decir, la media de cada columna de la matriz).

[160]:
```
# Calcular la media por asignatura
mean = lambda matriz : np.mean(matriz, axis = 0)
"""
Calcula la media por asignatura de una matriz de calificaciones.

Nota:
    axis=0: Calcula la media a lo largo de las columnas.
    Esto significa que se toma la media de cada columna,
    lo que resulta en un arreglo donde cada elemento es
    la media de una asignatura (columna).

Parámetros:
matriz (numpy.ndarray): Una matriz de calificaciones donde cada fila representa␣
 ↪un estudiante y cada columna representa una asignatura.

Devuelve:
```

```
numpy.ndarray: Un arreglo que contiene la media de las calificaciones por cada asignatura.
"""
media_asignatura = mean(calificaciones)
print("Media por asignatura:")
formatear_salida(media_asignatura, "Asignatura")
```

```
Media por asignatura:
 Asignatura 1: 7.0
 Asignatura 2: 6.0
 Asignatura 3: 7.2
 Asignatura 4: 6.2
```

4.2 Varianza por estudiante

Calcula la varianza de las calificaciones por cada estudiante (es decir, la varianza de cada fila de la matriz).

[161]:
```
# Calcular la varianza por estudiante
var = lambda matriz : np.var(matriz, axis = 1)
"""
Calcula la varianza de las calificaciones por estudiante.

Parámetros:
matriz (numpy.ndarray): Una matriz de calificaciones donde cada fila representa a un estudiante y cada columna representa una calificación.

Valor de retorno:
numpy.ndarray: Un arreglo que contiene la varianza de las calificaciones para cada estudiante.

Nota:
    El parámetro `axis=1` indica que la
    varianza se calcula a lo largo de las
    filas, es decir, para cada estudiante i
    ndividualmente.
"""

varianza_estudiante = var(calificaciones)
print("\nVarianza por estudiante:")
formatear_salida(varianza_estudiante, "Estudiante")
```

```
Varianza por estudiante:
 Estudiante 1: 1.25
 Estudiante 2: 1.25
 Estudiante 3: 2.19
 Estudiante 4: 0.69
```

```
Estudiante 5: 0.5
```

4.3 Desviación estandar de la matriz completa

Calcula la desviación estándar de todas las calificaciones de la matriz.

[162]:
```
# Calcular desviación estándar de la matriz completa
standard = lambda matriz: np.std(matriz)
"""
Calcula la desviación estándar de una matriz.

Parámetros:
matriz (numpy.ndarray): La matriz de la cual se calculará la desviación␣
 ↪estándar.

Devuelve:
float: La desviación estándar de los elementos de la matriz.
"""
desviacion_estandar_total = standard(calificaciones)
print(f"\nDesviación estándar de la matriz completa: {desviacion_estandar_total:
 ↪.2f}")
```

```
Desviación estándar de la matriz completa: 1.39
```

5 Generación de gráficos

5.1 Media por asignatura

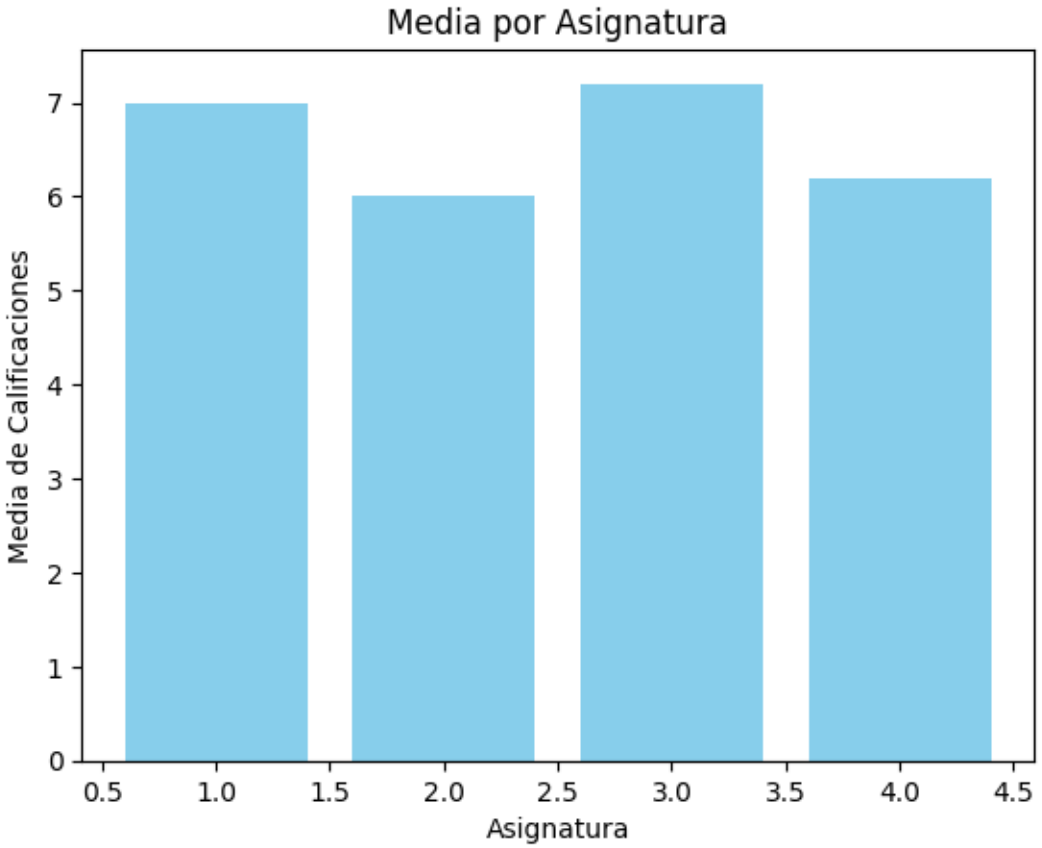
[163]:
```
"""
Genera una gráfica de barras que muestra la media de calificaciones por␣
 ↪asignatura.

Variables:
    media_asignatura (list): Lista de valores que representan la media de␣
 ↪calificaciones para cada asignatura.

Gráfica:
    - Eje X: Asignaturas (numeradas del 1 al 4).
    - Eje Y: Media de calificaciones.
    - Color de las barras: 'skyblue'.
    - Título: "Media por Asignatura".
    - Etiqueta del eje X: "Asignatura".
    - Etiqueta del eje Y: "Media de Calificaciones".
"""
# plt.subplot(1, 2, 1)
plt.bar(range(1, 5), media_asignatura, color='skyblue')
plt.title("Media por Asignatura")
```

```
plt.xlabel("Asignatura")
plt.ylabel("Media de Calificaciones")
```

[163]: Text(0, 0.5, 'Media de Calificaciones')



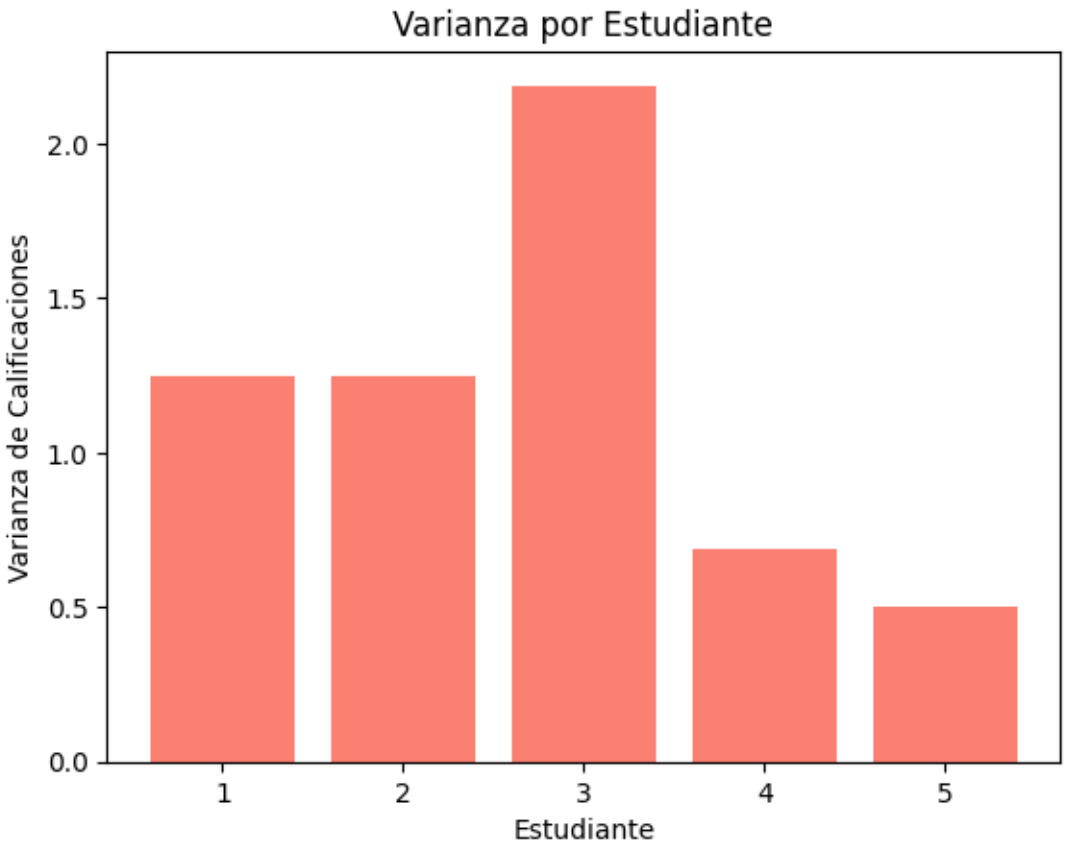
6 Gráfica de la varianza por estudiante

[164]:
```
# plt.subplot(1, 2, 2)
"""
This code snippet creates a bar chart using matplotlib to visualize the␣
 ↪variance of grades for students.

- The `plt.bar` function is used to create a bar chart.
    - The x-axis represents the students, ranging from 1 to 5.
    - The y-axis represents the variance of grades for each student.
    - The bars are colored 'salmon'.
- The `plt.title` function sets the title of the chart to "Varianza por␣
 ↪Estudiante".
- The `plt.xlabel` function labels the x-axis as "Estudiante".
```

```
- The `plt.ylabel` function labels the y-axis as "Varianza de Calificaciones".
"""
plt.bar(range(1, 6), varianza_estudiante, color='salmon')
plt.title("Varianza por Estudiante")
plt.xlabel("Estudiante")
plt.ylabel("Varianza de Calificaciones")
```

[164]: `Text(0, 0.5, 'Varianza de Calificaciones')`



7 Obtener valores únicos y ordenarlos

Utiliza `np.unique()` para obtener los valores únicos de las calificaciones en toda la matriz. Luego, utiliza `sorted(set())` para ordenar esos valores de mayor a menor

[165]:
```
"""
Obtiene los valores únicos de la lista de calificaciones, los ordena y los␣
↪convierte a enteros.
```

```
1. Se utiliza `np.unique` para obtener los valores únicos de la lista `calificaciones`.
2. Se usa `map(int, ...)` para convertir cada valor único a un entero.
3. Se ordenan los valores únicos convertidos a enteros utilizando `sorted`.
4. Se imprime la lista de valores únicos ordenados.

Nota:
    Se utilizó `map` en lugar de `set` para mostrar los datos porque `map`
    permite aplicar una función (en este caso, `int`) a cada elemento de
    una secuencia, mientras que `set` solo elimina duplicados y no realiza
    la conversión de tipo. Al usar `map`, nos aseguramos de que todos los
    valores sean convertidos a enteros antes de ser ordenados.
"""
valores_unicos = sorted(map(int, np.unique(calificaciones)))
print("\nValores únicos ordenados:", valores_unicos)
```

```
Valores únicos ordenados: [4, 5, 6, 7, 8, 9]
```

8 Filtrar estudiantes con calificaciones mayores a 6

Filtra los estudiantes que tienen calificaciones mayores a 6 en todas las asignaturas. Luego, modifica la matriz reemplazando todas las calificaciones menores o iguales a 6 por 10. Funciones a utilizar: `np.all()` para filtrar por filas. `np.where` para reemplazar los valores.

8.1 Estudiantes con calificaciones mayores a 6 en todas las asignaturas

[166]:
```
filtro_mayores_6 = np.all(calificaciones > 6, axis=1)
"""
Filtra y encuentra estudiantes con todas las calificaciones mayores a 6.

Este fragmento de código utiliza un filtro para identificar a los estudiantes
cuyas calificaciones son todas mayores a 6. Luego, imprime los índices de
estos estudiantes.

Variables:
    filtro_mayores_6 (numpy.ndarray): Array booleano que indica si todas las
                                      calificaciones de cada estudiante son
                                      mayores a 6.
    estudiantes_mayores_6 (numpy.ndarray): Array de índices de los estudiantes
                                           que cumplen con la condición del
                                           filtro.

Salida:
    Imprime los índices de los estudiantes con todas las calificaciones
    mayores a 6.
"""
```

```
estudiantes_mayores_6 = np.where(filtro_mayores_6)[0] + 1
print("\nEstudiantes con todas las calificaciones mayores a 6:",␣
 ↪estudiantes_mayores_6)
```

```
Estudiantes con todas las calificaciones mayores a 6: []
```

8.2 Reemplazo de calificaciones menores o iguales a 6 con 10

[167]:
```
calificaciones_modificadas = np.where(calificaciones <= 6, 10, calificaciones)
"""
Reemplaza las calificaciones menores o iguales a 6 con 10 en una matriz de␣
 ↪calificaciones y la imprime.

Utiliza la función np.where para identificar las calificaciones que son menores␣
 ↪o iguales a 6 y las reemplaza con 10.
Luego, imprime la matriz modificada con un formato específico para los enteros.

Variables:
    calificaciones (numpy.ndarray): Matriz original de calificaciones.
    calificaciones_modificadas (numpy.ndarray): Matriz de calificaciones con␣
 ↪valores menores o iguales a 6 reemplazados por 10.

Salida:
    Imprime la matriz modificada con calificaciones menores o iguales a 6␣
 ↪reemplazadas por 10.
"""
print("\nMatriz con calificaciones menores o iguales a 6 reemplazadas por 10:")
print(np.array2string(calificaciones_modificadas, formatter={'int': '{:2d}'.
 ↪format}), "\n")
```

```
Matriz con calificaciones menores o iguales a 6 reemplazadas por 10:
[[ 7  8 10 10]
 [ 9  7  8 10]
 [10 10  9  7]
 [ 8 10  7  8]
 [10 10 10 10]]
```

9 Calcular estadísticas por asignatura

Calcular las siguientes estadísticas por cada asignatura:
Máximo: La calificación más alta.
Mínimo: La calificación más baja.
Mediana: El valor central de las calificaciones.
Funciones a utilizar:

```
np.max()
np.min()
np.median()
```

9.1 Calcular valores máximos

[168]:
```
max_por_asignatura = np.max(calificaciones, axis=0)
print("Valores máximos por asignatura:")
formatear_salida(max_por_asignatura, "Asignatura")
```

```
Valores máximos por asignatura:
 Asignatura 1: 9
 Asignatura 2: 8
 Asignatura 3: 9
 Asignatura 4: 8
```

9.2 Calcular valores mínimos

[169]:
```
min_por_asignatura = np.min(calificaciones, axis=0)
print("\nValores mínimos por asignatura:")
formatear_salida(min_por_asignatura, "Asignatura")
```

```
Valores mínimos por asignatura:
 Asignatura 1: 5
 Asignatura 2: 4
 Asignatura 3: 6
 Asignatura 4: 5
```

9.3 Calcular medianas por asignatura

[170]:
```
mediana_por_asignatura = np.median(calificaciones, axis=0)
print("\nMedianas por asignatura:")
formatear_salida(mediana_por_asignatura, "Asignatura")
```

```
Medianas por asignatura:
 Asignatura 1: 7.0
 Asignatura 2: 6.0
 Asignatura 3: 7.0
 Asignatura 4: 6.0
```

10 Crear un resumen completo

Imprime un resumen que contenga: La media, máximo, mínimo, mediana, y los valores únicos ordenados de cada asignatura.
El resumen debe verse de la siguiente forma:

Asignatura 1 (Matemáticas):
Media: ...
Máximo: ...
Mínimo: ...
Mediana: ...
Valores únicos: ...

[171]:
```
print("Resumen de estadísticas por asignatura:")
"""
Imprime un resumen de estadísticas por asignatura.

Este fragmento de código recorre las estadísticas de varias asignaturas y las␣
 ↪imprime en un formato legible.
Las estadísticas incluyen la media, el valor máximo, el valor mínimo, la␣
 ↪mediana y los valores únicos de las calificaciones.

Variables:
- media_asignatura: Lista de medias de calificaciones por asignatura.
- max_por_asignatura: Lista de valores máximos de calificaciones por asignatura.
- min_por_asignatura: Lista de valores mínimos de calificaciones por asignatura.
- mediana_por_asignatura: Lista de medianas de calificaciones por asignatura.
- calificaciones: Matriz de calificaciones de los estudiantes.

Nota:
    La función zip se utiliza para agrupar las listas de
    estadísticas (media, máximo, mínimo, mediana y valores únicos)
    por asignatura. Esto permite iterar sobre todas las estadísticas
    de una asignatura en cada iteración del bucle for, facilitando
    la impresión de los resultados.
"""
for i, (media, max_val, min_val, mediana, unicos) in enumerate(
    zip(media_asignatura, max_por_asignatura, min_por_asignatura,␣
 ↪mediana_por_asignatura, np.unique(calificaciones, axis=0).T), start=1
):
    print(f"\n Asignatura {i}:")
    print(f"   Media: {media:.2f}")
    print(f"   Máximo: {max_val}")
    print(f"   Mínimo: {min_val}")
    print(f"   Mediana: {mediana}")
    print(f"   Valores únicos: {sorted(map(int, unicos))}")
```

```
Resumen de estadísticas por asignatura:

 Asignatura 1:
   Media: 7.00
   Máximo: 9
   Mínimo: 5
   Mediana: 7.0
```

```
  Valores únicos: [5, 6, 7, 8, 9]

Asignatura 2:
  Media: 6.00
  Máximo: 8
  Mínimo: 4
  Mediana: 6.0
  Valores únicos: [4, 5, 6, 7, 8]

Asignatura 3:
  Media: 7.20
  Máximo: 9
  Mínimo: 6
  Mediana: 7.0
  Valores únicos: [6, 6, 7, 8, 9]

Asignatura 4:
  Media: 6.20
  Máximo: 8
  Mínimo: 5
  Mediana: 6.0
  Valores únicos: [5, 5, 6, 7, 8]
```