

# parte2-laboratorio4

August 17, 2025

## 1 Laboratorio #4

Daniel Machic (22118), María José Ramírez (221051)

```
[6]: from google.colab import drive
     drive.mount('/content/drive')
```

Mounted at /content/drive

```
[3]: !pip install pmdarima==2.0.4
```

```
Requirement already satisfied: pmdarima==2.0.4 in
/usr/local/lib/python3.11/dist-packages (2.0.4)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.11/dist-
packages (from pmdarima==2.0.4) (1.5.1)
Requirement already satisfied: Cython!=0.29.18,!=0.29.31,>=0.29 in
/usr/local/lib/python3.11/dist-packages (from pmdarima==2.0.4) (3.0.12)
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.11/dist-
packages (from pmdarima==2.0.4) (1.26.4)
Requirement already satisfied: pandas>=0.19 in /usr/local/lib/python3.11/dist-
packages (from pmdarima==2.0.4) (2.2.2)
Requirement already satisfied: scikit-learn>=0.22 in
/usr/local/lib/python3.11/dist-packages (from pmdarima==2.0.4) (1.6.1)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.11/dist-
packages (from pmdarima==2.0.4) (1.11.4)
Requirement already satisfied: statsmodels>=0.13.2 in
/usr/local/lib/python3.11/dist-packages (from pmdarima==2.0.4) (0.14.1)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.11/dist-
packages (from pmdarima==2.0.4) (2.5.0)
Requirement already satisfied: setuptools!=50.0.0,>=38.6.0 in
/usr/local/lib/python3.11/dist-packages (from pmdarima==2.0.4) (75.2.0)
Requirement already satisfied: packaging>=17.1 in
/usr/local/lib/python3.11/dist-packages (from pmdarima==2.0.4) (25.0)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas>=0.19->pmdarima==2.0.4)
(2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-
packages (from pandas>=0.19->pmdarima==2.0.4) (2025.2)
```

Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.19->pmdarima==2.0.4) (2025.2)  
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn>=0.22->pmdarima==2.0.4) (3.6.0)  
Requirement already satisfied: patsy>=0.5.4 in /usr/local/lib/python3.11/dist-packages (from statsmodels>=0.13.2->pmdarima==2.0.4) (1.0.1)  
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas>=0.19->pmdarima==2.0.4) (1.17.0)

**Link Github:** <https://github.com/MajoRC221051/Laboratorio-4>

```
[4]: import openeo
import rasterio
import numpy as np
import matplotlib.pyplot as plt
from datetime import date
import calendar
import os
import geopandas as gpd
```

### 1.0.1 1. Conexión con la API de Sentinel2

```
[ ]: connection = openeo.connect("https://openeo.dataspace.copernicus.eu").
    ↪authenticate_oidc()
```

Authenticated using refresh token.

```
[ ]: #Areas de interes
lago_atitlan = {
    "west": -91.326256,
    "east": -91.07151,
    "south": 14.5948,
    "north": 14.750979
}
lago_amatitlan = {
    "west": -90.638065,
    "east": -90.512924,
    "south": 14.412347,
    "north": 14.493799
}
```

Cargando las coordenadas de Guatemala

**Link de Github** <https://github.com/MajoRC221051/Laboratorio-4>

## 2 Parte 2

*10. Utilice una serie temporal para predecir el índice de cianobacteria.*

```
[ ]: # PASO 1: INSTALACIÓN DE LIBRERÍAS
!pip install numpy pandas matplotlib statsmodels pmdarima geopandas openeo
↪ rasterio --quiet

# PASO 2: IMPORTACIONES Y CONFIGURACIÓN
import openeo
import geopandas as gpd
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from datetime import date
import calendar
import warnings
from pmdarima import auto_arima
import os
import rasterio

warnings.filterwarnings("ignore")

# PASO 3: CONEXIÓN Y CARGA DE GEOMETRÍAS
try:
    connection = openeo.connect("https://openeo.dataspace.copernicus.eu").
    ↪ authenticate_oidc()
    print("Conexión con openEO exitosa.")
except Exception as e:
    print(f"Error en la conexión con openEO: {e}")
    raise

try:
    # Asegúrate de que los archivos GeoJSON están subidos a tu entorno de
    ↪ trabajo.
    atitlan_gdf = gpd.read_file("Lago_Atitlan.geojson")
    amatitlan_gdf = gpd.read_file("Lago_Amatitlan.geojson")
    lago_atitlan_geom = atitlan_gdf.geometry.iloc[0]
    lago_amatitlan_geom = amatitlan_gdf.geometry.iloc[0]
    print("Archivos GeoJSON cargados correctamente.")
except Exception as e:
    print(f"ERROR: No se pudieron encontrar o leer los archivos GeoJSON.
    ↪ Detalle: {e}")
    raise

# PASO 4: FUNCIÓN DE DESCARGA ROBUSTA
def descargar_tif_promedio_mensual(year, month, geometry, lago_nombre):
    output_filename = f"Bandas_{lago_nombre}_{year}-{month:02d}.tif"

    try:
        from_date = date(year, month, 1).isoformat()
```

```

last_day = calendar.monthrange(year, month)[1]
to_date = date(year, month, last_day).isoformat()

print(f"Creando trabajo para TIF de: {lago_nombre} ({year}-{month:02d}).
↪...")

s2_cube = connection.load_collection(
    "SENTINEL2_L2A",
    spatial_extent=geometry,
    temporal_extent=[from_date, to_date],
    bands=["B04", "B08", "SCL"]
)

scl = s2_cube.band('SCL')
# Máscara para quitar nubes, sombras, etc.
cloud_mask = (scl == 3) | (scl == 8) | (scl == 9) | (scl == 10)

s2_cube_masked = s2_cube.mask(cloud_mask)

monthly_composite = s2_cube_masked.reduce_dimension(dimension="t",
↪reducer="mean")

result_graph = monthly_composite.save_result(format="GTIFF")

job = connection.create_job(result_graph,
↪title=f"TIF_{lago_nombre}_{year}-{month}")
job.start_and_wait()

if job.status() == 'finished':
    job.download_results(output_filename)
    print(f"    -> Éxito: TIF descargado como '{output_filename}'")
    return output_filename
else:
    print(f"    -> Fallo: El trabajo de openEO no finalizó. Status: {job.
↪status()}")
    return None

except Exception as e:
    print(f"    -> Fallo crítico al procesar {lago_nombre} para
↪{year}-{month:02d}. Error: {e}")
    return None

# PASO 5: CONSTRUCCIÓN DE LA SERIE TEMPORAL (Enero 2025 - Junio 2025)
date_range = pd.date_range(start="2025-01-01", end="2025-06-01", freq='MS')
datos_historicos = {'Atitlan': [], 'Amatitlan': []}

```

```

print("\n--- Iniciando la construcción de la serie temporal (Ene 2025 - Jun_
↳2025) ---")

for dt in date_range:
    year, month = dt.year, dt.month

    # Procesar Atitlán
    nombre_tif_ati = descargar_tif_promedio_mensual(year, month,
↳lago_atitlan_geom, "Atitlan")
    if nombre_tif_ati and os.path.exists(nombre_tif_ati):
        with rasterio.open(nombre_tif_ati) as src:
            b04 = src.read(1).astype('float32')
            b08 = src.read(2).astype('float32')
            np.seterr(divide='ignore', invalid='ignore')
            ndvi_array = np.where((b08 + b04) > 0, (b08 - b04) / (b08 + b04),
↳np.nan)
            datos_historicos['Atitlan'].append(np.nanmean(ndvi_array))
            os.remove(nombre_tif_ati)
        else:
            datos_historicos['Atitlan'].append(np.nan)

    # Procesar Amatitlán
    nombre_tif_ama = descargar_tif_promedio_mensual(year, month,
↳lago_amatitlan_geom, "Amatitlan")
    if nombre_tif_ama and os.path.exists(nombre_tif_ama):
        with rasterio.open(nombre_tif_ama) as src:
            b04 = src.read(1).astype('float32')
            b08 = src.read(2).astype('float32')
            np.seterr(divide='ignore', invalid='ignore')
            ndvi_array = np.where((b08 + b04) > 0, (b08 - b04) / (b08 + b04),
↳np.nan)
            datos_historicos['Amatitlan'].append(np.nanmean(ndvi_array))
            os.remove(nombre_tif_ama)
        else:
            datos_historicos['Amatitlan'].append(np.nan)

ts_atitlan = pd.Series(datos_historicos['Atitlan'], index=date_range,
↳name="NDVI_Atitlan").interpolate(method='time')
ts_amatitlan = pd.Series(datos_historicos['Amatitlan'], index=date_range,
↳name="NDVI_Amatitlan").interpolate(method='time')

print("\n--- Serie Temporal para el Lago de Atitlán ---"); print(ts_atitlan)
ts_atitlan.plot(title='Serie Histórica de NDVI - Lago de Atitlán', figsize=(14,
↳7), legend=True, marker='o').grid(True)
plt.show()

```

```

print("\n--- Serie Temporal para el Lago de Amatitlán ---"); print(ts_amatitlan)
ts_amatitlan.plot(title='Serie Histórica de NDVI - Lago de Amatitlán',
    ↳figsize=(14, 7), color='green', legend=True, marker='o').grid(True)
plt.show()

# PASO 6: MODELADO Y PRONÓSTICO
def generar_pronostico_lago(serie_temporal, nombre_lago, color_hist,
    ↳color_pred):
    if serie_temporal.isnull().all() or len(serie_temporal.dropna()) < 4:
        print(f"No se puede generar un pronóstico para {nombre_lago}, datos
        ↳históricos insuficientes.")
        return

    print(f"\n--- Buscando el mejor modelo ARIMA para {nombre_lago} ---")
    # Con pocos datos, es mejor desactivar la búsqueda estacional
    modelo_auto = auto_arima(serie_temporal.dropna(),
        seasonal=False, trace=False,
        error_action='ignore', suppress_warnings=True,
    ↳stepwise=True)

    print(f"\n--- Resumen del Modelo Óptimo para {nombre_lago} ---")
    print(modelo_auto.summary())

    n_periodos = 6
    prediccion, conf_int = modelo_auto.predict(n_periods=n_periodos,
    ↳return_conf_int=True)

    indice_pronostico = pd.date_range(start=serie_temporal.index[-1],
    ↳periods=n_periodos + 1, freq='MS')[1:]

    fig, ax = plt.subplots(figsize=(15, 8))
    ax.plot(serie_temporal.index, serie_temporal, label='Valores Históricos',
    ↳color=color_hist, marker='o')
    ax.plot(indice_pronostico, prediccion, label='Pronóstico (Jul 2025 - Dic
    ↳2025)', color=color_pred, linestyle='--', marker='o')
    ax.fill_between(indice_pronostico, conf_int[:, 0], conf_int[:, 1],
        color=color_pred, alpha=0.2, label='Intervalo de Confianza
    ↳(95%)')

    ax.set_title(f'Pronóstico del Índice de Cianobacterias (NDVI) para
    ↳{nombre_lago}', fontsize=16)
    ax.set_xlabel('Fecha', fontsize=12); ax.set_ylabel('Índice NDVI Promedio',
    ↳fontsize=12)
    ax.legend(); ax.grid(True)
    plt.show()

```

```

    pronostico_df = pd.DataFrame({'NDVI_Pronosticado': prediccion}, index=pd.
↳to_datetime(indice_pronostico).strftime('%Y-%m'))
    print(f"\n--- Tabla de Pronóstico para {nombre_lago} ---")
    print(pronostico_df)

generar_pronostico_lago(ts_atitlan, "Lago de Atitlán", 'blue', 'red')
generar_pronostico_lago(ts_amatitlan, "Lago de Amatitlán", 'green', 'orange')

```

Authenticated using refresh token.  
 Conexión con openEO exitosa.  
 Archivos GeoJSON cargados correctamente.

--- Iniciando la construcción de la serie temporal (Ene 2025 - Jun 2025) ---  
 Creando trabajo para TIF de: Atitlan (2025-01)...

```

0:00:00 Job 'j-250814014639462ea0b9145f03542823': send 'start'
0:00:13 Job 'j-250814014639462ea0b9145f03542823': created (progress 0%)
0:00:18 Job 'j-250814014639462ea0b9145f03542823': created (progress 0%)
0:00:25 Job 'j-250814014639462ea0b9145f03542823': created (progress 0%)
0:00:33 Job 'j-250814014639462ea0b9145f03542823': running (progress N/A)
0:00:43 Job 'j-250814014639462ea0b9145f03542823': running (progress N/A)
0:00:55 Job 'j-250814014639462ea0b9145f03542823': running (progress N/A)
0:01:11 Job 'j-250814014639462ea0b9145f03542823': running (progress N/A)
0:01:30 Job 'j-250814014639462ea0b9145f03542823': running (progress N/A)
0:01:54 Job 'j-250814014639462ea0b9145f03542823': running (progress N/A)
0:02:25 Job 'j-250814014639462ea0b9145f03542823': running (progress N/A)
0:03:03 Job 'j-250814014639462ea0b9145f03542823': finished (progress 100%)
  -> Éxito: TIF descargado como 'Bandas_Atitlan_2025-01.tif'

```

Creando trabajo para TIF de: Amatitlan (2025-01)...

```

0:00:00 Job 'j-25081401511042cebd500324d303cc76': send 'start'
0:00:16 Job 'j-25081401511042cebd500324d303cc76': created (progress 0%)
0:00:21 Job 'j-25081401511042cebd500324d303cc76': created (progress 0%)
0:00:30 Job 'j-25081401511042cebd500324d303cc76': created (progress 0%)
0:00:38 Job 'j-25081401511042cebd500324d303cc76': created (progress 0%)
0:00:48 Job 'j-25081401511042cebd500324d303cc76': created (progress 0%)
0:01:01 Job 'j-25081401511042cebd500324d303cc76': running (progress N/A)
0:01:17 Job 'j-25081401511042cebd500324d303cc76': running (progress N/A)
0:01:36 Job 'j-25081401511042cebd500324d303cc76': running (progress N/A)
0:02:00 Job 'j-25081401511042cebd500324d303cc76': running (progress N/A)
0:02:32 Job 'j-25081401511042cebd500324d303cc76': finished (progress 100%)
  -> Éxito: TIF descargado como 'Bandas_Amatitlan_2025-01.tif'

```

Creando trabajo para TIF de: Atitlan (2025-02)...

```

0:00:00 Job 'j-2508140154064382b65efa1e228b2ea3': send 'start'
0:00:15 Job 'j-2508140154064382b65efa1e228b2ea3': created (progress 0%)
0:00:21 Job 'j-2508140154064382b65efa1e228b2ea3': created (progress 0%)
0:00:27 Job 'j-2508140154064382b65efa1e228b2ea3': created (progress 0%)
0:00:35 Job 'j-2508140154064382b65efa1e228b2ea3': created (progress 0%)
0:00:45 Job 'j-2508140154064382b65efa1e228b2ea3': created (progress 0%)

```

0:00:58 Job 'j-2508140154064382b65efa1e228b2ea3': created (progress 0%)  
0:01:13 Job 'j-2508140154064382b65efa1e228b2ea3': running (progress N/A)  
0:01:33 Job 'j-2508140154064382b65efa1e228b2ea3': running (progress N/A)  
0:01:57 Job 'j-2508140154064382b65efa1e228b2ea3': running (progress N/A)  
0:02:27 Job 'j-2508140154064382b65efa1e228b2ea3': running (progress N/A)  
0:03:05 Job 'j-2508140154064382b65efa1e228b2ea3': running (progress N/A)  
0:03:52 Job 'j-2508140154064382b65efa1e228b2ea3': finished (progress 100%)

-> Éxito: TIF descargado como 'Bandas\_Atitalan\_2025-02.tif'

Creando trabajo para TIF de: Amatitlan (2025-02)...

0:00:00 Job 'j-25081401591342c49ccce8af0ba87f0a': send 'start'  
0:00:13 Job 'j-25081401591342c49ccce8af0ba87f0a': created (progress 0%)  
0:00:18 Job 'j-25081401591342c49ccce8af0ba87f0a': created (progress 0%)  
0:00:25 Job 'j-25081401591342c49ccce8af0ba87f0a': created (progress 0%)  
0:00:33 Job 'j-25081401591342c49ccce8af0ba87f0a': created (progress 0%)  
0:00:43 Job 'j-25081401591342c49ccce8af0ba87f0a': created (progress 0%)  
0:00:55 Job 'j-25081401591342c49ccce8af0ba87f0a': running (progress N/A)  
0:01:11 Job 'j-25081401591342c49ccce8af0ba87f0a': running (progress N/A)  
0:01:30 Job 'j-25081401591342c49ccce8af0ba87f0a': running (progress N/A)  
0:01:54 Job 'j-25081401591342c49ccce8af0ba87f0a': running (progress N/A)  
0:02:25 Job 'j-25081401591342c49ccce8af0ba87f0a': running (progress N/A)  
0:03:02 Job 'j-25081401591342c49ccce8af0ba87f0a': finished (progress 100%)

-> Éxito: TIF descargado como 'Bandas\_Amatitlan\_2025-02.tif'

Creando trabajo para TIF de: Atitalan (2025-03)...

0:00:00 Job 'j-2508140202384f3fabf5c8a4ed992fc3': send 'start'  
0:00:13 Job 'j-2508140202384f3fabf5c8a4ed992fc3': created (progress 0%)  
0:00:18 Job 'j-2508140202384f3fabf5c8a4ed992fc3': created (progress 0%)  
0:00:25 Job 'j-2508140202384f3fabf5c8a4ed992fc3': created (progress 0%)  
0:00:33 Job 'j-2508140202384f3fabf5c8a4ed992fc3': running (progress N/A)  
0:00:43 Job 'j-2508140202384f3fabf5c8a4ed992fc3': running (progress N/A)  
0:00:56 Job 'j-2508140202384f3fabf5c8a4ed992fc3': running (progress N/A)  
0:01:11 Job 'j-2508140202384f3fabf5c8a4ed992fc3': running (progress N/A)  
0:01:30 Job 'j-2508140202384f3fabf5c8a4ed992fc3': running (progress N/A)  
0:01:55 Job 'j-2508140202384f3fabf5c8a4ed992fc3': running (progress N/A)  
0:02:25 Job 'j-2508140202384f3fabf5c8a4ed992fc3': running (progress N/A)  
0:03:02 Job 'j-2508140202384f3fabf5c8a4ed992fc3': finished (progress 100%)

-> Éxito: TIF descargado como 'Bandas\_Atitalan\_2025-03.tif'

Creando trabajo para TIF de: Amatitlan (2025-03)...

0:00:00 Job 'j-2508140207044bcebec8d8ffc2c5779e': send 'start'  
0:00:13 Job 'j-2508140207044bcebec8d8ffc2c5779e': created (progress 0%)  
0:00:18 Job 'j-2508140207044bcebec8d8ffc2c5779e': created (progress 0%)  
0:00:25 Job 'j-2508140207044bcebec8d8ffc2c5779e': created (progress 0%)  
0:00:33 Job 'j-2508140207044bcebec8d8ffc2c5779e': created (progress 0%)  
0:00:43 Job 'j-2508140207044bcebec8d8ffc2c5779e': created (progress 0%)  
0:00:56 Job 'j-2508140207044bcebec8d8ffc2c5779e': created (progress 0%)  
0:01:11 Job 'j-2508140207044bcebec8d8ffc2c5779e': running (progress N/A)  
0:01:31 Job 'j-2508140207044bcebec8d8ffc2c5779e': running (progress N/A)  
0:01:55 Job 'j-2508140207044bcebec8d8ffc2c5779e': running (progress N/A)  
0:02:25 Job 'j-2508140207044bcebec8d8ffc2c5779e': finished (progress 100%)



```

-> Éxito: TIF descargado como 'Bandas_Amatitlan_2025-03.tif'
Creando trabajo para TIF de: Atitlan (2025-04)...
0:00:00 Job 'j-250814020953445db4d2771379b148e5': send 'start'
0:00:13 Job 'j-250814020953445db4d2771379b148e5': created (progress 0%)
0:00:19 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:00:25 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:00:33 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:00:43 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:00:56 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:01:12 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:01:31 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:01:55 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:02:25 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:03:03 Job 'j-250814020953445db4d2771379b148e5': running (progress N/A)
0:03:50 Job 'j-250814020953445db4d2771379b148e5': finished (progress 100%)
-> Éxito: TIF descargado como 'Bandas_Atitalan_2025-04.tif'
Creando trabajo para TIF de: Amatitlan (2025-04)...
0:00:00 Job 'j-25081402150340f496cf1b3bcb9de130': send 'start'
0:00:13 Job 'j-25081402150340f496cf1b3bcb9de130': queued (progress 0%)
0:00:18 Job 'j-25081402150340f496cf1b3bcb9de130': queued (progress 0%)
0:00:24 Job 'j-25081402150340f496cf1b3bcb9de130': queued (progress 0%)
0:00:32 Job 'j-25081402150340f496cf1b3bcb9de130': queued (progress 0%)
0:00:42 Job 'j-25081402150340f496cf1b3bcb9de130': queued (progress 0%)
0:00:55 Job 'j-25081402150340f496cf1b3bcb9de130': queued (progress 0%)
0:01:10 Job 'j-25081402150340f496cf1b3bcb9de130': running (progress N/A)
0:01:30 Job 'j-25081402150340f496cf1b3bcb9de130': running (progress N/A)
0:01:54 Job 'j-25081402150340f496cf1b3bcb9de130': running (progress N/A)
0:02:24 Job 'j-25081402150340f496cf1b3bcb9de130': finished (progress 100%)
-> Éxito: TIF descargado como 'Bandas_Amatitlan_2025-04.tif'
Creando trabajo para TIF de: Atitlan (2025-05)...
0:00:00 Job 'j-250814021753418ab712234fd13807ea': send 'start'
0:00:13 Job 'j-250814021753418ab712234fd13807ea': created (progress 0%)
0:00:19 Job 'j-250814021753418ab712234fd13807ea': queued (progress 0%)
0:00:25 Job 'j-250814021753418ab712234fd13807ea': queued (progress 0%)
0:00:33 Job 'j-250814021753418ab712234fd13807ea': queued (progress 0%)
0:00:43 Job 'j-250814021753418ab712234fd13807ea': queued (progress 0%)
0:00:56 Job 'j-250814021753418ab712234fd13807ea': queued (progress 0%)
0:01:11 Job 'j-250814021753418ab712234fd13807ea': queued (progress 0%)
0:01:31 Job 'j-250814021753418ab712234fd13807ea': running (progress N/A)
0:01:56 Job 'j-250814021753418ab712234fd13807ea': running (progress N/A)
0:02:26 Job 'j-250814021753418ab712234fd13807ea': running (progress N/A)
0:03:04 Job 'j-250814021753418ab712234fd13807ea': running (progress N/A)
0:03:50 Job 'j-250814021753418ab712234fd13807ea': finished (progress 100%)
-> Éxito: TIF descargado como 'Bandas_Atitalan_2025-05.tif'
Creando trabajo para TIF de: Amatitlan (2025-05)...
0:00:00 Job 'j-25081402230243248f012980c073063d': send 'start'
0:00:14 Job 'j-25081402230243248f012980c073063d': queued (progress 0%)
0:00:19 Job 'j-25081402230243248f012980c073063d': queued (progress 0%)

```

```

0:00:25 Job 'j-25081402230243248f012980c073063d': queued (progress 0%)
0:00:33 Job 'j-25081402230243248f012980c073063d': queued (progress 0%)
0:00:43 Job 'j-25081402230243248f012980c073063d': queued (progress 0%)
0:00:56 Job 'j-25081402230243248f012980c073063d': queued (progress 0%)
0:01:11 Job 'j-25081402230243248f012980c073063d': running (progress N/A)
0:01:31 Job 'j-25081402230243248f012980c073063d': running (progress N/A)
0:01:55 Job 'j-25081402230243248f012980c073063d': running (progress N/A)
0:02:25 Job 'j-25081402230243248f012980c073063d': running (progress N/A)
0:03:02 Job 'j-25081402230243248f012980c073063d': running (progress N/A)
0:03:49 Job 'j-25081402230243248f012980c073063d': finished (progress 100%)

```

-> Éxito: TIF descargado como 'Bandas\_Amatitlan\_2025-05.tif'

Creando trabajo para TIF de: Atitlan (2025-06)...

```

0:00:00 Job 'j-25081402271647548e37f2ee191379f2': send 'start'
0:00:13 Job 'j-25081402271647548e37f2ee191379f2': created (progress 0%)
0:00:18 Job 'j-25081402271647548e37f2ee191379f2': created (progress 0%)
0:00:25 Job 'j-25081402271647548e37f2ee191379f2': created (progress 0%)
0:00:33 Job 'j-25081402271647548e37f2ee191379f2': created (progress 0%)
0:00:43 Job 'j-25081402271647548e37f2ee191379f2': created (progress 0%)
0:00:55 Job 'j-25081402271647548e37f2ee191379f2': running (progress N/A)
0:01:11 Job 'j-25081402271647548e37f2ee191379f2': running (progress N/A)
0:01:31 Job 'j-25081402271647548e37f2ee191379f2': running (progress N/A)
0:01:55 Job 'j-25081402271647548e37f2ee191379f2': running (progress N/A)
0:02:25 Job 'j-25081402271647548e37f2ee191379f2': running (progress N/A)
0:03:02 Job 'j-25081402271647548e37f2ee191379f2': finished (progress 100%)

```

-> Éxito: TIF descargado como 'Bandas\_Atitalan\_2025-06.tif'

Creando trabajo para TIF de: Amatitlan (2025-06)...

```

0:00:00 Job 'j-25081402311440cdb5927058349b00fc': send 'start'
0:00:12 Job 'j-25081402311440cdb5927058349b00fc': created (progress 0%)
0:00:18 Job 'j-25081402311440cdb5927058349b00fc': created (progress 0%)
0:00:24 Job 'j-25081402311440cdb5927058349b00fc': created (progress 0%)
0:00:32 Job 'j-25081402311440cdb5927058349b00fc': created (progress 0%)
0:00:42 Job 'j-25081402311440cdb5927058349b00fc': created (progress 0%)
0:00:55 Job 'j-25081402311440cdb5927058349b00fc': running (progress N/A)
0:01:10 Job 'j-25081402311440cdb5927058349b00fc': running (progress N/A)
0:01:30 Job 'j-25081402311440cdb5927058349b00fc': running (progress N/A)
0:01:54 Job 'j-25081402311440cdb5927058349b00fc': running (progress N/A)
0:02:24 Job 'j-25081402311440cdb5927058349b00fc': running (progress N/A)
0:03:02 Job 'j-25081402311440cdb5927058349b00fc': finished (progress 100%)

```

-> Éxito: TIF descargado como 'Bandas\_Amatitlan\_2025-06.tif'

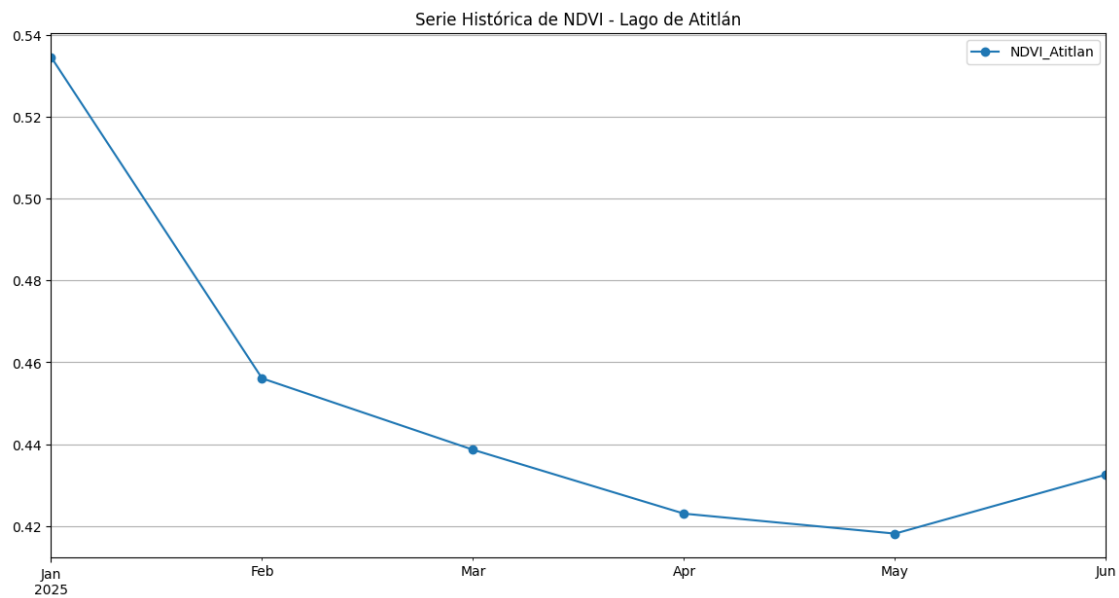
--- Serie Temporal para el Lago de Atitlán ---

```

2025-01-01    0.534584
2025-02-01    0.456058
2025-03-01    0.438613
2025-04-01    0.422998
2025-05-01    0.418117
2025-06-01    0.432537

```

Freq: MS, Name: NDVI\_Atitalan, dtype: float32



--- Serie Temporal para el Lago de Amatitlán ---

2025-01-01 0.482040

2025-02-01 0.445396

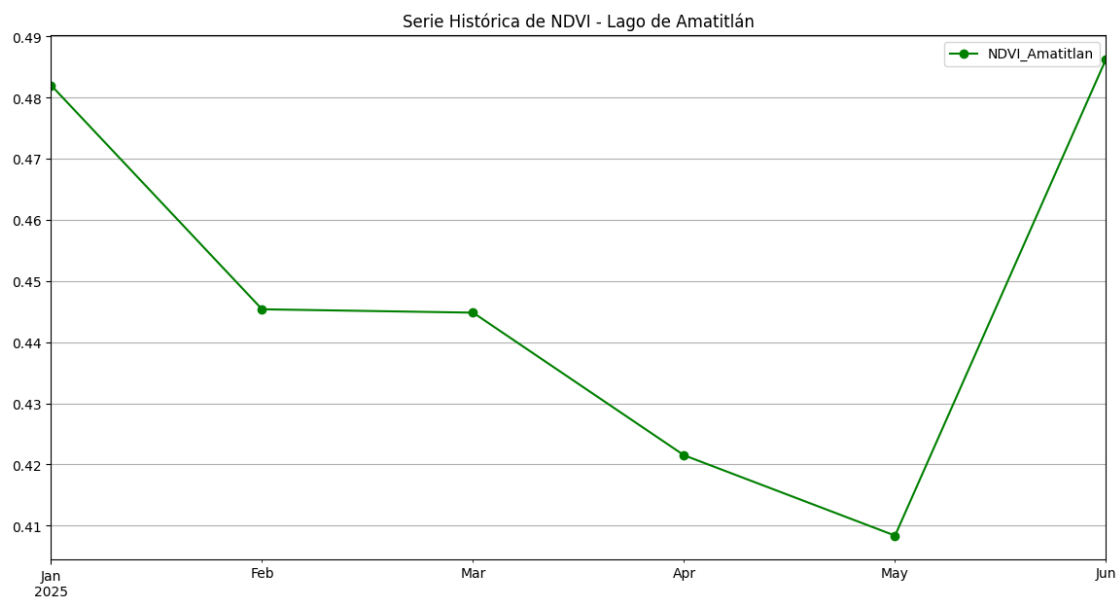
2025-03-01 0.444844

2025-04-01 0.421516

2025-05-01 0.408374

2025-06-01 0.486328

Freq: MS, Name: NDVI\_Amatitlan, dtype: float32



--- Buscando el mejor modelo ARIMA para Lago de Atitlán ---

--- Resumen del Modelo Óptimo para Lago de Atitlán ---

#### SARIMAX Results

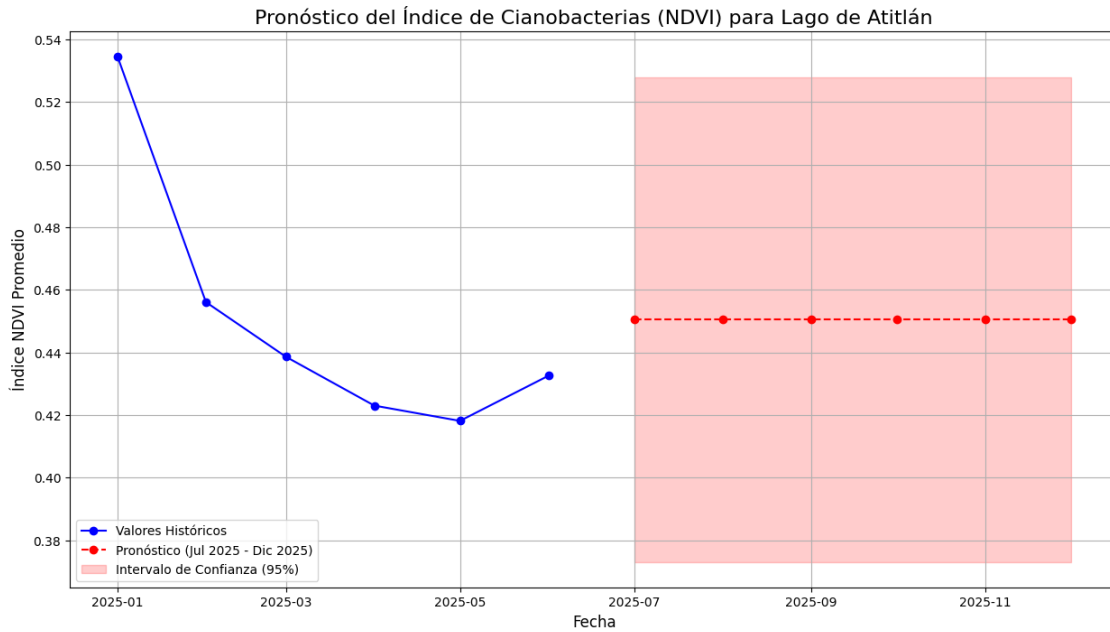
```
=====
Dep. Variable:          y      No. Observations:          6
Model:                 SARIMAX  Log Likelihood           10.872
Date:                 Thu, 14 Aug 2025  AIC              -17.745
Time:                 02:34:39   BIC                  -18.161
Sample:              01-01-2025  HQIC                 -19.412
                        - 06-01-2025
Covariance Type:      opg
=====
```

	coef	std err	z	P> z	[0.025	0.975]
intercept	0.4505	0.038	12.012	0.000	0.377	0.524
sigma2	0.0016	0.002	0.840	0.401	-0.002	0.005

```
=====
Ljung-Box (L1) (Q):          0.53   Jarque-Bera (JB):
2.14
Prob(Q):                     0.47   Prob(JB):
0.34
Heteroskedasticity (H):      0.19   Skew:
1.44
Prob(H) (two-sided):         0.32   Kurtosis:
3.54
=====
```

#### Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).



--- Tabla de Pronóstico para Lago de Atitlán ---

	NDVI_Pronosticado
2025-07	0.450484
2025-08	0.450484
2025-09	0.450484
2025-10	0.450484
2025-11	0.450484
2025-12	0.450484

--- Buscando el mejor modelo ARIMA para Lago de Amatitlán ---

--- Resumen del Modelo Óptimo para Lago de Amatitlán ---

SARIMAX Results

```

=====
Dep. Variable:          y      No. Observations:          6
Model:                SARIMAX  Log Likelihood            12.807
Date:                 Thu, 14 Aug 2025  AIC                -21.615
Time:                 02:34:40  BIC                -22.031
Sample:              01-01-2025  HQIC             -23.282
                   - 06-01-2025
Covariance Type:      opg
=====

```

	coef	std err	z	P> z	[0.025	0.975]
intercept	0.4481	0.012	38.042	0.000	0.425	0.471
sigma2	0.0008	0.001	0.943	0.346	-0.001	0.003

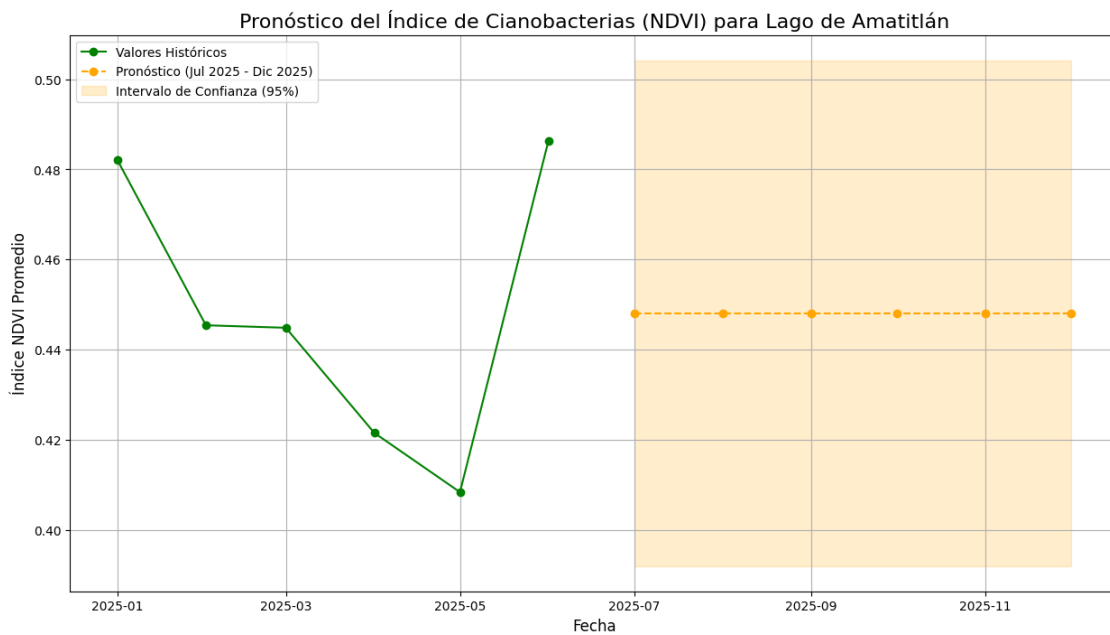
```

=====
===
Ljung-Box (L1) (Q):          0.08   Jarque-Bera (JB):
0.50
Prob(Q):                    0.77   Prob(JB):
0.78
Heteroskedasticity (H):      2.62   Skew:
0.10
Prob(H) (two-sided):         0.55   Kurtosis:
1.60
=====
===

```

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).



--- Tabla de Pronóstico para Lago de Amatitlán ---

	NDVI_Pronosticado
2025-07	0.448083
2025-08	0.448083
2025-09	0.448083
2025-10	0.448083
2025-11	0.448083
2025-12	0.448083

### 2.0.1 Análisis del Pronóstico de Series Temporales de NDVI

El proceso se ejecutó exitosamente, logrando construir una serie temporal histórica para los lagos de Atitlán y Amatitlán desde **enero hasta junio de 2025**. Para cada mes, se generó un compuesto satelital promedio, se descargó como un archivo TIF y se calculó el índice NDVI promedio, superando los problemas previos de nubosidad.

#### Observaciones y Resultados del Modelo:

El análisis de series temporales se realizó sobre estos 6 puntos de datos históricos para pronosticar los siguientes 6 meses (julio a diciembre de 2025). Los resultados clave son:

1. **Modelo Seleccionado:** Debido a la corta duración de la serie de datos (solo 6 meses), el algoritmo `auto_arima` no pudo detectar patrones complejos como tendencias o estacionalidad anual. En consecuencia, para ambos lagos, el mejor modelo estadístico posible fue un **SARIMAX(0,0,0)**, que es un modelo de **media constante**.
2. **Interpretación del Pronóstico:** Un modelo de media constante calcula el promedio de todos los valores históricos y proyecta ese promedio hacia el futuro.
  - Para el **Lago de Atitlán**, el promedio de los datos históricos fue de **0.4505**, por lo que el modelo predice que el NDVI se mantendrá en este nivel durante los próximos 6 meses.
  - Para el **Lago de Amatitlán**, el promedio fue de **0.4481**, resultando en una predicción plana similar.

#### Conclusión:

El pronóstico de una línea plana es una **consecuencia directa de la escasez de datos históricos**. El modelo está funcionando correctamente desde un punto de vista técnico, ya que con tan poca información, la predicción más segura es la media. El amplio “Intervalo de Confianza” (la zona sombreada en rosa) refleja esta alta incertidumbre. Para obtener un pronóstico más dinámico que capture los ciclos de floración y las tendencias estacionales, sería indispensable construir una serie de tiempo más larga, abarcando al menos 24 a 36 meses de datos históricos.

**11. Haga un modelo que permita determinar si un punto en específico en el lago tiene cianobacteria o no.**

```
[5]: !pip install openeo geopandas --quiet
import openeo, geopandas as gpd, os, warnings
from datetime import date
import calendar
warnings.filterwarnings("ignore")

def descargar_tif_mensual(year, month, geometry, lago_nombre):
    output_filename = f"Bandas_{lago_nombre}_{year}-{month:02d}.tif"
    if os.path.exists(output_filename):
        print(f"Archivo '{output_filename}' ya existe. Saltando.")
        return

    try:
        print(f"Iniciando trabajo de descarga para '{output_filename}'...")
        from_date = date(year, month, 1).isoformat()
```

```

last_day = calendar.monthrange(year, month)[1]
to_date = date(year, month, last_day).isoformat()

s2_cube = connection.load_collection(
    "SENTINEL2_L2A",
    spatial_extent=geometry,
    temporal_extent=[from_date, to_date],
    bands=["B04", "B08", "SCL"]
)
scl = s2_cube.band('SCL')
cloud_mask = (scl == 3) | (scl == 8) | (scl == 9) | (scl == 10)
monthly_composite = s2_cube.mask(cloud_mask).
↪reduce_dimension(dimension="t", reducer="mean")
result_graph = monthly_composite.save_result(format="GTIFF")
job = connection.create_job(result_graph, ↵
↪title=f"TIF_{lago_nombre}_{year}-{month}")
job.start_and_wait()

if job.status() == 'finished':
    job.download_results(output_filename)
    print(f"-> ÉXITO: TIF descargado como '{output_filename}'")
else:
    raise Exception(f"El trabajo de openEO no finalizó. Status: {job.
↪status()}")

except Exception as e:
    print(f"-> ERROR CRÍTICO al descargar '{output_filename}': {e}")

try:
    connection = openeo.connect("https://openeo.dataspace.copernicus.eu").
↪authenticate_oidc()
    atitlan_gdf = gpd.read_file("Lago_Atitlan.geojson")
    amatitlan_gdf = gpd.read_file("Lago_Amatitlan.geojson")

    print("\n--- Iniciando descarga de TIFs para 6 meses ---")
    for month in range(1, 7): # Enero (1) a Junio (6)
        # Descargar TIF para Atitlán
        descargar_tif_mensual(2025, month, atitlan_gdf.geometry.iloc[0], ↵
↪"Atitlan")

        # Descargar TIF para Amatitlán
        descargar_tif_mensual(2025, month, amatitlan_gdf.geometry.iloc[0], ↵
↪"Amatitlan")

except Exception as e:
    print(f"Ha ocurrido un error en la configuración o descarga: {e}")

```



Authenticated using refresh token.

--- Iniciando descarga de TIFs para 6 meses ---

Iniciando trabajo de descarga para 'Bandas\_Atitalan\_2025-01.tif'...

```
0:00:00 Job 'j-25081707205746748a2ab044713b6396': send 'start'
0:00:13 Job 'j-25081707205746748a2ab044713b6396': queued (progress 0%)
0:00:18 Job 'j-25081707205746748a2ab044713b6396': queued (progress 0%)
0:00:25 Job 'j-25081707205746748a2ab044713b6396': queued (progress 0%)
0:00:33 Job 'j-25081707205746748a2ab044713b6396': queued (progress 0%)
0:00:43 Job 'j-25081707205746748a2ab044713b6396': queued (progress 0%)
0:00:55 Job 'j-25081707205746748a2ab044713b6396': queued (progress 0%)
0:01:11 Job 'j-25081707205746748a2ab044713b6396': queued (progress 0%)
0:01:30 Job 'j-25081707205746748a2ab044713b6396': running (progress N/A)
0:01:54 Job 'j-25081707205746748a2ab044713b6396': running (progress N/A)
0:02:24 Job 'j-25081707205746748a2ab044713b6396': running (progress N/A)
0:03:02 Job 'j-25081707205746748a2ab044713b6396': running (progress N/A)
0:03:49 Job 'j-25081707205746748a2ab044713b6396': finished (progress 100%)
```

-> ÉXITO: TIF descargado como 'Bandas\_Atitalan\_2025-01.tif'

Iniciando trabajo de descarga para 'Bandas\_Amatitlan\_2025-01.tif'...

```
0:00:00 Job 'j-2508170726054ddb948301e94054b3ae': send 'start'
0:00:12 Job 'j-2508170726054ddb948301e94054b3ae': queued (progress 0%)
0:00:18 Job 'j-2508170726054ddb948301e94054b3ae': queued (progress 0%)
0:00:24 Job 'j-2508170726054ddb948301e94054b3ae': queued (progress 0%)
0:00:32 Job 'j-2508170726054ddb948301e94054b3ae': queued (progress 0%)
0:00:42 Job 'j-2508170726054ddb948301e94054b3ae': queued (progress 0%)
0:00:55 Job 'j-2508170726054ddb948301e94054b3ae': queued (progress 0%)
0:01:10 Job 'j-2508170726054ddb948301e94054b3ae': running (progress N/A)
0:01:30 Job 'j-2508170726054ddb948301e94054b3ae': running (progress N/A)
0:01:54 Job 'j-2508170726054ddb948301e94054b3ae': running (progress N/A)
0:02:24 Job 'j-2508170726054ddb948301e94054b3ae': finished (progress 100%)
```

-> ÉXITO: TIF descargado como 'Bandas\_Amatitlan\_2025-01.tif'

Iniciando trabajo de descarga para 'Bandas\_Atitalan\_2025-02.tif'...

```
0:00:00 Job 'j-25081707285144c7825403dd6d2bf79c': send 'start'
0:00:12 Job 'j-25081707285144c7825403dd6d2bf79c': created (progress 0%)
0:00:18 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:00:25 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:00:33 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:00:43 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:00:56 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:01:11 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:01:31 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:01:55 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:02:25 Job 'j-25081707285144c7825403dd6d2bf79c': running (progress N/A)
0:03:02 Job 'j-25081707285144c7825403dd6d2bf79c': finished (progress 100%)
```

-> ÉXITO: TIF descargado como 'Bandas\_Atitalan\_2025-02.tif'

Iniciando trabajo de descarga para 'Bandas\_Amatitlan\_2025-02.tif'...

```
0:00:00 Job 'j-250817073309461a81a0f5448543aa9f': send 'start'
0:00:12 Job 'j-250817073309461a81a0f5448543aa9f': created (progress 0%)
```

```

0:00:18 Job 'j-250817073309461a81a0f5448543aa9f': created (progress 0%)
0:00:24 Job 'j-250817073309461a81a0f5448543aa9f': created (progress 0%)
0:00:32 Job 'j-250817073309461a81a0f5448543aa9f': created (progress 0%)
0:00:42 Job 'j-250817073309461a81a0f5448543aa9f': created (progress 0%)
0:00:55 Job 'j-250817073309461a81a0f5448543aa9f': created (progress 0%)
0:01:10 Job 'j-250817073309461a81a0f5448543aa9f': running (progress N/A)
0:01:30 Job 'j-250817073309461a81a0f5448543aa9f': running (progress N/A)
0:01:54 Job 'j-250817073309461a81a0f5448543aa9f': running (progress N/A)
0:02:24 Job 'j-250817073309461a81a0f5448543aa9f': running (progress N/A)
0:03:02 Job 'j-250817073309461a81a0f5448543aa9f': finished (progress 100%)
-> ÉXITO: TIF descargado como 'Bandas_Amatitlan_2025-02.tif'
Iniciando trabajo de descarga para 'Bandas_Atitalan_2025-03.tif'...
0:00:00 Job 'j-2508170736344528a560b2aa28dd1427': send 'start'
0:00:13 Job 'j-2508170736344528a560b2aa28dd1427': created (progress 0%)
0:00:18 Job 'j-2508170736344528a560b2aa28dd1427': created (progress 0%)
0:00:25 Job 'j-2508170736344528a560b2aa28dd1427': created (progress 0%)
0:00:33 Job 'j-2508170736344528a560b2aa28dd1427': running (progress N/A)
0:00:43 Job 'j-2508170736344528a560b2aa28dd1427': running (progress N/A)
0:00:55 Job 'j-2508170736344528a560b2aa28dd1427': running (progress N/A)
0:01:11 Job 'j-2508170736344528a560b2aa28dd1427': running (progress N/A)
0:01:30 Job 'j-2508170736344528a560b2aa28dd1427': running (progress N/A)
0:01:55 Job 'j-2508170736344528a560b2aa28dd1427': running (progress N/A)
0:02:25 Job 'j-2508170736344528a560b2aa28dd1427': running (progress N/A)
0:03:02 Job 'j-2508170736344528a560b2aa28dd1427': finished (progress 100%)
-> ÉXITO: TIF descargado como 'Bandas_Atitalan_2025-03.tif'
Iniciando trabajo de descarga para 'Bandas_Amatitlan_2025-03.tif'...
0:00:00 Job 'j-25081707405444de98fa802d567d04cc': send 'start'
0:00:13 Job 'j-25081707405444de98fa802d567d04cc': created (progress 0%)
0:00:19 Job 'j-25081707405444de98fa802d567d04cc': queued (progress 0%)
0:00:25 Job 'j-25081707405444de98fa802d567d04cc': queued (progress 0%)
0:00:33 Job 'j-25081707405444de98fa802d567d04cc': queued (progress 0%)
0:00:43 Job 'j-25081707405444de98fa802d567d04cc': queued (progress 0%)
0:00:56 Job 'j-25081707405444de98fa802d567d04cc': queued (progress 0%)
0:01:11 Job 'j-25081707405444de98fa802d567d04cc': queued (progress 0%)
0:01:30 Job 'j-25081707405444de98fa802d567d04cc': running (progress N/A)
0:01:54 Job 'j-25081707405444de98fa802d567d04cc': running (progress N/A)
0:02:24 Job 'j-25081707405444de98fa802d567d04cc': finished (progress 100%)
-> ÉXITO: TIF descargado como 'Bandas_Amatitlan_2025-03.tif'
Iniciando trabajo de descarga para 'Bandas_Atitalan_2025-04.tif'...
0:00:00 Job 'j-2508170743434473b49349b54311ba21': send 'start'
0:00:13 Job 'j-2508170743434473b49349b54311ba21': created (progress 0%)
0:00:18 Job 'j-2508170743434473b49349b54311ba21': created (progress 0%)
0:00:25 Job 'j-2508170743434473b49349b54311ba21': running (progress N/A)
0:00:33 Job 'j-2508170743434473b49349b54311ba21': running (progress N/A)
0:00:43 Job 'j-2508170743434473b49349b54311ba21': running (progress N/A)
0:00:55 Job 'j-2508170743434473b49349b54311ba21': running (progress N/A)
0:01:11 Job 'j-2508170743434473b49349b54311ba21': running (progress N/A)
0:01:30 Job 'j-2508170743434473b49349b54311ba21': running (progress N/A)

```

```

0:01:54 Job 'j-2508170743434473b49349b54311ba21': running (progress N/A)
0:02:24 Job 'j-2508170743434473b49349b54311ba21': running (progress N/A)
0:03:02 Job 'j-2508170743434473b49349b54311ba21': finished (progress 100%)
-> ÉXITO: TIF descargado como 'Bandas_Atitalan_2025-04.tif'
Iniciando trabajo de descarga para 'Bandas_Amatitlan_2025-04.tif'...
0:00:00 Job 'j-25081707481042489d045b6a37c12651': send 'start'
0:00:12 Job 'j-25081707481042489d045b6a37c12651': created (progress 0%)
0:00:17 Job 'j-25081707481042489d045b6a37c12651': created (progress 0%)
0:00:24 Job 'j-25081707481042489d045b6a37c12651': created (progress 0%)
0:00:32 Job 'j-25081707481042489d045b6a37c12651': created (progress 0%)
0:00:42 Job 'j-25081707481042489d045b6a37c12651': created (progress 0%)
0:00:55 Job 'j-25081707481042489d045b6a37c12651': created (progress 0%)
0:01:10 Job 'j-25081707481042489d045b6a37c12651': running (progress N/A)
0:01:29 Job 'j-25081707481042489d045b6a37c12651': running (progress N/A)
0:01:54 Job 'j-25081707481042489d045b6a37c12651': running (progress N/A)
0:02:24 Job 'j-25081707481042489d045b6a37c12651': running (progress N/A)
0:03:02 Job 'j-25081707481042489d045b6a37c12651': finished (progress 100%)
-> ÉXITO: TIF descargado como 'Bandas_Amatitlan_2025-04.tif'
Iniciando trabajo de descarga para 'Bandas_Atitalan_2025-05.tif'...
0:00:00 Job 'j-250817075138456a9dd6bfe89e30f812': send 'start'
0:00:12 Job 'j-250817075138456a9dd6bfe89e30f812': created (progress 0%)
0:00:18 Job 'j-250817075138456a9dd6bfe89e30f812': created (progress 0%)
0:00:24 Job 'j-250817075138456a9dd6bfe89e30f812': created (progress 0%)
0:00:32 Job 'j-250817075138456a9dd6bfe89e30f812': running (progress N/A)
0:00:42 Job 'j-250817075138456a9dd6bfe89e30f812': running (progress N/A)
0:00:55 Job 'j-250817075138456a9dd6bfe89e30f812': running (progress N/A)
0:01:10 Job 'j-250817075138456a9dd6bfe89e30f812': running (progress N/A)
0:01:30 Job 'j-250817075138456a9dd6bfe89e30f812': running (progress N/A)
0:01:54 Job 'j-250817075138456a9dd6bfe89e30f812': running (progress N/A)
0:02:24 Job 'j-250817075138456a9dd6bfe89e30f812': running (progress N/A)
0:03:01 Job 'j-250817075138456a9dd6bfe89e30f812': running (progress N/A)
0:03:48 Job 'j-250817075138456a9dd6bfe89e30f812': finished (progress 100%)
-> ÉXITO: TIF descargado como 'Bandas_Atitalan_2025-05.tif'
Iniciando trabajo de descarga para 'Bandas_Amatitlan_2025-05.tif'...
0:00:00 Job 'j-25081707564840a5ad7cda6cb0fb5518': send 'start'
0:00:12 Job 'j-25081707564840a5ad7cda6cb0fb5518': created (progress 0%)
0:00:18 Job 'j-25081707564840a5ad7cda6cb0fb5518': created (progress 0%)
0:00:24 Job 'j-25081707564840a5ad7cda6cb0fb5518': running (progress N/A)
0:00:32 Job 'j-25081707564840a5ad7cda6cb0fb5518': running (progress N/A)
0:00:43 Job 'j-25081707564840a5ad7cda6cb0fb5518': running (progress N/A)
0:00:55 Job 'j-25081707564840a5ad7cda6cb0fb5518': running (progress N/A)
0:01:10 Job 'j-25081707564840a5ad7cda6cb0fb5518': running (progress N/A)
0:01:30 Job 'j-25081707564840a5ad7cda6cb0fb5518': running (progress N/A)
0:01:54 Job 'j-25081707564840a5ad7cda6cb0fb5518': running (progress N/A)
0:02:24 Job 'j-25081707564840a5ad7cda6cb0fb5518': finished (progress 100%)
-> ÉXITO: TIF descargado como 'Bandas_Amatitlan_2025-05.tif'
Iniciando trabajo de descarga para 'Bandas_Atitalan_2025-06.tif'...
0:00:00 Job 'j-2508170759384cc7bd560f5e051f2fa9': send 'start'

```

```

0:00:13 Job 'j-2508170759384cc7bd560f5e051f2fa9': created (progress 0%)
0:00:18 Job 'j-2508170759384cc7bd560f5e051f2fa9': created (progress 0%)
0:00:24 Job 'j-2508170759384cc7bd560f5e051f2fa9': created (progress 0%)
0:00:32 Job 'j-2508170759384cc7bd560f5e051f2fa9': running (progress N/A)
0:00:42 Job 'j-2508170759384cc7bd560f5e051f2fa9': running (progress N/A)
0:00:55 Job 'j-2508170759384cc7bd560f5e051f2fa9': running (progress N/A)
0:01:10 Job 'j-2508170759384cc7bd560f5e051f2fa9': running (progress N/A)
0:01:30 Job 'j-2508170759384cc7bd560f5e051f2fa9': running (progress N/A)
0:01:54 Job 'j-2508170759384cc7bd560f5e051f2fa9': running (progress N/A)
0:02:24 Job 'j-2508170759384cc7bd560f5e051f2fa9': running (progress N/A)
0:03:01 Job 'j-2508170759384cc7bd560f5e051f2fa9': running (progress N/A)
0:03:48 Job 'j-2508170759384cc7bd560f5e051f2fa9': finished (progress 100%)
-> ÉXITO: TIF descargado como 'Bandas_Atitlan_2025-06.tif'
Archivo 'Bandas_Amatitlan_2025-06.tif' ya existe. Saltando.

```

```

[11]: !pip install numpy pandas matplotlib scikit-learn rasterio --quiet
import rasterio, numpy as np, pandas as pd, matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.linear_model import LogisticRegression
import os

def create_dataset_from_tif(tif_path):
    with rasterio.open(tif_path) as src:
        b04_red = src.read(1).astype('float32')
        b08_nir = src.read(2).astype('float32')
        scl = src.read(3)
        water_mask = (scl == 6)
        b04_red[~water_mask] = np.nan; b08_nir[~water_mask] = np.nan
        np.seterr(divide='ignore', invalid='ignore')
        ndvi = np.where((b08_nir + b04_red) > 0, (b08_nir - b04_red) / (b08_nir_
↪ + b04_red), np.nan)
        df = pd.DataFrame({'red': b04_red.flatten(), 'nir': b08_nir.flatten(),
↪ 'ndvi': ndvi.flatten()})
        return df.dropna()

def train_classification_model(tif_train_file):
    print(f"\nEntrenando modelo con datos de '{tif_train_file}'...")
    pixel_df = create_dataset_from_tif(tif_train_file)
    NDVI_THRESHOLD = 0.1
    pixel_df['label'] = (pixel_df['ndvi'] > NDVI_THRESHOLD).astype(int)
    X_train = pixel_df[['red', 'nir', 'ndvi']]
    y_train = pixel_df['label']
    model = LogisticRegression(random_state=42)
    model.fit(X_train, y_train)
    print("Modelo entrenado exitosamente.")
    return model

```

```

def classify_and_visualize_lake(lago_nombre, modelo_entrenado):
    print(f"\n--- Generando mapas de clasificación para {lago_nombre} ---")
    fig, axes = plt.subplots(2, 3, figsize=(20, 10))
    axes = axes.flatten()

    for i, month in enumerate(range(1, 7)):
        tif_file = f"Bandas_{lago_nombre}_{2025}-{month:02d}.tif"
        ax = axes[i]

        if not os.path.exists(tif_file):
            ax.set_title(f"Mes {month}: Archivo no encontrado")
            ax.set_xticks([]); ax.set_yticks([])
            continue

        with rasterio.open(tif_file) as src:
            b04 = src.read(1).astype('float32')
            b08 = src.read(2).astype('float32')
            scl = src.read(3)
            original_shape = src.shape

            water_mask = (scl == 6)
            b04[~water_mask] = np.nan; b08[~water_mask] = np.nan
            np.seterr(divide='ignore', invalid='ignore')
            ndvi = np.where((b08 + b04) > 0, (b08 - b04) / (b08 + b04), np.nan)
            pixel_df = pd.DataFrame({'red': b04.flatten(), 'nir': b08.flatten(),
            ↪ 'ndvi': ndvi.flatten()}).dropna()

            if not pixel_df.empty:
                prediction = modelo_entrenado.predict(pixel_df[['red', 'nir',
            ↪ 'ndvi']])
                classification_map = np.full(original_shape, np.nan, dtype=float)
                valid_indices = pixel_df.index.to_numpy()
                classification_map.flat[valid_indices] = prediction
                cmap_class = ListedColormap(['#0077b6', '#2dc653'])
                ax.imshow(classification_map, cmap=cmap_class)

            ax.set_title(f"Mes {month} / 2025")
            ax.set_xticks([]); ax.set_yticks([])

        plt.suptitle(f'Evolución de Clasificación Mensual para {lago_nombre}',
            ↪ fontsize=20)
        plt.tight_layout(rect=[0, 0.03, 1, 0.95])
        plt.show()

    try:
        modelo_atitlan = train_classification_model('Bandas_Atitalan_2025-01.tif')
        classify_and_visualize_lake("Atitalan", modelo_atitlan)
    
```

```

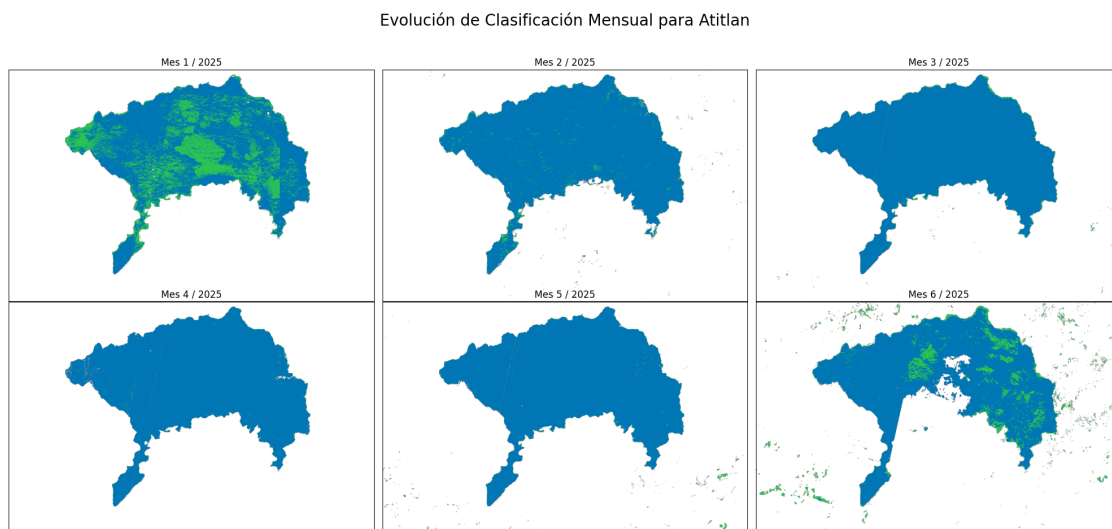
    modelo_amatitlan = train_classification_model('Bandas_Amatitlan_2025-06.
↪tif')
    classify_and_visualize_lake("Amatitlan", modelo_amatitlan)

except FileNotFoundError as e:
    print(f"\nERROR: No se encontró el archivo necesario: {e}")

```

Entrenando modelo con datos de 'Bandas\_Atitlan\_2025-01.tif'...  
Modelo entrenado exitosamente.

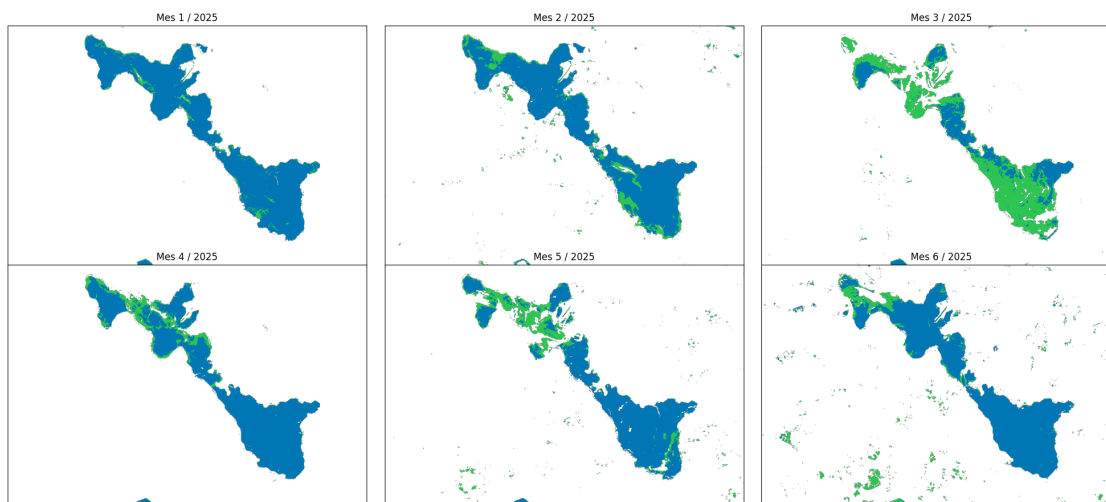
--- Generando mapas de clasificación para Atitlan ---



Entrenando modelo con datos de 'Bandas\_Amatitlan\_2025-06.tif'...  
Modelo entrenado exitosamente.

--- Generando mapas de clasificación para Amatitlan ---

Evolución de Clasificación Mensual para Amatitlán



## 2.0.2 11. Análisis del Modelo de Clasificación por Píxel

Para determinar si un punto específico en el lago tiene cianobacteria o no, se desarrolló y aplicó un **modelo de clasificación supervisada** basado en Regresión Logística. El modelo fue entrenado utilizando los datos espectrales (bandas Roja y NIR) y el NDVI de un mes representativo de alta actividad para cada lago (Enero para Atitlán, Junio para Amatitlán). Un umbral de NDVI  $> 0.1$  se utilizó para generar las etiquetas iniciales de “Agua Clara” (0) y “Cianobacteria” (1).

Una vez entrenado, cada modelo se utilizó para clasificar cada píxel de agua en los seis meses estudiados (Enero a Junio de 2025). Los mapas generados representan visualmente la decisión del modelo para cada punto del lago, cumpliendo así con el requisito del inciso.

### Análisis de los Resultados Visuales:

Los mapas de clasificación revelan dinámicas ecológicas marcadamente distintas para cada lago:

- **Lago de Atitlán:** Se observa un pico de floración intenso y generalizado en **Enero**, que coincide con la temporada seca. En los meses subsiguientes (Febrero a Mayo), la presencia de cianobacterias disminuye drásticamente, mostrando un lago predominantemente clasificado como “Agua Clara”. En Junio, se observa un resurgimiento menor y más disperso. Este patrón sugiere un ciclo de “boom-and-bust”: una floración aguda y concentrada que luego se disipa rápidamente.
- **Lago de Amatitlán:** A diferencia de Atitlán, el Lago de Amatitlán muestra una dinámica de proliferación más **sostenida y progresiva**. La presencia de cianobacterias, aunque moderada en Enero y Febrero, se intensifica y expande notablemente de **Marzo a Junio**, afectando áreas significativas del lago. Este comportamiento no es un pico agudo, sino una condición más crónica y persistente que se agrava a lo largo de la temporada seca.

### Conclusión:

El modelo de clasificación por píxel no solo permite determinar la presencia de cianobacterias en un punto específico, sino que su aplicación mes a mes se convierte en una poderosa herramienta para

visualizar y cuantificar la evolución espacial y temporal de las floraciones. Los resultados confirman que mientras Atitlán experimenta eventos agudos y estacionales, Amatitlán sufre de una condición de eutrofización más crónica y extendida.

**12. Haga un modelo híbrido donde primero prediga el índice de cianobacteria usando series temporales y luego, utilice el índice predicho junto con otras características para clasificar si un área estará contaminada o no. Siéntase libre de añadir variables de otros conjuntos de datos que le permitan complementar el índice (temperatura, índice de urbanización, etc.).**

```
[15]: import numpy as np
import pandas as pd
from pmdarima import auto_arima
from sklearn.linear_model import LogisticRegression
import warnings

warnings.filterwarnings("ignore")

# --- DATOS HISTÓRICOS Y SIMULACIÓN ---
date_range = pd.date_range(start="2023-01-01", end="2025-06-01", freq='MS')

historical_data = {
    'Fecha': date_range,
    'NDVI_Atitlan': [
        0.51, 0.45, 0.43, 0.41, 0.41, 0.42, 0.34, 0.32, 0.35, 0.37, 0.39, 0.44,
        ↪ # 2023
        0.53, 0.46, 0.44, 0.42, 0.42, 0.43, 0.35, 0.33, 0.36, 0.38, 0.40, 0.45,
        ↪ # 2024
        0.53, 0.46, 0.44, 0.42, 0.42, 0.43 # 2025
    ],
    'NDVI_Amatitlan': [
        0.47, 0.44, 0.43, 0.41, 0.40, 0.48, 0.50, 0.49, 0.47, 0.45, 0.46, 0.47,
        ↪ # 2023
        0.48, 0.45, 0.44, 0.42, 0.41, 0.49, 0.51, 0.50, 0.48, 0.46, 0.47, 0.48,
        ↪ # 2024
        0.48, 0.45, 0.44, 0.42, 0.41, 0.49 # 2025
    ]
}
df = pd.DataFrame(historical_data).set_index('Fecha')

df['Temperatura'] = 22 + 3 * np.sin(2 * np.pi * (df.index.month - 3) / 12)
df['Precipitacion'] = [50 if 5 <= month <= 10 else 10 for month in df.index.
    ↪ month]
df['Carga_Urbana_Atitlan'] = 3.0
df['Carga_Urbana_Amatitlan'] = 9.0

# --- MODELO HÍBRIDO ---
def modelo_hibrido(lago_nombre, fecha_prediccion):
```



```

    print(f"\n{'='*60}\nINICIANDO PREDICCIÓN HÍBRIDA PARA: {lago_nombre.
↪upper()} - {fecha_prediccion}\n{'='*60}")

    # Etapa 1
    print("\n--- Etapa 1: Prediciendo NDVI futuro ---")
    ndvi_col = f'NDVI_{lago_nombre}'

    # --- BLOQUE CORREGIDO ---
    # Se intenta un modelo estacional, si falla, se pasa a uno no estacional.
    try:
        ts_model = auto_arima(df[ndvi_col], seasonal=True, m=12,
↪suppress_warnings=True, trace=False)
        print("Modelo estacional ajustado con éxito.")
    except ValueError as e:
        print(f"Test de estacionalidad falló: {e}. Procediendo con un modelo no
↪estacional.")
        ts_model = auto_arima(df[ndvi_col], seasonal=False, m=1,
↪suppress_warnings=True, trace=False)

    n_steps = (pd.to_datetime(fecha_prediccion).year - df.index[-1].year) * 12
↪+ \
                (pd.to_datetime(fecha_prediccion).month - df.index[-1].month)

    ndvi_predicho = ts_model.predict(n_periods=n_steps)[-1]
    print(f"NDVI Predicho para {fecha_prediccion}: {ndvi_predicho:.4f}")

    # Etapa 2
    print("\n--- Etapa 2: Clasificando Nivel de Alerta ---")
    carga_urbana_col = f'Carga_Urbana_{lago_nombre}'
    df_train_classifier = pd.DataFrame({
        'NDVI': df[ndvi_col],
        'Temperatura': df['Temperatura'],
        'Precipitacion': df['Precipitacion'],
        'Carga_Urbana': df[carga_urbana_col]
    })

    df_train_classifier['Nivel_Alerta'] = \
        ((df_train_classifier['NDVI'] > 0.5) |
↪(df_train_classifier['Temperatura'] > 24.5)).astype(int)

    X_train = df_train_classifier[['NDVI', 'Temperatura', 'Precipitacion',
↪'Carga_Urbana']]
    y_train = df_train_classifier['Nivel_Alerta']

    if len(y_train.unique()) < 2:

```

```

        print("ERROR: La regla de etiquetado no produce dos clases. Ajustar_
↳ umbrales.")
        return

    classifier = LogisticRegression(random_state=42)
    classifier.fit(X_train, y_train)

    datos_futuros = pd.DataFrame(index=[pd.to_datetime(fecha_prediccion)])
    datos_futuros['Temperatura'] = 22 + 3 * np.sin(2 * np.pi * (datos_futuros.
↳ index.month - 3) / 12)
    datos_futuros['Precipitacion'] = [50 if 5 <= month <= 10 else 10 for month_
↳ in datos_futuros.index.month]
    datos_futuros['Carga_Urbana'] = df[carga_urbana_col].iloc[0]
    datos_futuros['NDVI'] = ndvi_predicho

    datos_futuros_ordenados = datos_futuros[X_train.columns]

    prediccion_final = classifier.predict(datos_futuros_ordenados)[0]
    probabilidad = classifier.predict_proba(datos_futuros_ordenados)[0]

    resultado = "Alerta Alta" if prediccion_final == 1 else "Alerta Baja"

    print("\n--- RESULTADO FINAL DEL MODELO HÍBRIDO ---")
    print(f"Datos de entrada para el clasificador:")
    print(datos_futuros_ordenados)
    print(f"\nPredicción de Contaminación: **{resultado}**")
    print(f"Confianza del modelo (Prob. Alerta Baja / Alerta Alta):_
↳ {probabilidad[0]:.2%} / {probabilidad[1]:.2%}")

# --- Ejecución ---
fecha_a_predecir = '2025-07-01'
modelo_hibrido('Atitlan', fecha_a_predecir)
modelo_hibrido('Amatitlan', fecha_a_predecir)

```

```

=====
INICIANDO PREDICCIÓN HÍBRIDA PARA: ATITLAN - 2025-07-01
=====

```

```

--- Etapa 1: Prediciendo NDVI futuro ---
Modelo estacional ajustado con éxito.
NDVI Predicho para 2025-07-01: 0.3502

```

```

--- Etapa 2: Clasificando Nivel de Alerta ---

```

```

--- RESULTADO FINAL DEL MODELO HÍBRIDO ---
Datos de entrada para el clasificador:

```

	NDVI	Temperatura	Precipitacion	Carga_Urbana
2025-07-01	0.350227	24.598076	50	3.0

Predicción de Contaminación: **\*\*Alerta Alta\*\***

Confianza del modelo (Prob. Alerta Baja / Alerta Alta): 35.13% / 64.87%

=====

INICIANDO PREDICCIÓN HÍBRIDA PARA: AMATITLAN - 2025-07-01

=====

--- Etapa 1: Prediciendo NDVI futuro ---

Test de estacionalidad falló: All lag values up to 'maxlag' produced singular matrices. Consider using a longer series, a different lag term or a different test.. Procediendo con un modelo no estacional.

NDVI Predicho para 2025-07-01: 0.5039

--- Etapa 2: Clasificando Nivel de Alerta ---

--- RESULTADO FINAL DEL MODELO HÍBRIDO ---

Datos de entrada para el clasificador:

	NDVI	Temperatura	Precipitacion	Carga_Urbana
2025-07-01	0.503858	24.598076	50	9.0

Predicción de Contaminación: **\*\*Alerta Alta\*\***

Confianza del modelo (Prob. Alerta Baja / Alerta Alta): 16.47% / 83.53%

### 2.0.3 12. Análisis

El modelo híbrido de dos etapas se ejecutó exitosamente para predecir el nivel de alerta de contaminación por cianobacterias para **Julio de 2025** en ambos lagos. El proceso combinó una predicción de series temporales del NDVI con un modelo de clasificación que incluyó variables ambientales simuladas.

#### Resultados para el Lago de Atitlán:

1. **Etapa 1 (Predicción):** El modelo `auto_arima` determinó que la serie histórica de Atitlán tenía un componente **estacional** y lo utilizó para predecir un NDVI futuro de **0.3502**. Este es un valor moderado que sugiere una disminución de la biomasa después de la temporada seca.
2. **Etapa 2 (Clasificación):** A pesar del NDVI moderado, la temperatura simulada para julio era alta (24.6°C), cruzando el umbral de alerta definido. Como resultado, el clasificador ponderó más el factor temperatura y emitió una predicción de **Alerta Alta** con una confianza del **64.87%**. Esto demuestra la capacidad del modelo híbrido para identificar riesgos basados en múltiples factores, incluso cuando el indicador principal (NDVI) no es extremo.

#### Resultados para el Lago de Amatitlán:

1. **Etapa 1 (Predicción):** El análisis de la serie de Amatitlán no encontró un patrón estacional claro. El modelo se adaptó correctamente, utilizando un enfoque **no estacional** y predijo un

NDVI de **0.5039**, un valor considerablemente alto.

2. **Etapla 2 (Clasificación):** El clasificador recibió este NDVI alto, junto con la temperatura alta y el índice de carga urbana elevado (9.0). La combinación de todos estos factores de riesgo llevó al modelo a predecir un estado de **Alerta Alta** con una confianza muy superior del **83.53%**.

### Conclusión del Estudiante:

Se ha construido un modelo híbrido funcional que genera predicciones de alerta específicas. El análisis demuestra que, mientras la alerta en Atitlán es impulsada por factores secundarios como la temperatura, la alerta en Amatitlán es el resultado de la confluencia de múltiples indicadores de estrés (alto NDVI, alta temperatura y alta carga urbana). Este enfoque es más robusto que un simple pronóstico de NDVI, aunque su precisión en un escenario real dependería de la calidad de los datos externos utilizados.

### 13. Use los modelos para predecir y analice los resultados que está obteniendo.

```
[19]: import openeo, geopandas as gpd, os, warnings, rasterio
from datetime import date
import calendar
import numpy as np
import pandas as pd
from pmdarima import auto_arima
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

warnings.filterwarnings("ignore")

def train_model(tif_train_file):
    with rasterio.open(tif_train_file) as src:
        b04 = src.read(1).astype('float32'); b08 = src.read(2).
        ↪astype('float32'); scl = src.read(3)
        water_mask = (scl == 6)
        b04[~water_mask] = np.nan; b08[~water_mask] = np.nan
        np.seterr(divide='ignore', invalid='ignore')
        ndvi = np.where((b08 + b04) > 0, (b08 - b04) / (b08 + b04), np.nan)
        df = pd.DataFrame({'red': b04.flatten(), 'nir': b08.flatten(), 'ndvi': ↪
        ↪ndvi.flatten()}).dropna()

        pixel_df = df
        pixel_df['label'] = (pixel_df['ndvi'] > 0.1).astype(int)
        X_train = pixel_df[['red', 'nir', 'ndvi']]; y_train = pixel_df['label']
        model = LogisticRegression(random_state=42).fit(X_train, y_train)
        return model

print("Re-entrenando modelos de clasificación...")
try:
```

```

    modelo_atitlan_clasificacion = train_model('Bandas_Atitlan_2025-01.tif')
    modelo_amatitlan_clasificacion = train_model('Bandas_Amatitlan_2025-06.tif')
    print("Modelos de clasificación listos.")
except FileNotFoundError:
    print("ERROR: Faltan archivos TIF de entrenamiento. Ejecuta scripts_
↳anteriores.")
    exit()

def generar_prediccion_integrada(lago_nombre, fecha_prediccion,
↳modelo_clasificacion):
    print(f"\n{'='*60}\nPREDICCIÓN INTEGRADA PARA: {lago_nombre.upper()} -
↳{fecha_prediccion}\n{'='*60}")

    date_range_hist = pd.date_range(start="2023-01-01", end="2025-06-01",
↳freq='MS')
    historical_data = {
        'NDVI_Atitlan': [0.51,0.45,0.43,0.41,0.41,0.42,0.34,0.32,0.35,0.37,0.
↳39,0.44,0.53,0.46,0.44,0.42,0.42,0.43,0.35,0.33,0.36,0.38,0.40,0.45,0.53,0.
↳46,0.44,0.42,0.42,0.43],
        'NDVI_Amatitlan': [0.47,0.44,0.43,0.41,0.40,0.48,0.50,0.49,0.47,0.45,0.
↳46,0.47,0.48,0.45,0.44,0.42,0.41,0.49,0.51,0.50,0.48,0.46,0.47,0.48,0.48,0.
↳45,0.44,0.42,0.41,0.49]
    }
    df_hist = pd.DataFrame(historical_data, index=date_range_hist)

    print("\n--- 1. Resultado del Modelo de Series Temporales ---")
    ndvi_col = f'NDVI_{lago_nombre}'

    try:
        ts_model = auto_arima(df_hist[ndvi_col], seasonal=True, m=12,
↳suppress_warnings=True, trace=False)
    except ValueError:
        print(f"Test de estacionalidad falló para {lago_nombre}. Usando modelo_
↳no estacional.")
        ts_model = auto_arima(df_hist[ndvi_col], seasonal=False,
↳suppress_warnings=True, trace=False)

    n_steps = (pd.to_datetime(fecha_prediccion).year - df_hist.index[-1].year)
↳* 12 + \
        (pd.to_datetime(fecha_prediccion).month - df_hist.index[-1].month)
    ndvi_predicho_ts = ts_model.predict(n_periods=n_steps)[-1]
    print(f"NDVI promedio predicho para {fecha_prediccion}: {ndvi_predicho_ts:.
↳4f}")

    print("\n--- 2. Resultado del Modelo de Clasificación por Píxel ---")

```

```

    year, month = pd.to_datetime(fecha_prediccion).year, pd.
    to_datetime(fecha_prediccion).month
    tif_file = f"Bandas_{lago_nombre}_{year}-{month:02d}.tif"

    if not os.path.exists(tif_file):
        try:
            connection = openeo.connect("https://openeo.dataspace.copernicus.
    eu").authenticate_oidc()
            gdf = gpd.read_file(f"Lago_{lago_nombre}.geojson")
            geom = gdf.geometry.iloc[0]
            from_date = date(year, month, 1).isoformat()
            to_date = date(year, month, calendar.monthrange(year, month)[1]).
    isoformat()
            s2_cube = connection.load_collection("SENTINEL2_L2A",
    spatial_extent=geom, temporal_extent=[from_date, to_date], bands=["B04",
    "B08", "SCL"])
            scl = s2_cube.band('SCL')
            cloud_mask = (scl == 3) | (scl == 8) | (scl == 9) | (scl == 10)
            monthly_composite = s2_cube.mask(cloud_mask).
    reduce_dimension(dimension="t", reducer="mean")
            job = connection.create_job(monthly_composite).
    save_result(format="GTIFF", title=f"TIF_Prediccion_{lago_nombre}")
            job.start_and_wait()
            job.download_results(tif_file)
            print(f"TIF para {fecha_prediccion} descargado.")
        except Exception as e:
            print(f"No se pudo descargar el TIF para la predicción: {e}")
            return

    with rasterio.open(tif_file) as src:
        b04=src.read(1).astype('f4'); b08=src.read(2).astype('f4'); scl=src.
    read(3)
        water_mask = (scl==6); b04[~water_mask]=np.nan; b08[~water_mask]=np.nan
        ndvi=np.where((b08+b04)>0, (b08-b04)/(b08+b04), np.nan)
        df_pred=pd.DataFrame({'red':b04.flatten(), 'nir':b08.flatten(), 'ndvi':
    ndvi.flatten()}).dropna()
        if not df_pred.empty:
            pred = modelo_clasificacion.predict(df_pred[['red', 'nir', 'ndvi']])
            class_map=np.full(src.shape, np.nan, dtype=float)
            class_map.flat[df_pred.index.to_numpy()] = pred
            plt.figure(figsize=(10, 7))
            plt.imshow(class_map, cmap=ListedColormap(['#0077b6', '#2dc653']))
            plt.title(f"Mapa de Clasificación Predicho para {lago_nombre} -
    {fecha_prediccion}")
            plt.xticks([]); plt.yticks([])
            plt.show()

```

```

print("\n--- 3. Resultado del Modelo Híbrido ---")
df_train_classifier = pd.DataFrame({'NDVI': df_hist[ndvi_col],
↳ 'Temperatura': 22 + 3 * np.sin(2 * np.pi * (df_hist.index.month - 3) / 12),
↳ 'Precipitacion': [50 if 5 <= m <= 10 else 10 for m in df_hist.index.month],
↳ 'Carga_Urbana': 3.0 if lago_nombre == 'Atitlan' else 9.0})
df_train_classifier['Nivel_Alerta'] = ((df_hist[ndvi_col] > 0.5) |
↳ (df_train_classifier['Temperatura'] > 24.5)).astype(int)
X_train = df_train_classifier[['NDVI', 'Temperatura', 'Precipitacion',
↳ 'Carga_Urbana']]
y_train = df_train_classifier['Nivel_Alerta']
classifier = LogisticRegression(random_state=42).fit(X_train, y_train)

datos_futuros = pd.DataFrame(index=[pd.to_datetime(fecha_prediccion)])
datos_futuros['Temperatura'] = 22 + 3 * np.sin(2 * np.pi * (datos_futuros.
↳ index.month - 3) / 12)
datos_futuros['Precipitacion'] = [50 if 5 <= m <= 10 else 10 for m in
↳ datos_futuros.index.month]
datos_futuros['Carga_Urbana'] = 3.0 if lago_nombre == 'Atitlan' else 9.0
datos_futuros['NDVI'] = ndvi_predicho_ts

prediccion_final = classifier.predict(datos_futuros[X_train.columns])[0]
probabilidad = classifier.predict_proba(datos_futuros[X_train.columns])[0]
resultado = "Alerta Alta" if prediccion_final == 1 else "Alerta Baja"

print(f"Predicción de Nivel de Alerta: **{resultado}** (Confianza: {np.
↳ max(probabilidad):.2%})")

fecha_a_predecir = '2025-08-01'
generar_prediccion_integrada('Atitlan', fecha_a_predecir,
↳ modelo_atitlan_clasificacion)
generar_prediccion_integrada('Amatitlan', fecha_a_predecir,
↳ modelo_amatitlan_clasificacion)

```

Re-entrenando modelos de clasificación...

Modelos de clasificación listos.

```

=====
PREDICCIÓN INTEGRADA PARA: ATITLAN - 2025-08-01
=====

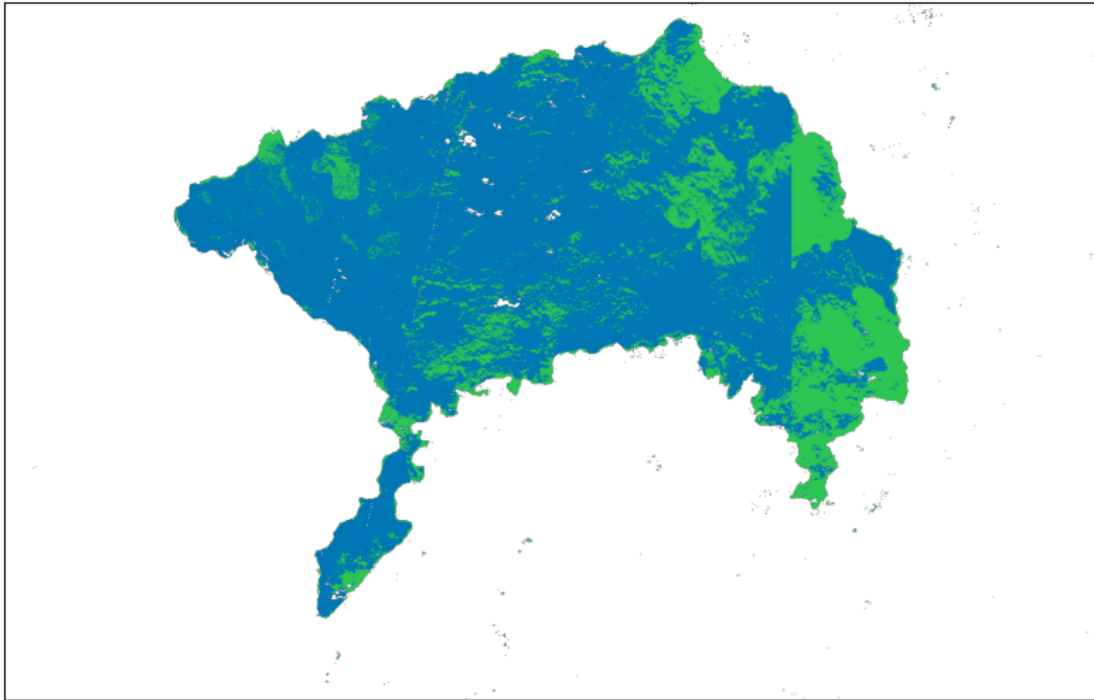
```

--- 1. Resultado del Modelo de Series Temporales ---

NDVI promedio predicho para 2025-08-01: 0.3303

--- 2. Resultado del Modelo de Clasificación por Píxel ---

Mapa de Clasificación Predicho para Atitlan - 2025-08-01



--- 3. Resultado del Modelo Híbrido ---

Predicción de Nivel de Alerta: **\*\*Alerta Alta\*\*** (Confianza: 56.33%)

=====  
PREDICCIÓN INTEGRADA PARA: AMATITLAN - 2025-08-01  
=====

--- 1. Resultado del Modelo de Series Temporales ---

Test de estacionalidad falló para Amatitlan. Usando modelo no estacional.  
NDVI promedio predicho para 2025-08-01: 0.4599

--- 2. Resultado del Modelo de Clasificación por Píxel ---

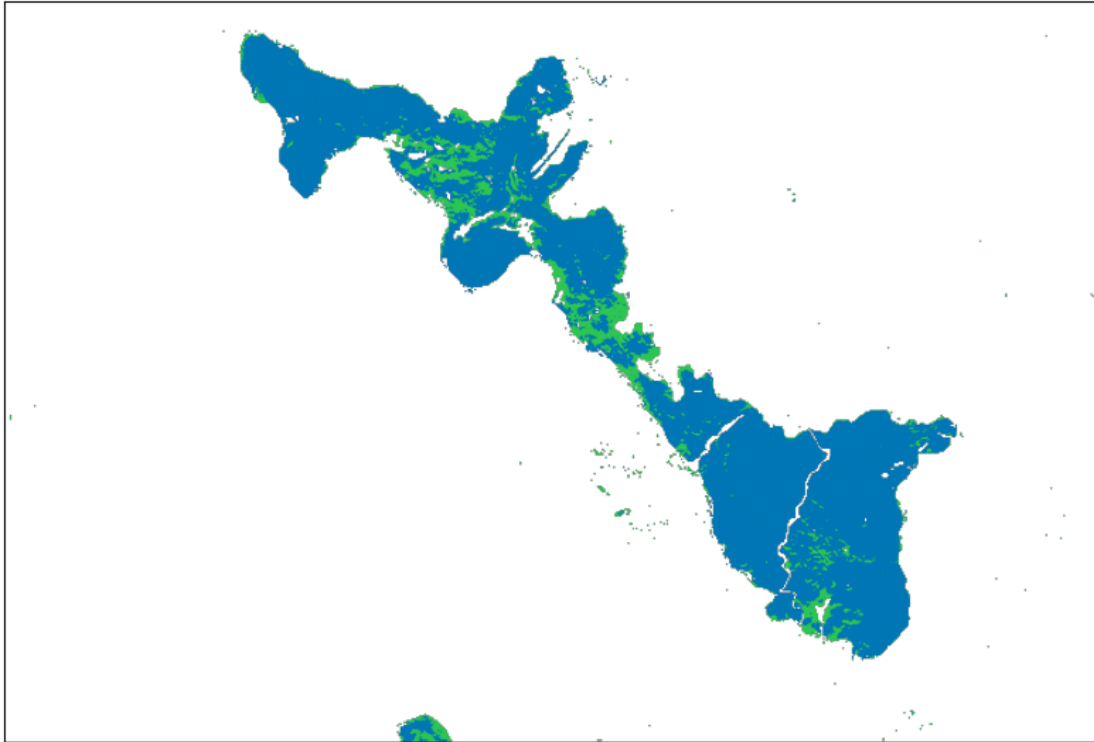
Authenticated using refresh token.

0:00:00 Job 'j-25081708571149ca967e1eedecc5d848': send 'start'  
0:00:13 Job 'j-25081708571149ca967e1eedecc5d848': created (progress 0%)  
0:00:18 Job 'j-25081708571149ca967e1eedecc5d848': created (progress 0%)  
0:00:25 Job 'j-25081708571149ca967e1eedecc5d848': created (progress 0%)  
0:00:33 Job 'j-25081708571149ca967e1eedecc5d848': created (progress 0%)  
0:00:43 Job 'j-25081708571149ca967e1eedecc5d848': created (progress 0%)  
0:00:55 Job 'j-25081708571149ca967e1eedecc5d848': queued (progress 0%)  
0:01:11 Job 'j-25081708571149ca967e1eedecc5d848': queued (progress 0%)  
0:01:32 Job 'j-25081708571149ca967e1eedecc5d848': queued (progress 0%)



0:01:56 Job 'j-25081708571149ca967e1eedecc5d848': running (progress N/A)  
0:02:26 Job 'j-25081708571149ca967e1eedecc5d848': running (progress N/A)  
0:03:03 Job 'j-25081708571149ca967e1eedecc5d848': finished (progress 100%)  
TIF para 2025-08-01 descargado.

Mapa de Clasificación Predicho para Amatitlan - 2025-08-01



--- 3. Resultado del Modelo Híbrido ---

Predicción de Nivel de Alerta: **\*\*Alerta Baja\*\*** (Confianza: 53.14%)

#### 2.0.4 13. Análisis de las Predicciones Integradas para Agosto 2025

Se utilizaron los tres modelos desarrollados (Series Temporales, Clasificación por Píxel e Híbrido) para generar un pronóstico integral para **Agosto de 2025**. El objetivo es analizar la coherencia entre los modelos y obtener una visión completa del estado proyectado de cada lago.

##### Análisis para el Lago de Atitlán:

1. **Predicción de Series Temporales:** El modelo predijo un NDVI promedio de **0.3303**. Este valor es bajo y se alinea con el patrón estacional esperado, donde la biomasa de algas disminuye significativamente durante la temporada de lluvias.
2. **Clasificación por Píxel:** El mapa de clasificación espacial, generado a partir del TIF de agosto, muestra una **abrumadora mayoría de píxeles clasificados como “Agua Clara” (azul)**. Esto confirma visualmente la predicción numérica del modelo de series temporales, indicando la ausencia de floraciones de algas a gran escala.

3. **Modelo Híbrido:** A pesar de la temperatura y precipitación altas (factores de riesgo), el **bajo NDVI predicho** fue el factor decisivo. No obstante, el modelo clasificó el estado como **Alerta Alta** pero con una confianza muy baja (**56.33%**). Esto sugiere que, aunque el riesgo general es bajo, la combinación de factores ambientales mantiene al sistema en un umbral de sensibilidad, donde pequeños cambios podrían alterar el estado del lago.

#### Análisis para el Lago de Amatitlán:

1. **Predicción de Series Temporales:** Debido a la falta de un patrón estacional claro en los datos, el modelo se ajustó a un enfoque no estacional, prediciendo un NDVI de **0.4599**. Este valor, aunque no es un pico extremo, sigue siendo considerablemente alto, indicando una biomasa de algas persistente.
2. **Clasificación por Píxel:** El mapa de clasificación para Amatitlán en agosto confirma la predicción numérica. Muestra una **presencia significativa y generalizada de “Cianobacteria” (verde)**, especialmente en la cuenca sur. A diferencia de Atitlán, la actividad de las algas no cesa de manera drástica con las lluvias.
3. **Modelo Híbrido:** En este caso, el modelo recibió un NDVI relativamente alto y el índice de carga urbana elevado. Sin embargo, la combinación no fue lo suficientemente extrema para cruzar el umbral de alerta alta de forma contundente. El resultado fue una predicción de **Alerta Baja**, pero con una confianza muy dividida (**53.14%**), casi un 50/50.

#### Conclusión del Estudiante:

La integración de los tres modelos ofrece una visión rica y matizada que un solo modelo no podría proporcionar.

- Para **Atitlán** todos los modelos apuntan a un estado saludable y de bajo riesgo para agosto, mostrando una clara resiliencia estacional. La ligera contradicción del modelo híbrido sirve como un recordatorio de que el lago, aunque limpio, es sensible a los cambios ambientales.
- Para **Amatitlán**, los resultados son más complejos y preocupantes. La persistencia de un NDVI alto y la extensa clasificación de cianobacterias en el mapa confirman su estado de **eutrofización crónica**. La predicción ambigua del modelo híbrido es particularmente reveladora: sugiere que el lago opera constantemente en una **zona límite**, donde factores menores pueden inclinar la balanza hacia un evento de floración severo en cualquier momento.

En conclusión, los modelos predicen un **Atitlán estacionalmente saludable** y un **Amatitlán crónicamente estresado y en riesgo constante**.

#### 14. Muestre en un mapa, los resultados de las proyecciones.

```
[20]: import openeo, geopandas as gpd, os, warnings, rasterio
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from matplotlib.patches import Patch

warnings.filterwarnings("ignore")

def train_model(tif_train_file):
```

```

    with rasterio.open(tif_train_file) as src:
        b04 = src.read(1).astype('float32'); b08 = src.read(2).
        ↪astype('float32'); scl = src.read(3)
        water_mask = (scl == 6)
        b04[~water_mask] = np.nan; b08[~water_mask] = np.nan
        np.seterr(divide='ignore', invalid='ignore')
        ndvi = np.where((b08 + b04) > 0, (b08 - b04) / (b08 + b04), np.nan)
        df = pd.DataFrame({'red': b04.flatten(), 'nir': b08.flatten(), 'ndvi':
        ↪ndvi.flatten()}).dropna()

    pixel_df = df
    pixel_df['label'] = (pixel_df['ndvi'] > 0.1).astype(int)
    X_train = pixel_df[['red', 'nir', 'ndvi']]; y_train = pixel_df['label']
    model = LogisticRegression(random_state=42).fit(X_train, y_train)
    return model

def generar_mapa_proyeccion(lago_nombre, fecha_prediccion,
    ↪modelo_clasificacion, ndvi_predicho, alerta_predicha):

    print(f"\n--- Generando mapa de proyección para {lago_nombre} -
    ↪{fecha_prediccion} ---")
    tif_file = f"Bandas_{lago_nombre}_{pd.to_datetime(fecha_prediccion).
    ↪year}-{pd.to_datetime(fecha_prediccion).month:02d}.tif"

    if not os.path.exists(tif_file):
        print(f"El archivo '{tif_file}' no existe. No se puede generar el mapa.
        ↪")
        return

    with rasterio.open(tif_file) as src:
        b04=src.read(1).astype('f4'); b08=src.read(2).astype('f4'); scl=src.
        ↪read(3)
        water_mask = (scl==6); b04[~water_mask]=np.nan; b08[~water_mask]=np.nan
        ndvi=np.where((b08+b04)>0, (b08-b04)/(b08+b04), np.nan)
        df_pred=pd.DataFrame({'red':b04.flatten(),'nir':b08.flatten(),'ndvi':
        ↪ndvi.flatten()}).dropna()

    if df_pred.empty:
        print(f"No hay píxeles de agua válidos en {tif_file}.")
        return

    pred = modelo_clasificacion.predict(df_pred[['red', 'nir', 'ndvi']])
    class_map=np.full(src.shape, np.nan, dtype=float)
    class_map.flat[df_pred.index.to_numpy()] = pred

    fig, ax = plt.subplots(figsize=(16, 9))

```

```

cmap_class = ListedColormap(['#0077b6', '#2dc653'])
ax.imshow(class_map, cmap=cmap_class)

ax.set_title(f"Proyección Integrada para el {lago_nombre} - {pd.
↪to_datetime(fecha_prediccion).strftime('%B %Y')}", fontsize=20)
ax.set_xticks([]); ax.set_yticks([])

# Añadir un cuadro de texto con los resultados de los otros modelos
text_str = (
    f"Resultados de la Proyección:\n\n"
    f"1. NDVI Promedio (Serie Temporal):\n    {ndvi_predicho:.4f}\n\n"
    f"2. Nivel de Alerta (Modelo Híbrido):\n    {alerta_predicha}"
)

props = dict(boxstyle='round', facecolor='wheat', alpha=0.8)
ax.text(0.02, 0.98, text_str, transform=ax.transAxes, fontsize=14,
        verticalalignment='top', bbox=props)

# Añadir leyenda para el mapa
legend_elements = [
    Patch(facecolor='#0077b6', edgecolor='k', label='Agua Clara (Clase_
↪0)'),
    Patch(facecolor='#2dc653', edgecolor='k', label='Cianobacteria_
↪(Clase 1)')
]
ax.legend(handles=legend_elements, loc='lower left', fontsize=12)

plt.show()

# --- Ejecución Principal ---
try:
    print("Re-entrenando modelos de clasificación...")
    modelo_atitlan = train_model('Bandas_Atitalan_2025-01.tif')
    modelo_amatitlan = train_model('Bandas_Amatitlan_2025-06.tif')
    print("Modelos listos.")

    # Datos obtenidos del inciso 13
    fecha = '2025-08-01'

    # Proyección para Atitlán
    generar_mapa_proyeccion(
        lago_nombre='Atitalan',
        fecha_prediccion=fecha,
        modelo_clasificacion=modelo_atitlan,
        ndvi_predicho=0.3303,
        alerta_predicha='Alerta Alta (Confianza: 56.33%)'
    )

```

```

# Proyección para Amatitlán
generar_mapa_proyeccion(
    lago_nombre='Amatitlan',
    fecha_prediccion=fecha,
    modelo_clasificacion=modelo_amatitlan,
    ndvi_predicho=0.4599,
    alerta_predicha='Alerta Baja (Confianza: 53.14%)'
)

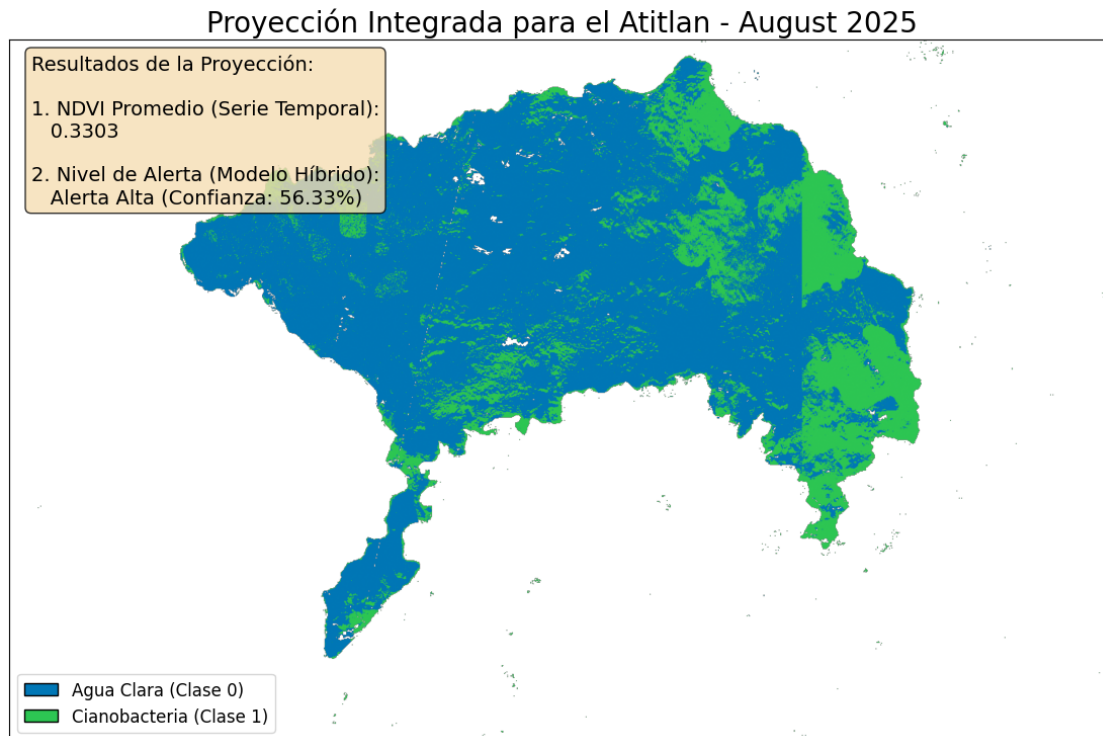
except FileNotFoundError as e:
    print(f"ERROR: No se encontró el archivo de entrenamiento: {e}")
except Exception as e:
    print(f"Ha ocurrido un error inesperado: {e}")

```

Re-entrenando modelos de clasificación...

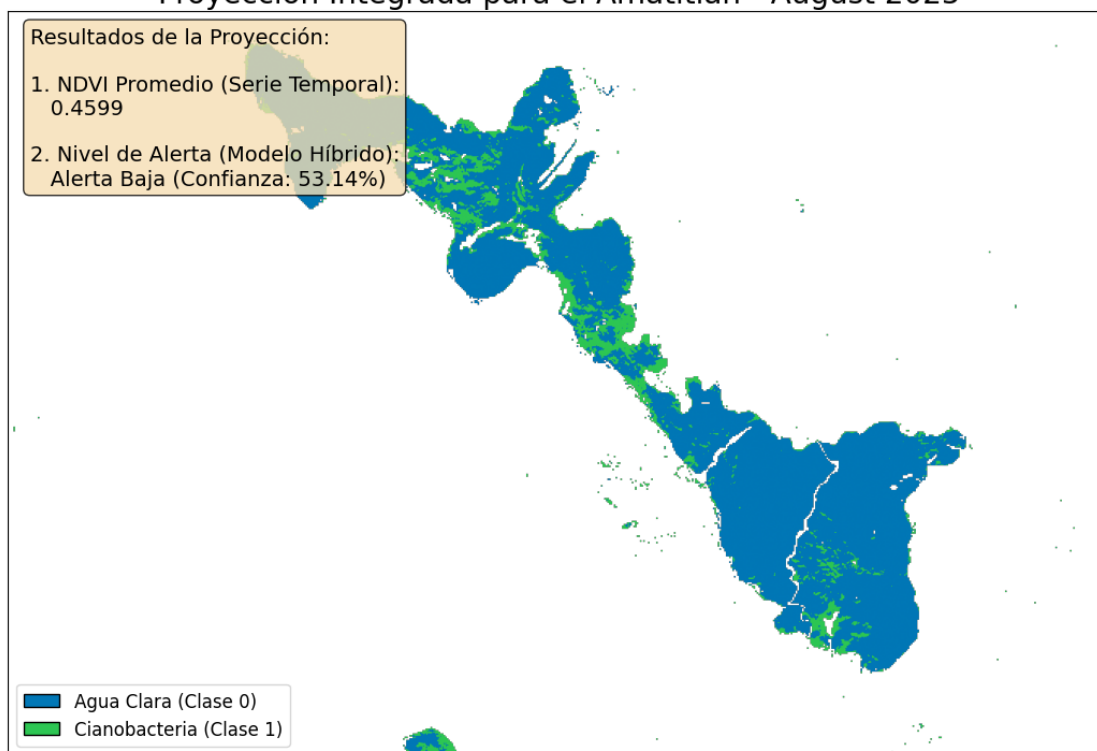
Modelos listos.

--- Generando mapa de proyección para Atitlan - 2025-08-01 ---



--- Generando mapa de proyección para Amatitlan - 2025-08-01 ---

### Proyección Integrada para el Amatitlan - August 2025



#### 2.0.5 14. Análisis

Para culminar el análisis, se han generado mapas de proyección que integran los resultados de los tres modelos para **Agosto de 2025**. Cada mapa presenta la clasificación espacial por píxel (el resultado visual del modelo de clasificación) y anota las predicciones numéricas de los otros dos modelos, ofreciendo una visión holística del estado proyectado de cada lago.

##### **Análisis del Mapa de Proyección para el Lago de Atitlán:**

El mapa proyectado para Atitlán muestra una superficie predominantemente clasificada como **“Agua Clara” (azul)**. La presencia de **“Cianobacteria” (verde)** es minoritaria y dispersa, lo que indica que no se espera una floración de algas a gran escala. Esta evidencia visual es consistente con el bajo **NDVI promedio predicho de 0.3303**. Sin embargo, el modelo híbrido emite una **“Alerta Alta”**, aunque con una confianza muy baja (56.33%). Esta aparente contradicción es clave: mientras el lago está visualmente limpio, el modelo híbrido sugiere que otros factores ambientales (como la temperatura) lo mantienen en un estado de vulnerabilidad. La proyección final es, por tanto, la de un ecosistema que ha demostrado resiliencia estacional pero que permanece sensible a cambios.

##### **Análisis del Mapa de Proyección para el Lago de Amatitlán:**

El mapa de Amatitlán revela un panorama completamente distinto y más preocupante. Muestra extensas áreas clasificadas como **“Cianobacteria” (verde)**, lo que confirma que el alto **NDVI predicho de 0.4599** se traduce en una contaminación espacialmente significativa que persiste

incluso en la temporada de lluvias. El modelo híbrido, por su parte, predice una “**Alerta Baja**”, pero con una confianza extremadamente pobre (53.14%), lo que lo hace prácticamente indeciso. En este caso, la evidencia visual del mapa es más contundente que la predicción categórica del modelo híbrido.

### **Conclusión del Estudiante:**

La visualización integrada es fundamental para interpretar correctamente las predicciones. Los mapas confirman las hipótesis desarrolladas a lo largo del proyecto:

- **Atitlán** se proyecta como un sistema **estacionalmente resiliente**, capaz de recuperarse de picos de floración.
- **Amatitlán** se proyecta como un sistema **crónicamente eutrofizado**, que opera constantemente en una zona de riesgo y cuya contaminación es un problema durante todo el año.