Sentinel Hub NDVI Data Retrieval and Analysis Script

Overview

This document provides a detailed description of a Python script that interacts with the Sentinel Hub API to retrieve, process, and analyze NDVI (Normalized Difference Vegetation Index) data. The script is designed to handle various steps, including authentication, data retrieval, processing, and a simple statistical analysis including the calculation of the mean, minimum and maximum values of the NDVI in a provided area.

Steps Performed by the Script

1. Imports necessary libraries and configures logging.
2. Defines configuration parameters for OAuth2 authentication and API endpoints.
3. Creates a function to obtain an OAuth2 session using client credentials.
4. Defines a function to send a request to the Sentinel Hub API to retrieve data.
5. Creates a function to extract the contents of a TAR file.
6. Defines a function to calculate NDVI statistics (mean, minimum, maximum) from a GeoTIFF file.
7. Creates a function to save the calculated NDVI statistics to a JSON file.
8. Orchestrates the entire process in the main function:
   - Obtains an OAuth2 session.
   - Defines an evalscript for data processing.
   - Creates a request payload with specific parameters and bounds.

- Sends the request to the API and saves the response as a TAR file.
- Extracts the TAR file to a specified directory.
- Calculates NDVI statistics from the extracted GeoTIFF file.
- Saves the NDVI statistics to a JSON file.

Script Documentation

Import libraries

The script starts by importing the necessary libraries:

- **os**: For interacting with the operating system, particularly for file path manipulations.
- **json**: For reading and writing JSON files.
- **tarfile**: For working with TAR files.
- **numpy (np)**: For numerical operations, particularly on arrays.
- **rasterio**: For reading and writing geospatial raster data.
- **oauthlib.oauth2**: For handling OAuth2 authentication.
- **requests_oauthlib**: For making requests using OAuth2 authentication.
- **logging**: For logging information and errors during script execution.

Configure Logging

The logging is configured to display messages with the level of INFO or higher, formatted with the time, log level, and message.

```
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
```

Configuration

Several configuration parameters are defined, including the client ID, client secret, URLs for token and process requests, file paths, and coordinates for the area of interest.

OAuth2 Session

The `get_oauth_session` function creates and returns an OAuth2 session for accessing the Sentinel Hub API.

Request Data from Sentinel Hub API

The `request_data` function sends a POST request to the Sentinel Hub API to retrieve data based on the provided request payload.

Extract TAR File

The `extract_tar_file` function extracts the contents of a TAR file to a specified directory.

Calculate NDVI Statistics

The `calculate_ndvi_statistics` function calculates the mean, minimum, and maximum values of NDVI from a given GeoTIFF file.

Save Statistics to JSON File

The `save_to_json` function saves the provided data to a JSON file.

Main Function

The `main` function orchestrates the entire process, from obtaining an OAuth2 session to saving the NDVI statistics to a JSON file.

Detailed Function Descriptions

GET_OAUTH_SESSION(CLIENT_ID: STR, CLIENT_SECRET: STR) -> OAUTH2SESSION

Description:
Creates and returns an OAuth2 session for accessing the Sentinel Hub API.

Parameters:

- `client_id (str)`: The client ID for the OAuth2 application.
- `client_secret (str)`: The client secret for the OAuth2 application.

Returns:

- `OAuth2Session`: An authenticated OAuth2 session object for making API requests.

Raises:

- `Exception`: If there is an error during the OAuth2 session creation or token fetching.

REQUEST_DATA(OAUTH: OAUTH2SESSION, REQUEST_PAYLOAD: DICT) -> BOOL

Description:
Sends a POST request to the Sentinel Hub API to retrieve data based on the provided request payload.

Parameters:

- `oauth (OAuth2Session)`: An authenticated OAuth2 session object for making API requests.
- `request_payload (dict)`: The request payload JSON containing the necessary parameters for data retrieval.

Returns:

- `bool`: True if the data is successfully retrieved and saved to the TAR_FILE, False otherwise.

Raises:

- `Exception`: If there is an error during the data request or writing the response content to the TAR_FILE.

EXTRACT_TAR_FILE(TAR_PATH: STR, EXTRACT_TO: STR) -> NONE

Description:
Extracts the contents of a TAR file to a specified directory.

Parameters:

- `tar_path (str)`: The path to the TAR file to be extracted.
- `extract_to (str)`: The directory where the contents of the TAR file will be extracted.

Returns:
- None

Raises:
- `Exception`: If there is an error during the extraction process.

CALCULATE_NDVI_STATISTICS(TIF_FILE: STR) -> DICT

Description:
Calculates the mean, minimum, and maximum values of NDVI from a given GeoTIFF file.

Parameters:
- `tif_file (str)`: The path to the GeoTIFF file containing the NDVI data.

Returns:
- `dict`: A dictionary containing the mean, minimum, and maximum NDVI values.

Raises:
- `Exception`: If there is an error reading the GeoTIFF file or calculating the NDVI statistics.

SAVE_TO_JSON(DATA: DICT, JSON_FILE: STR) -> NONE

Description:
Saves the provided data to a JSON file.

Parameters:
- `data (dict)`: The data to be saved. It should be a dictionary.
- `json_file (str)`: The path to the JSON file where the data will be saved.

Returns:
- None

Raises:
- `Exception`: If there is an error while opening or writing to the JSON file.

MAIN() -> NONE

Description:
Orchestrates the entire process of retrieving data from the Sentinel Hub API, processing it using an evalscript, extracting the data from the TAR file, calculating NDVI statistics, and saving the statistics to a JSON file.

Returns:
- None

Raises:
- `Exception`: If any error occurs during the process.

Usage Instructions

1. Replace `CLIENT_ID` and `CLIENT_SECRET` with your actual Sentinel Hub credentials.
2. Modify the given `POLYGON_COORDINATES` with your wished bounds for the desired map you want to calculate the NDVI for. Remember that the last coordinate must equal the first given coordinate to close the polygon.
3. Ensure all necessary libraries are installed.
4. Run the script to retrieve and process NDVI data.