Documentation for PDF Download and Table Extraction Script for New Zealand Tarrif

Note: Do not run the script in one go, run selectively as they were few manual validation in between scraping the data and cleaning the data.

This script is designed to automate the process of downloading PDF files from a specified base URL - https://www.customs.govt.nz/globalassets/documents/tariff-documents
, extracting tables from those PDFs, and saving the extracted tables into a combined CSV file. The script is particularly useful for handling large sets of PDFs that contain tabular data.

Script Overview:
The script performs the following tasks:

1. Defines valid sections for each year.
2. Generates PDF URLs dynamically based on the valid sections.
3. Downloads the PDF files from the generated URLs.
4. Extracts tables from the downloaded PDFs using the pdfplumber library.
5. Combines all extracted tables into a single CSV file.

Step-by-Step:

1. Define Valid Sections for Each Year
    - The script starts by defining a dictionary called valid_sections. This dictionary maps years (e.g., 2024, 2025) to lists of valid sections (e.g., "i", "ii", "iii"). These sections represent parts of the PDF documents that need to be processed.
2. Generate PDF URLs Dynamically
    - The generate_pdf_urls function creates a list of PDF URLs based on the valid sections for each year. The URLs are constructed using a base URL and the section names. For example, if the base URL is "https://www.customs.govt.nz/globalassets/documents/tariff-documents", and the year is 2024 with section "i", the generated URL will be "https://www.customs.govt.nz/globalassets/documents/tariff-documents/wtd-2024/section-i-july-2024.pdf".

3. Download PDF Files
    - The download_pdf function downloads a PDF file from a given URL and saves it to a specified local path. It uses the requests library to fetch the PDF content. If the download is successful, the PDF is saved locally, and the function returns True. If the download fails (e.g., due to a network error or invalid URL), the function returns False.

4. Extract Tables from PDF Files
    - The extract_tables_from_pdf function uses the pdfplumber library to extract tables from a PDF file. It iterates through each page of the PDF and extracts tables using custom settings to improve accuracy. The extracted tables are returned as a list of lists, where each inner list represents a row in the table.

5. Main Function to Download and Process PDFs
    - The process_pdfs function orchestrates the entire process. It performs the following steps:
        a. Creates an output directory if it does not already exist.
        b. Generates PDF URLs using the generate_pdf_urls function.
        c. Downloads each PDF using the download_pdf function.
        d. Extracts tables from each downloaded PDF using the extract_tables_from_pdf function.
        e. Combines all extracted tables into a single pandas DataFrame.
        f. Saves the combined DataFrame as a CSV file in the output directory.

Key Functions:

1. generate_pdf_urls(base_url, valid_sections)
    - Input: base_url (str), valid_sections (dict)
    - Output: List of PDF URLs (list of str)
    - Description: Generates PDF URLs dynamically based on the valid sections for each year.

2. download_pdf(url, save_path)
    - Input: url (str), save_path (str)
    - Output: Boolean (True if download is successful, False otherwise)
    - Description: Downloads a PDF file from the specified URL and saves it to the given path.

3. extract_tables_from_pdf(pdf_path)
    - Input: pdf_path (str)
    - Output: List of tables (list of lists)
    - Description: Extracts tables from a PDF file using the pdfplumber library.

4. process_pdfs(base_url, valid_sections, output_dir)
    - Input: base_url (str), valid_sections (dict), output_dir (str)
    - Output: None (saves extracted tables to a CSV file)
    - Description: Downloads PDFs, extracts tables, and saves the combined tables to a CSV file.

Usage:

1. Set the base_url variable to the base URL of the PDFs.
2. Define the valid_sections dictionary with the appropriate sections for each year.
3. Specify the output_directory where the downloaded PDFs and extracted tables will be saved.
4. Run the script. It will:

- Download the PDFs.
- Extract tables from the PDFs.
- Save the combined tables to a CSV file in the output directory.

Dependencies:

- **os**: For handling file paths and directories.
- **requests:** For downloading PDF files.
- **pdfplumber**: For extracting tables from PDFs.
- **pandas**: For combining and saving tables as a CSV file.

- Install the required libraries using pip:
    - **pip install requests pdfplumber pandas**

Further Data Cleaning:

- Due to non-uniform format of the tables after scrapping, to avoid losing valuable info, all column values from **Goods to the end of the data was joined together into one column for cleaning and analysis using REGEX.**
- A lot of cleaning happened manually as all could not be scripted.
- **KINDLY DO THIS IN EXCEL FIRST BEFORE RUNING THE CODES FOR THE DOCUMENTATION BELOW;**

Regex-Based Data Assignment and Cleaning:

Step-by-Step:

1. Filter Out Rows Where 'Goods' is Numeric
    - The script first filters out rows where the Goods column contains only numeric values (integers or floats). This is done using the is_numeric function, which checks if a value is an instance of int or float.
2. Initialize New Columns
    - Two new columns, Normal Tariff and Preferential Tariff, are initialized with None values. These columns will store the processed data based on the rules.
3. Apply Rules to Assign Data
    - The apply_rules function processes each row of the Goods column and assigns values to the Normal Tariff and Preferential Tariff columns based on the following rules:
      a. Rule 2: If the Goods column contains "FreeFree", both Normal Tariff and Preferential Tariff are set to "Free".
      b. Rule 4: If the Goods column starts with "CA", "LDC", "RCEP", or "CPT", the entire content of the cell is assigned to the Preferential Tariff column.
      c. Rule 5: If the Goods column contains "SeeBelow", the Preferential Tariff column is set to "SeeBelow".
      d. Rule 7: If the Goods column contains "Free Free", both Normal Tariff and Preferential Tariff are set to "Free".
      e. Rule 1: If a digit appears between words in the Goods column, the digit is assigned to the Normal Tariff column, and the text following the digit is assigned to the Preferential Tariff column.
      f. Rule 3: If a digit is followed by " Free" in the Goods column, the digit is assigned to the Normal Tariff column, and "Free" is assigned to the Preferential Tariff column.
      g. Rule 6: If a word in the Goods column ends with a digit followed by "Free", the digit is assigned to the Normal Tariff column, and "Free" is assigned to the Preferential Tariff column.
4. Clean the Normal Tariff Column

    - The clean_normal_tariff function ensures that any number greater than 10 in the Normal Tariff column is replaced with an empty string. Numbers less than or equal to 10 and text values (e.g., "Free") are left unchanged.
5. Clean the Preferential Tariff Column
    - The clean_preferential_tariff function ensures that only values starting with "CA", "LDC", "RCEP", "CPT", or "Free" are kept in the Preferential Tariff column. All other values are replaced with an empty string.
6. Clean the Goods Column
    - The clean_goods function removes standalone occurrences of "o", "g", and "kg" from the Goods column. It also removes unnecessary white spaces (leading, trailing, and multiple spaces between words).
7. Save the Final Output
    - The cleaned and processed DataFrame is saved to a new CSV file named "new_zealand_tarrifs.csv".

Key Functions

1. is_numeric(value)
    - Input: value (any type)
    - Output: Boolean (True if the value is an instance of int or float, False otherwise)
    - Description: Checks if a value is numeric (integer or float).
2. apply_rules(row)
    - Input: row (pandas Series)
    - Output: row (pandas Series with updated Normal Tariff and Preferential Tariff columns)
    - Description: Applies regex-based rules to assign values to the Normal Tariff and Preferential Tariff columns.
3. clean_normal_tariff(value)
    - Input: value (any type)
    - Output: value (str or empty string)
    - Description: Cleans the Normal Tariff column by removing numbers greater than 10.
4. clean_preferential_tariff(value)
    - Input: value (any type)
    - Output: value (str or empty string)
    - Description: Cleans the Preferential Tariff column by keeping only values that start with "CA", "LDC", "RCEP", "CPT", or "Free".
5. clean_goods(value)
    - Input: value (str)
    - Output: value (str)
    - Description: Cleans the Goods column by removing standalone "o", "g", and "kg" and unnecessary white spaces.

Usage:

1. Load the dataset from the CSV file "combined_tables_improved.csv".

2. Filter out rows where the Goods column contains only numeric values.
3. Apply the rules to assign values to the Normal Tariff and Preferential Tariff columns.
4. Clean the Normal Tariff, Preferential Tariff, and Goods columns.
5. Save the cleaned and processed dataset to a new CSV file named "new_zealand_tarrifs.csv".