Project Documentation for Song Feedback and Genre Extraction Process

Overview

This document provides a step-by-step guide on using the scripts to collect song feedback, profile links, and genre data for various songs. It explains the order of script execution and the required modifications to each script for specific songs. The scripts are designed to process data for each song separately to avoid issues caused by handling large volumes of data.

Step 1: Run `update_feedbacks.py`

1. Execute the `update_feedbacks.py` script. This script will:
    - Save feedback details for all songs in a CSV file format.
    - The file will include:
        - Feedback content
        - Feedback giver's name
        - Song name (to distinguish each song's feedback)
        - Feedback type (e.g., 'promise to share' marked as Success; 'Feedback in red' marked as Fail).
2. Output: A CSV file (**`song_feedback.csv`**) with feedback data for all songs.

Step 2: Run `LinksProfiles.py`

Before Running `LinksProfiles.py`

Make the following modifications to personalize the script for each song:

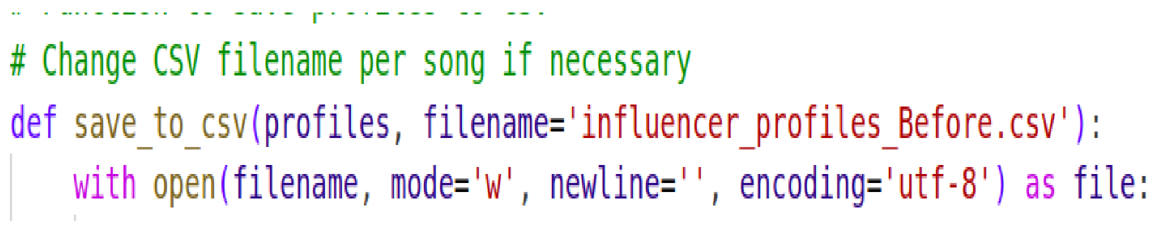
Modifications

1. **Update the Song Name in Code**

```
# Finding the specific song and clicking it for feedback scraping
obsessed_element = WebDriverWait(driver, 10).until(
    EC.element_to_be_clickable((By.XPATH, "//span[text()='Before']"))
    #[Before,BEFORE,Obsessed,INFINITE,Searching,Alternate Universes,Infinite]
)
obsessed_element.click()
```

   - Replace the existing song name with the desired song name (e.g., Before, Obsessed, INFINITE, etc.).
   - **e.g** for song INFINITE you must write like this "EC.element_to_be_clickable((By.XPATH, "//span[text()=INFINITE" and like this for other song names.
   - **Tip:** Use `Ctrl + F` to quickly locate this line in the code and write " EC.element_to_be_clickable((By.XPATH, "//span[text()=" to find this part quickly.

2. **Update the CSV File Name for Each Song**



```
# Change CSV filename per song if necessary
def save_to_csv(profiles, filename='influencer_profiles_Before.csv'):
    with open(filename, mode='w', newline='', encoding='utf-8') as file:
```

   - Replace the existing file name with a unique name for each song.
   - **e.g** for song INFINITE you can save it" **influencer_profiles_INFINITE.csv**" file and so on for other songs.

Run the Script

- After making the necessary changes for a song, run the code. Repeat the above steps for each song you need to process.
- This will generate separate CSV files for each song's profile links.

<u>Note:</u> I attempted to structure the song name changes in this code similarly to `update_feedbacks.py`, which processed data for all songs. However, due to the large volume of data, the browser could not execute all commands. I also tried using Scrapy, but due to the extensive data, errors occurred. To handle this efficiently, it's

recommended to scrape data separately for each song—this is faster, better, and more manageable.

After obtaining all links from running `LinksProfiles.py`. You need to run the `Genres.py` code.

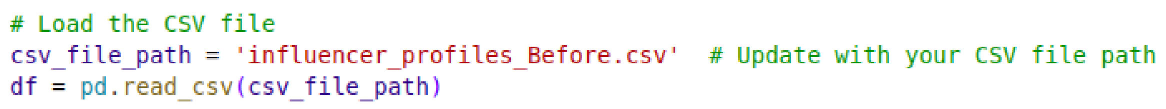
Step 3: Run `Genres.py`

Before Running `Genres.py`

Once you have collected the profile links for each song in Step 2, proceed to modify and run `Genres.py` for each song.

Modifications

1. **Update the Input CSV File Path**

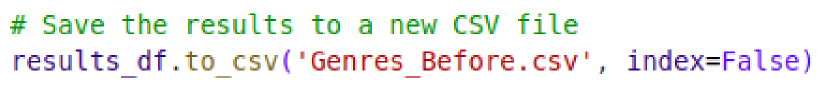


```
# Load the CSV file
csv_file_path = 'influencer_profiles_Before.csv'  # Update with your CSV file path
df = pd.read_csv(csv_file_path)
```

   Replace the existing file name with the corresponding CSV file name created in Step 2 for each song.

   e.g if in "**LinkesProfiles.py**" for **INFINITE** you saved the csv like "**influencer_profiles_INFINITE.csv**" you need write "csv_file_path = influencer_profiles_INFINITE.csv**"** and for others like this.

2. **Update the Output CSV File Name**



```
# Save the results to a new CSV file
results_df.to_csv('Genres_Before.csv', index=False)
```

   - **e.g for** song **INFINITE** you can save it "**Genres_INFINITE**"
   - Replace the existing file name with a unique name for each song's output file.

Run the Script

- Execute the modified script for each song to obtain separate genre data files for each feedback writer.

Optional: Generate Song Names List

If needed, the `song_name.py` script can generate a list of song names, either as printed output or saved in a CSV file, to assist you in managing and referencing song names for the modifications above.

Summary

By following this process, you will:

1. Run `update_feedbacks.py` to collect initial feedback data.
2. Run `LinksProfiles.py` for each song, making necessary name updates, to collect profile links.
3. Run `Genres.py` for each song to extract genre data based on the collected profiles.

Each song will have individual CSV files to keep data organized and manageable. This approach optimizes performance and minimizes errors associated with handling large datasets in a single execution.