README

Project Code

https://github.com/jinge2019/SI507-Final-Project-Assignment

Data Sources

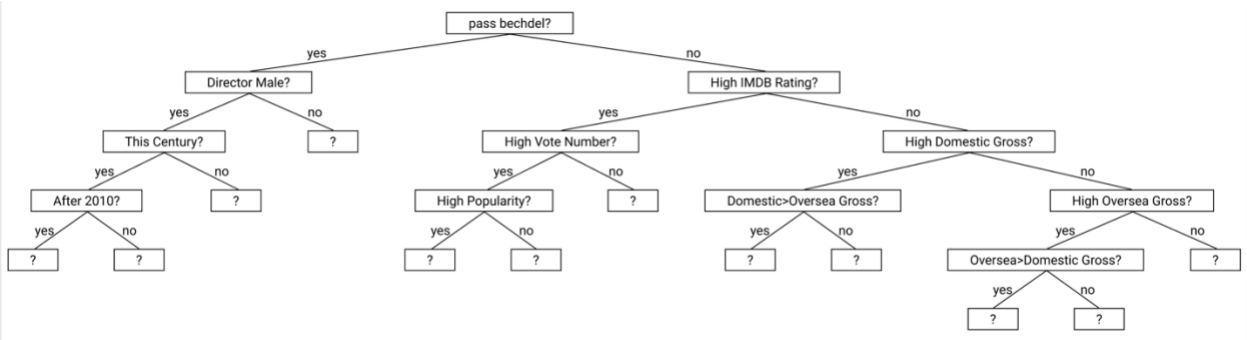
- TMDB API
    - I found director information for 200 movies here. This is done by first, locate the movies with IMDB ids and find the movie's internal ids, then get their credit lists with the internal ids and extract director data.
    - https://developers.themoviedb.org/3/find/find-by-id
    - Web API you haven't used before that requires API key or HTTP Basic authorization
- Bechdel Test movie API
    a. I used this API to find about 9000 movies and their Bechdel Test result info.
    b. https://bechdeltest.com/api/v1/doc
    c. Web API you haven't used before that requires no authorization
- IMDb API
    a. I only used the 200 top box office movie list in this API.
    b. https://imdb-api.com/api#BoxOfficeAllTime-header
    c. Web API you haven't used before that requires API key or HTTP Basic authorization
- OMDb API
    a. I got movies' IMDB ratings and votes numbers with this API.
    b. http://www.omdbapi.com/
    c. Web API you haven't used before that requires API key or HTTP Basic authorization

The movie list is built upon the 200 top box office movie list. I found other data from the other three data sources and added them to the 200 movie list. The data from 4 APIs are combined by shared IMDB ids. The data is processed in file data_processing.py.

The final 200 movie info is saved into a file, movie.txt.

Data Structure

I build a tree with class Node. The tree contains a list of question used to filter the 200 movies. The tree looks like this.



The value of each tree node is a list of numbers which are used to locate movie info. I converted the tree into a list and saved it in tree.json.

Interaction and Presentation Options

- Part 1:
    - Filter movie: The users will be asked to answer yes or no to a series of questions. Depending on their answers, they will go through any branch of the given question tree.
    - Stop filtering: The filtering process will stop if the user reach to the leaf of a branch, or if the user inputs “stop” before they finish the whole line. Once the filtering process stops, the program will return a list of satisfied movies and the questions that user went through.
    - Ask for details: The user can have a look at more detailed view of the movies by answering yes to the next question. The program will present a table of the same movies with 7 columns in it.
    - Sort the table: The user will be asked if they want to sort the table by year or IMDB rating. Answering yes will make the program sort the table and display the updated table.
    - Start over or exit: At the end of this part, the user can choose to either go over the entire process again, or say no to exit part 1.

- Part 2:
    - Display Bechdel Test result: The program will first display all movies and their Bechdel Test result if they have any. The movies are divided into two groups, the pass one and the fail one. The fail one contains reasons for their failing.
    - Add Bechdel Test result to the list: There are 8 movies that don't have Bechdel results. (There aren't any in the Bechdel Test Movie API). The users will be asked if they want to add Bechdel Test result to one of the 8 movies.
    - Determine if a movie passes Bechdel Test: If the user chooses to add Bechdel Test result to one of the movies, they will be asked to answer three questions about the movie. The program will use the answers to determine if a movie passes the test. Only when a user answers yes-yes-no, will a movie be claimed as pass the test. Once the user answers all three questions, the program will present a updated table of movies with the new data added to it.
    - Repeat or exit: The users will be asked if they want to do the same to another movie. Answering yes will make the program execute add Bechdel Test result again, otherwise the program will stop and reach the end.

Demo Link

https://drive.google.com/file/d/1FqJ4SmMXh8q0t8OJbhk8OIlpLnmR6Ty-/view?usp=sharing