Project Proposal

Summary of the project

Basically, my final project result will be a movie recommending program that accesses movie data from open movie database and Reddit, organizes that data into a tree, and then asks users questions about the movie's year, genre, comments (keywords of the people's comments, etc.) through an interface like a bot until it provides a set of recommendations of movies that meet the user's requirements.

Describing your data sources

1. (point 4)
   From the Open Movie Database, I will get data of movies' title, year, rate, release date, runtime, genre, director, actors, language, etc. The data is obtained from https://www.omdbapi.com/ which is a Web API that requires API key.
2. (point 6)
   From Reddit, I will get the data that contains answers of web users for questions about movies, and also answers below questions of movie recommendations (e.g. "What is the movie that most impressed you?" and the movie names in the answers). The data is obtained from https://www.reddit.com/ which is a Web API you haven't used before that requires OAuth.

How you plan to use them (processing, interaction, and presentation)

For data processing, I will add a good/bad feature for each moving from the ratings from open movie database. I will process the raw data from Reddit and make a table that contains the keywords of comments for each movie. Then I will join the two tables from the two databases together by the titles of movies. Finally, I will create a tree that enables searching movies by their features.

For interaction, I will create an HTML interface like a bot that asks the user questions, and the user texts the answer until it provides a set of recommendations of movies that meet the user's options. I will create it by Flask App.

For presentation, still on the HTML interface, after the program makes the suggestions for the movies, I will prompt the user to choose 4 different data visualization options: (1) a graph that contains the number of the final recommendation (only 1) in the middle and the keywords of the users' requirements (e.g. 2018, car, rating > 6, positive, interesting, action). (2) A sequence of movies names from top to the bottom. Top ones have bigger front sizes which means they are more likely to satisfy the user's requirements. (3) An animation for the search route (like a tree) which shows how the program finally find the movie. (4) describe the features and the recommended movie via text: "Here is my recommendation: *movie's name*, which is *user requirement 1, 2, 3,….* You can also have a look at these movies: *movies' name 1, 2, 3,….*"