CPS842 – Movie Recommender project

Report:

This movie recommender system is made using Flask and written in Python Code. There are some 1 time use code in the program such as the parseCSV.py file that is only used to grab a csv movie file and make a json file for faster reading in python. The main file that should be ran is the run.py file which will start the localhost site. You can get to the site after running by going onto a browser and typing in “localhost:5000”.

In this project, the number of movies is cut from the original movie file of 7000+ to only 100 movies but as it works on 100 movies it would still work.

The main bulk of the used files is in the ‘template’ folder, the ‘\_\_init\_\_.py’ file only initiates a couple of configurations, the database, and the application. The ‘forms.py’ is used to create the forms in Flask by using a package called ‘wtforms’, this package just makes it easier to create forms. There are 3 forms that are required in this website, the registration form, login form, and ratings form. The ‘models.py’ file is used to create specific tables in the database, the only 3 tables created is the users table that contains the registration information, the movieRatings table which contains the ratings of different people on a movie, and finally the movies table which contains all the movies.

The largest file is the ‘routes.py’ file which is where the bulk of the work is made. It consists of all the webpages that can be accessed. These pages are normal functions in python except that they are all directly below ‘@app.route’. There are a couple of global variables which contains the current user and the sim scores. People cannot access anything until they are logged in so when they try clicking on an option, they will be taken to the login page.

The register page will ask for information and input them into the User table where the login page will check. Each of the forms have restrictions and will give errors if not filled in correctly. Flash messages will be shown above the page.

When a new user is logged in, they have 2 options, to rate movies or get a recommendation list. A new user will not be able to get a recommendation until they rated some movies so if they clicked on it then the page will redirect to the rating page. The rating page contains a search bar and only when a user types in words will there be options as the search is dynamic and will search for words or letter in the movie title, the genre, and the director. If the user had already made some ratings to some movies the value will be shown in the rating form box. The rating page is connected to the movierating table which will change depending on the rating and will also change the average rating.

The recommendations page when clicked will generate movie recommendations using item-based collaborative filtering algorithm. This page is made so that there are a maximum of 30 recommendations per page, user can change pages to see more. The items are recommended by highest predicted ratings first and will continue down. The function used here is called ‘itemBasedCF()’ which starts the comparisons between the movies. This function calls ‘comparedToMovie’ function which compares that movie to other movies, as this is item-based it takes the average of the movies by only using ratings that intersect. Essentially, the first function takes movie 1 and then calls the second function that compares all movies after movie 1 to movie 1, this will also fill the sim global variable dictionary with the key-value pairs being (movie\_id\_x,movie\_id\_y) : sim score. Both the functions use other functions like ‘movieRateList’is used to grab all the ratings and sort them to match the next function, ‘getAllRaters’ function gets all the raters of movie x and y into a list, ‘getIntersectingRaters’ gets all the users that rated both movies which will be used to calculate the predicted rating.

The ‘comparedToMovie’ function only records sim score that are higher than 0.7 however this can be changed to something higher. The scores are then put into the ‘sim’ dictionary. ‘itemBasedCF()’ then writes out a json file of the sim scores called ‘similarity\_matrix.json’.

Finally the ‘recommend()’ function loops through all the movies that the current user did not rate, then for that unrated movie it checks if the sim score is recorded. If there is a record of the sim score then it will go through the movies to find the id which will get the name of the movie. Then it will take that sim score and multiple it with the current user rating of that movie. The calculations are done to get the predicted rating which is stored in a dictionary.

The recommendation page checks if a global variable called ‘new\_ratings” is true or not. If it is true it means that the user just rated something and everything must be recalculated, the final recommended list is stored in a json file in the ‘saved’ folder so that the system does not need to re-calculate every time and instead can just read the json file to put out the old recommendations.

The user rating data is stored in user\_movie\_ratings.json.