**CSC 385 Final Project Algorithm Explanation and Analysis**

Explanation:

The data structure I used to store movies.dat data is a HashMap in form of

HashMap<Integer, String>.

The data structure I used to store ratings.dat data is a nested HashMap in form of

HashMap<Integer, HashMap<Integer, Integer>>.

The algorithm I used to compute the top five recommendations is first find the Cosine Similarity values for all the movies by looping through all the movies and calculate the values and store the values in a 2D array. Then I calculate the predicted rating for movies not rated by the users by looping through each user and find the predicted ratings for unrated movies by using the algorithm in the specification. Then I loop through each user and find the top five unrated movies and output the results to a text file.

Time Efficiency Analysis:

M is the number of movies.

U is the number of users.

Reading, parsing, and storing the movies.dat data take O(M) time.

Reading, parsing, and storing the ratings.dat data take most O(UM) time, but most likely less than that because not every user rated every movies.

In the MoviesSimilarity class, it calls on the MoviesAndRatingParsing class, which takes O(UM) for that part as stated previous. Finding the similarity in the MoviesSimilarity class, the outer most loop is looping M times to go through all the movies. The first nested loop inside the outer most loop is looping through all users so it’s U times, and in total O(UM) times. The second loop inside the outer most loop is looping through at most half of all the movies again and another nested loop inside that loops through all users which in total take O((1/2M)MU) time and another loops through all the movies again to calculate the similarity which total for that takes O(MMM) time. The HashMap and array retrieval inside all the loops takes O(1). The last nested loop takes O(1/2MM) to copy the lower half of the array to upper half.

In the MoviesRatingPrediction class, it first calls on the MoviesAndRatingParsing and MoviesSimilarity class, run times for those explained previously. In the ratingPrediction method, the outer loop takes U times to go through all users. The first nested loop through all movies which total takes O(UM). The second nested loop loops through all the unrated movies which at most take M times, so total is O(UM). Then another nested inside that loops through all the rated movies which at most takes M times, so total for that is O(UMM). The HashMap retrieval inside all the loops takes O(1).

In the TopFiveRecommendations class, it calls on the MoviesRatingPrediction class, run time explained for that class previously. For the rest of the TopFiveRecommendations class, the outer loop go through all users takes U time, inside are 5 loops that loop through all the unrated movies which is at most 5M times, so total is O(5UM)=O(UM).

In the TopFiveOutput class, it calls on TopFiveRecommendation, MovieAndRatingParsing, and MovieRatingPrediction classes. For the rest of the TopFiveOutput class, it loops through each user to write the output to a text file takes U time, HashMap retrievals takes O(1), so total O(U). Accessing the external file is unknown.

So the dominant term in the program is O(UMM), the program is of O(UMM).