Aaron Gold
Movies Part 2

Github: https://github.com/aharonhillel/Movies_2
Code Climate: https://codeclimate.com/github/aharonhillel/Movies_2

The Algorithm. A description of your prediction algorithm and what you think are its advantages and drawbacks.

The algorithm: My algorithm is based off of similar users and their rating of a said movie. For example, in order to figure out what a user would rank a movie, one would pass in a user as well as the movie. First, the algorithm would take this information and find similar users (similar users ranked movies within one 60% of the time). Following that, it would create an array of all of the similar users that watched the movie you are looking for and what they rated it. Finally it would calculate the average rating and use that as the prediction. However, my code also takes into account if there are only a few people (less then 5) that came up as similar and ranked the movie. The reason for considering this is because say only one person is similar and ranked the movie then they would have an overarching sway on the predicted rating. Thus is there are less than 5 similar users that rated the movie then it calculates the overall average of everyone's ratings of the movie. Therefore, I feel like one of the advantages of my algorithm is that it bases the predicted rating on what similar users rated the movie and thus should be more accurate. However, there is a huge drawback which is time complexity. For every user, my algorithm has to run through the entire database (which is often 10000 users) to search for similar users. Thus it runs 100000 * 100000 times which is extremely slow. This is the reasoning I have a print statement in line 22 of my compare class so that I can see the program is running. I look forward to one day thinking of a faster algorithm!

The Analysis:
My algorithm consistently guessed approximately 30% of the ratings correct. Although this is not as high as I had hoped since my algorithm takes into account similar users I look forward to continuing to adjust it. Below are some of my results.

```
        u1.base/u1.test
                correct: 5813
                incorrect: 11677
                Within one 2510

        u2.base/u2.test
                Current Time : 2017-02-01 11:20:55 -0500
                correct: 5890
                incorrect: 11675
                Within one 2435
                Current Time : 2017-02-01 11:40:17 -0400

        u3.base/u3.test
                correct: 6062
                Within one 2531
                incorrect: 11407
```

My code outputs what number (every thousand it is at). This is just to see that the program is running since the time complexity is not what I had hoped but I understand why that is the case.

My current time for the test/base inputs was approximately 20 minutes. As obvious, this is a horrific time complexity that I am thinking on how to improve. As mentioned beforehand, the reason for this is that it has to query the entire database every single time. Thus this would not scale well to millions of movies. However, this would work for a small number such as 100 movies.