CS-449 Systems for Data Science

Project Milestone 1

Due 19/03/2021

Author
DHAENE Arnaud

Professor
KERMARREC Anne-Marie

Teaching Assistants
LAVOIE Erick
XYGKIS Athanasios

Global Average Deviation Questions

3.1 Global average rating

The global average rating $\bar{r}_{\bullet,\bullet}$ is calculated as follows, with R being the set of all ratings in the dataset:

$$\bar{r}_{\bullet,\bullet} = \frac{\sum_{r_{u,i}} r_{u,i}}{|R|} \tag{3.1}$$

The result, when rounded to 4 decimal places, is $\bar{r}_{\bullet,\bullet} = 3.5300$. This indicates that the global average rating does not coincide with the middle of the rating scale (3 from the scale {1,2,3,4,5}). The ratings are higher on average by approximately half a rating point (0.5).

3.2 Average rating for each user

The average rating for each user is calculated as follows:

$$\bar{r}_{u,\bullet} = \frac{\sum_{i \in I(u)} r_{u,i}}{|I(u)|} \tag{3.2}$$

The minimum and maximum user average ratings — rounded to 4 decimal places — are 1.4920 and 4.8670 respectively. These results indicate that not *all* users rate, on average, close to the global average. The ratio of users that rate close to the global average (a deviation smaller than 0.5 is considered close) is 74.7%, which indicates that *most* users rate, on average, close to the global average.

3.3 Average rating for each item

The average rating for each user is calculated as follows:

$$\bar{r}_{\bullet,i} = \frac{\sum_{u \in U(i)} r_{u,i}}{|U(i)|} \tag{3.3}$$

The minimum and maximum user average ratings are 1.0 and 5.0 respectively. This indicates that certain movies may have been rated only once with a very good (respectively bad) score. Evidently, these results indicate that not *all* items are rated, on average, close to the global average. The rounded ratio of items that are rated close to the global average (a deviation smaller than 0.5 is considered close) is 49.0%, which indicates that *most* items are not rated, on average, close to the global average.

3.4 Prediction accuracy

The prediction accuracies (as the MAE on the `ml-100k/u1.test` dataset) rounded to the fourth decimal point for the global, per user average, per item average, and baseline methods are reported in the table below.

Method	**MAE**
Global average	0.9680
User average	0.8502
Item average	0.8276
Baseline	0.7681

Table 1: MAE for the different tested prediction methods.

It can be observed in Table 1 that the baseline method is the most accurate method among the ones tested in the scope of this exercise. Furthermore, it seems that predicting the rating for a (user, item) pair is approximately 0.0226 rating points more accurate when using the item's average rating as opposed to the user's average rating. Perhaps this could stem from rating behavior, as users tend to rate movies either when they really liked or disliked the movie. My hypothesis is that such a behavior would lead to good (resp. bad) movies being rated with good (resp. bad) scores that are distributed quasi-normally. On the other hand, users are less likely to rate movies that are 'intermediate', which makes the user average a worse predictor for an unseen movie.

3.5 Computation time

Reported in a table below are the minimum, maximum, average, and standard-deviation of ten measurements of the computation time for the different prediction methods.

Method	**Minimum**	**Maximum**	**Average**	**Standard deviation**
Global average	80761.9550	108741.9160	92624.9019	9241.8807
User average	412263.7730	544233.2860	467759.0555	41315.4470
Item average	291834.8650	436771.0950	375244.4204	41039.0840
Baseline	1313249.2210	1615441.3110	1493768.4645	109245.7281

Table 2: Minimum, maximum, average, and standard deviation of the ten measurements of time for computing the predictions for each method (in microseconds).

The most expensive method to compute is the baseline method, as can be observed in table 2. The ratio between the average time for computing the baseline and the average time for computing the global average is 16.1271.

The technical specifications of my personal computer are the following:

- **Model**: MacBook Pro (Retina, 13-inch, Early 2015)
- **OS**: macOS Big Sur version 11.1
- **Processor**: 2.9 GHz Dual-Core Intel Core i5
- **Memory**: 16 GB 1867 MHz DDR3
- **Graphics**: Intel Iris Graphics 6100 1536 MB
- **Language**: English

Recommendation Questions

4.1 Personal top 5 recommendations

My personal top 5 recommendations using the baseline predictor are:

Movie	**ID**	**Predicted rating**
Great Day in Harlem	814	5.0
They Made Me a Criminal (1939)	1122	5.0
Prefontaine (1997)	1189	5.0
Marlene Dietrich: Shadow and Light (1996)	1201	5.0
Star Kid (1997)	1293	5.0

Table 3: My personal recommendations.

I have never seen (or heard of) any of these movies. After inspection on IMDB, they actually seem quite appealing, I might even watch them. However, it does seem as if these movies are not very popular, and their perfect predicted rating of 5.0 indicates they might not have been rated by many users.

4.2 Favour more popular movies

I created a bonus predictor function that penalizes movies that are unpopular (i.e. that have not been rated by many people). Using the ratings per item, with m being the minimum number of ratings—1 in this dataset, M being the maximum number of ratings—584 in this dataset, and $|U(i)|$ the number of ratings for the item in question i:

$$f = \frac{|U(i)| - m}{M - m} \tag{4.1}$$

I call f the popularity fraction. When put into a sigmoid-like function p called the unpopularity penalty, I am able to penalize predicted ratings of movies with few ratings while keeping the prediction score of those with many ratings almost identical.

$$p = \frac{1}{5(1 + \exp(10f - 1))} \tag{4.2}$$

Below is a graph displaying the relationship between f and p. This indicates that the least rated movies will be penalized by around 0.14 rating score points, whereas the most rated movies will not be penalized at all.

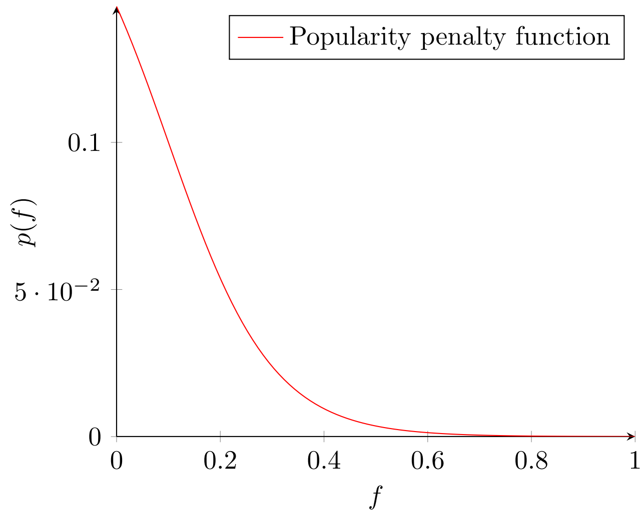

Figure 1: Unpopularity penalty function

My updated list of recommendations is captured in the following Table:

Movie	**ID**	**Predicted rating**
Prefontaine (1997)	1189	4.8551
Star Kid (1997)	1293	4.8551
Saint of Fort Washington	1468	4.8545
Santa with Muscles (1996)	1500	4.8545
Great Day in Harlem	814	4.8540

Table 4: My personal recommendations with unpopularity penalization.

The above list of movies includes 2 movies I have seen but did not rate in the `personal.csv` file that I quite liked. The ones I haven't seen seem like good recommendations when inspecting their IMDB pages, except for **Star Kid (1997)**, which looks like a pretty bad movie in my opinion.

Please note that the fixed parameters 5, 10, and 1 in Equation 4.2 were chosen by way of trial and error due to the limited scope of this exercise. In fact, when modifying the parameters in a specific manner, I was suggested very famous movies that I knew I did not really like. There are evidently some very important biases involved in this selection, some of which (as there are many others) are (1) the movies I have heard of, (2) the movies that were rated in this dataset, and (3) the ratings I gave to movies I had not recently seen.

To go further in this initiative, one should optimize these popularity penalty parameters with objective performance metrics. There are different ways to do this. While I didn't venture myself in such endeavours, I did test my modified predictor on the train/test split from Part 3.1.4, and I am able to obtain an MAE that is slightly better than baseline, 0.7667.