Systems for Data-science M2

Gengler Damien 272798

April 2021

1 Baseline: Prediction based on Global Average Deviation

- Question 2.2.1 :

  The prediction accuracy of equation 3 is 0.7478. The prediction accuracy is better than the baseline, the difference between the Adjusted Cosine similarity and the baseline is -0.0191.

- Question 2.2.2 :

  We give in Equation (1) the computation for the Jaccard Similarity Coefficient

  $$s_{u,v}^{Jaccard} = \frac{|U \bigcap V|}{|U| + |V| - |U \bigcup V|} \tag{1}$$

  where U is the set of items rated by u and V the set of items rated by v.

  The prediction accuracy using the Jaccard Similarity coefficient gives a prediction MAE of 0.7623. The difference between the Jaccard Coefficient and the Adjusted Cosine Similarity is 0.0145 which means that the Jaccard coefficient is worst than the Adjusted Cosine Similarity.

- Question 2.2.3 :

  If every user rated at least one item in common with every other user. In the worst case number of $s_{u,v}$ computation is $U * (U - 1)$. For the case of the ml-100k dataset, that would represent 888306 similarity computations.

- Question 2.2.4 :

  The min, max, average and standard deviation of the number of multiplications for each similarity computed are reported in Table 1

- Question 2.2.5 :

  In order to store all possible similarities we would need :

  $|U| * (|U| - 1) * \#bits_{persimilarity}$ bits

  In order to store all the non-zero similarities on the ml-100k dataset we would need 6536592 bytes.

Statistics	Number of multiplications
min	0
max	332
avg	12.0
std	21.56

Table 1: Min, Max, Average and Standard deviations of the number of multiplication for each Similarity computed

- Question 2.2.6 :

  The min, max, average and standard deviation of the time required to compute predictions are reported in Table 2

  Here the average time to compute the test predictions is around 81454 milliseconds. In milestone 1 we found an average time of 1380 milliseconds. We see that our personalized method is 80 longer than the baseline method used in milestone 1. This makes sense because we compute much more complicated and personalized similarity metric.

Statistics	time (in microseconds)
min	76514
max	93032
avg	81454.4
std	418.97

Table 2: Min, Max, Average and Standard deviations of the time to compute predictions

- Question 2.2.7 : The min, max, average and standard deviation of the number of the time required to compute all similarities are reported in Table 3. The average time to compute one $s_{u,v}$ is 0.02885 microseconds. And the ration between the time to compute the similarities and the time to predict is 0.3145. Which means we spend around 30% of the time to compute the similarities. So the computation of similarities is indeed quite significant.

Statistics	time (in microseconds)
min	24549
max	29301
avg	25624.4
std	1841.6

Table 3: Min, Max, Average and Standard deviations of the time to compute similarities

- Question 3.1.1 :

  For k = 10, 30 and 50 we have worst results than with all datapoint because we over specify our results. After 100 we beat the baseline and get better results. With K = 943 we get the same results as using the whole data in cosine similarity.

  The lowest K with a better MAE than baseline is k = 100 and the difference with baseline MAE is -0.01076.

k	MAE
10	0.8407
30	0.7914
50	0.7749
100	0.7561
200	0.7484
300	0.7469
400	0.7473
800	0.7472
943	0.7478

Table 4: MAEs obtained with different k

- Question 3.1.2 :

  The formula to compute the minimum number of bytes required to store the top k similarities for each users is $|U| * k * \#bits_{persimilarity}$ bits. Given the fact that we store similarity values in 8 bytes, we thus need $|U| * k * 8$ bytes.

k	MAE
10	75440
30	226320
50	377200
100	754400
200	1508800
300	2263200
400	3017600
800	6035200
943	7113992

Table 5: Size in bytes to store the top K similarities for ml-100k

- Question 3.1.3 :

  I have 8Go of RAM, with this amount I could store the top 100 similarities of 3534 users in memory.

- Question 3.1.4 :

  No because in order to keep the exact top k similarities, we need to compute all of them and then only keep the k best. So all similarities will need to be computed at some point.

- Question 3.1.5 :

  My top 5 movies predicted with k=30.

  - Mirror Has Two Faces, 5.0
  - Good Will Hunting (1997), 5.0
  - Fear of a Black Hat (1993), 5.0
  - Matilda (1996), 5.0
  - Wings of Desire (1987), 5.0

  and for k=300

  - Horseman on the Roof, 5.0
  - Maya Lin: A Strong Clear Vision (1994), 5.0
  - So Dear to My Heart (1949), 5.0
  - Great Day in Harlem, 5.0
  - Prefontaine (1997), 5.0

  Results with k=30 seem much more accurate and give better predictions than results with k=300. Some of the movies predicted for k=30 are movies that I would like to watch such as Good Will Hunting. We can see that given my personal ratings, we have a pretty good recommandation.

  Comparing with the results of milestone 1, we see that the results for k=30 are far from it but when taking top 300, we do find 3 movies that also appeared in Milestone 1 predictions and that we assumed were niche movies. So this shows that for k=300 we are back to the very wide and not so resilient prediction similar to milestone 1 as we get not specific enough.

2 Notes

In order to run the computations faster on my computer I increased the heap memory size during compilation. As this was discussed on the forum I think that it was allowed but I prefer to let you know in advance. Here's the command I use to run with extended heapsize :

⇒ sbt **-J-Xmx2G** "runMain recommend.Recommender –data data/ml-100k/u.data –personal data/personal.csv –json recommendations.json"