École Polytechnique Fédérale de Lausanne

CS 449 : Systems for Data Science

Milestone 2 Report

Karim Hadidane
SCIPER: 271018

Part 2

Question 2.2.1 :

Compute the prediction accuracy (MAE on ml-100k/u1.test) of (Eq. 3). Report the result. Compute the difference between the Adjusted Cosine similarity and the baseline (cosine baseline). Is the prediction accuracy better or worst than the baseline (Eq. 5 from Milestone 1)? (If you are reusing some of your code of Milestone 1, use your previous baseline MAE. Otherwise, use a baseline MAE of 0.7669.)

Answer :
Using the adjusted cosine similarity, we got a MAE of **0.74776**.
The difference between the adjusted cosine similarity method MAE and baseline MAE (0.7669) is **-0.01913**.
The prediction accuracy is better than the baseline method implemented in the previous milestone.

Question 2.2.2 :

Implement the Jaccard Coefficient. Provide the mathematical formulation of your similarity metric in your report. Compute the prediction accuracy and report the result. Compute the difference between the Jaccard Coefficient and the Adjusted Cosine similarity (Eq. 1, jaccardcosine) Is the Jaccard Coefficient better or worst than Adjusted Cosine similarity?

Answer :
The Jaccard similarity metric between user u and user v is defined as :

$$J_{u,v} = \frac{\mid I(u) \cap I(v) \mid}{\mid I(u) \cup I(v) \mid} = \frac{\mid I(u) \cap I(v)}{\mid I(u) \mid + \mid I(v) \mid - \mid I(u) \cap I(v) \mid}$$

where $\mid I(u) \mid$ denotes the number of items rated by user u.

The prediction done with Jaccard similarity metric yields MAE of **0.76225**.
The difference between this method MAE and the adjusted cosine similarity MAE is **0.01448**. Hence the prediction is worse than Adjusted cosine similarity.

Question 2.2.3 :

In the worst case and for any dataset, how many $s_{u,v}$ have to be computed, as a function of the size of U (the set of users), if every user rated at least one item in common with every other users? (Provide the formula in your report) How many would that represent if that was the case in the 'ml- 100k' dataset? (Compute the answer in your code from the number of users in the input dataset)

Answer :
In the worst case, similarities between each distinct pair of users need to be computed, including self similarities. The number of $s_{u,v}$ that need to be computed is the number of combinations of u and v. More formally, this is given by the formula: $\binom{|U|}{2} + |U|$.
For the case of this dataset, we have **943** users and hence $\binom{943}{2} + 943 =$ **445096** $s_{u,v}$ computations are needed.

Question 2.2.4 :

Compute the minimum number of multiplications required for each possible $s_{u,v}$ on the ml-100k/u1.base (Tip: This is the number of common items, $|I(u) \cap I(v)|$, between u and v.) What are the min, max, average, and standard deviation of the number of multiplications? Report those in a table.

Answer :
The statistics for the minimum number of multiplications needed for each possible $s_{u,v}$ are given in the table below :

	min	max	average	standard deviation
Measure	0	685	12.2820	18.5203

Question 2.2.5 :

How much memory, as a function of the size of U, is required to store all possible $s_{u,v}$, both zero and non-zero values, assuming both require the same amount of memory? How many bytes are needed to store only the non-zero $s_{u,v}$ on the 'ml-100k' dataset, assuming each non-zero $s_{u,v}$ is stored as a double (64-bit floating point value)? (Tip: Do not include memory usage for intermediate computations, only for storing the final results.)

Answer :
The memory needed to store all possible similarities $s_{u,v}$ is proportional to the number of combinations of pair of users (u,v) , where v can be equal to u. This yields $\binom{|U|}{2} + |U| = O(|U|^2)$.
For the 'ml-100k' dataset, **3275888** bytes are needed to store the non-zero similarities.

Question 2.2.6 :

Measure the time required for computing predictions (with 1 Spark executor), including computing the similarities $s_{u,v}$. (Tip: If you compute the similarities in batch prior to predictions, include the time for computing them all. If you compute similarities on a by-need basis, possibly with caching, include the time for all those that were actually computed.) Provide the min, max, average, and standard-deviation over five measurements. Discuss in your report whether the average is higher than the previous methods you measured in Milestone 1 (Q.3.1.5)? If this is so, discuss why.

Answer :

The prediction computation time over 5 measurements statistics are given in the table below:

	min	max	average	standard deviation
Measure (μs)	20193000	22326000	21388600.0	817247.7225

The average time for the adjusted cosine similarity is higher than all previous methods (71100, 988200, 807700, 3551500 for global average, user average, item average and baseline respectively). This is because this

prediction scheme is more complex and it needs prior user similarity computations, which might be time consuming.

Question 2.2.7 :

Measure only the time for computing similarities (with 1 Spark executor). (Tip: If you compute the similarities in batch prior to predictions, include the time for computing them all. If you compute similarities on a by-need basis, possibly with caching, include the time for all those that were actually computed.) Provide the min, max, average and standard deviation over five measurements. What is the average time per $s_{u,v}$ in microseconds? On average, what is the ratio between the computation of similarities and the total time required to make predictions? Are the computation of similarities significant for predictions? (Tip: To lower your total running time, you can combine this measurement in with that of the previous question in the same runs.)

Answer :

The similarity computation time over 5 measurements statistics are given in the table below:

	min	max	average	standard deviation
Measure (μs)	15350000	17124000	16125600.0	648722.6217

The average time per $s_{u,v}$ is **36.2294** μ**s.**
The ratio between the computation of similarities and the total time to make predictions is **0.7539**. Therefore, this prediction scheme spends ∼75% of the time on computing the similarities between the users. Hence, the computation of similarities is significant for predictions.

Part 3

Question 3.1.1 :

What is the impact of varying k on the prediction accuracy? Provide the MAE (on ml-100k/u1.test) for k = 10, 30, 50, 100, 200, 300, 400, 800, 943. What is the lowest k such that the MAE is lower than for the baseline method (Eq. 5 of Milestone 1)? How much lower? (Use a baseline MAE of 0.7669, and compute lowest k baseline)

Answer :

The MAE obtained for different K values are listed in the table below:

K	MAE
10	0.8407
30	0.7914
50	0.7749
100	0.7561
200	0.7484
300	0.7469
400	0.7471
800	0.7475
943	0.7477

Hence the lowest k such that its corresponding MAE is lower than baseline is **K= 100**: Its MAE is lower by **0.0108**.

Question 3.1.2 :

What is the minimum number of bytes required, as a function of the size of U, to store only the k nearest similarity values for all possible users u, i.e. top k $s_{u,v}$ for every u, for all previous values of k(with the ml-100k dataset)? Assume an ideal implementation that stored only similarity values with a double (64-bit floating point value) and did not use extra memory for the containing data structures (this represents a lower bound on memory

usage). Provide the formula in the report. Compute the number of bytes for each value of k in your code.

Answer :

The number of bytes to store for each user depends on the value of K. For a given value K, we store K similarity values (K doubles) for each user, so K* 8 Bytes.
Hence the minimum number of bytes required to store the k nearest similarity values for all possible users is given by **|U| * K * 8 Bytes**.
The minimum number of bytes for the different K values for the 'ml-100k' dataset are given in the table below:

K	min number of bytes
10	75440
30	226320
50	377200
100	754400
200	1508800
300	2263200
400	3016888
800	5950040
943	6584736

Question 3.1.3 :

Provide the RAM available in your laptop. Given the lowest k you have provided in Q.3.1.1, what is the maximum number of users you could store in RAM? Only count the similarity values, and assume you were storing values in a simple sparse matrix implementation that used 3x the memory than what you have computed in the previous section (2 64-bit integers for indices and 1 double for similarity values)

Answer :
The RAM available on my laptop is 8 GB = 8589934592 B. With K=100, each user would take a space of k * 3 * 8B = 2400B. Hence the max number of users that can be stored is $\lfloor \frac{8589934592}{2400} \rfloor = 3579139$.

Question 3.1.4 :

Does varying k has an impact on the number of similarity values ($s_{u,v}$) to compute, to obtain the exact k nearest neighbours? If so, which? Provide the answer in your report.

Answer :
Varying K does not affect the number of similarities to be computed. This is because all similarities need to be computed to select the largest K similarities and hence the K nearest neighbours.

Question 3.1.5 :

Report your personal top 5 recommendations with the neighbourhood predictor (Eq. 3) with k= 30 and k= 300. How much do they differ between the two different values of k? How much do they differ from those of the previous Milestone?

Answer :
Top 5 recommendations with K=30:
63, "Santa Clause", 5.0
64, "Shawshank Redemption", 5.0
138, "D3: The Mighty Ducks (1996)", 5.0
145, "Lawnmower Man", 5.0
161, "Top Gun (1986)", 5.0

Top 5 recommendations with K=300:
119, "Maya Lin: A Strong Clear Vision (1994)", 5.0
691, "Dark City (1998)", 5.0
850, "Perfect Candidate", 5.0
1298, "Band Wagon", 5.0
1358, "The Deadly Cure (1996)", 5.0

The recommendations returned when using the neighbourhood predictor with K= 300 are all different from the recommendations with K=30.
When comparing with recommendations found in Milestone 1, using neighbourhood predictor with K=30 and K=300 yields totally different recommendations.