Methods of classification and dimensionality reduction - Report 1

April 12, 2022

1. Introduction

Statement of the problem. In this task we have to create a movie recommender system for our users. We have users who rated some movies. Of course, not every user rated every movie and it is our task to fill those gaps. So if one user hasn't seen one movie, we want to predict how he would like it.

For this purpose we build few algorithms using different methods of predicting. Of course different methods will give us different results (errors). Our task is to tune parameters of those methods and try to get the best possible ratings prediction.

Description of methods. In this problem, we use different methods which are subset of PCA methods. They are often used for dimensionality reduction and matrix factorization.

SVD1. This method gets a $n \times d$ dimensional matrix Z and approximate it by a different matrix $\tilde{Z}$. Since we want somehow $\tilde{Z}$ to maintain only "the most important" information from Z, then the rank of $\tilde{Z}$ is to be much smaller than rank of Z. Precisely, we want to find matrix $\tilde{Z}_r$ of rank r ($r < rank(Z)$ and r is a parameter), so that $\|Z - \tilde{Z}_r\|$ is small.

Using SVD decomposition $Z = U\Lambda^{\frac{1}{2}}V^T$ we construct $\tilde{Z}$ as

$$\tilde{Z}_r = U_r\Lambda_r^{\frac{1}{2}}V_r^T,$$

where Λ_r contains r biggest eigenvalues of Z and U_r, V_r contains only columns corresponding to those eigenvalues.

SVD2. It is an iterative method. We perform SVD1 on matrix Z, then on the result of first SVD1 and so on. The algorithm can be stopped after a fixed number of iterations or some stop condition can be established.

NMF. Similarly as in SVD1 the method obtain a $n \times d$ dimensional matrix Z and approximate it by $\tilde{Z}$. This time $\tilde{Z}$ is constructed as $\tilde{Z}_r = W_rH_r$, where W_r and H_r are matrices with non-negative elements (W_r has r columns and H_r has r rows). Precisely, we look for such W_r and H_r that $\|Z - W_rH_r\|^2$ is the smallest, where $\|A\|^2 = \sum_{i,j} A_{ij}^2$.

SGD. This method, similarly as previous ones want to estimate matrix Z with a product of matrices W and H, but not necessarily obtaining the whole matrix Z.

Let's assume that we have only some values of z_{ij} and let call those pairs (i, j) where we know the value of Z as I. We look for

$$\arg\min_{W,H} \sum_{(i,j)\in I} (z_{ij} - w_i^T h_j)^2 + \lambda(\|w_i^T\|^2 + \|h_j\|^2),$$

where h_j is j-th column of h, w_i^T is i-th row of W and $\lambda > 0$ is a parameter. So roughly speaking we look for W and H such that $Z \approx WH$ for elements known in Z, but also we want W and H to have quite small values (it gives us the part of sum with parameter λ).

It is an iterative method and work this way: set some W and H,

(1) sample one pair (i, j) from I,

(2) let $\tilde{w}_i^T := w_i^T - \eta \cdot \big(2(z_{ij} - w_i^T h_j)h_j + 2\lambda w_i^T\big)$ and $\tilde{h}_j := h_j - \eta \cdot \big(2(z_{ij} - w_i^T h_j)w_i^T + 2\lambda h_j\big)$,

(3) the rests of matrices W and H stay unchanged, so $\tilde{w}_k^T = w_k^T$ for $k \neq i$ and $\tilde{h}_l = h_l$ for $l \neq j$,

(4) take $W = \tilde{W}$ and $H = \tilde{H}$,

and repeat.

η is a parameter that tells us how big steps we want to do. The method stops after a certain number of steps, or it can be given a stop condition.

2. Implementation

Description of the data. Our data contains information 100837 ratings - exactly 610 users rated 9724 movies. The columns are: userId (integer), movieId (integer) and rating (integer), where userId is a unique user id and movieId is a unique movie id.

We keep this data in two-dimensional matrix of size $n \times d$ where n is the number of users and d is the number of movies. In element (i, j) we put the rate of the user i of the movie j. If the user i haven't rated the movie j we leave the element empty.

Performing methods. ??tutaj jakaś intuicja po co dzielić dane??

So to be able to evaluate the quality of the programs we split our data to two parts: train set and test set. The train set is used to build the programs. And the test set is used to evaluate how our programs work.

To give our programs enough information about every user we split the data so that the train set contain 90% of ratings of each user (and the test set the remaining ones). tutaj coś o tym, że będziemy to powtarzać??

Let call the matrix containing the data from the train set as $\boldsymbol{Z}$ and the matrix containing the data from the test set as $\boldsymbol{T}$.

??In SVD2 we make a correction – czy to tu

Quality of the system. Assume that our algorithm return a matrix $\boldsymbol{Z}'$. Then the quality of our programs is computed as **root-mean square error**

$$\mathsf{RMSE} = \sqrt{\frac{1}{|\mathcal{T}|} \sum_{(u,m)\in\mathcal{T}} \left(\boldsymbol{Z}'[u,m] - \boldsymbol{T}[u,m]\right)^2}$$

where $\mathcal{T}$ contains pairs (u, m) from test set.

Imputing the missing data. Since three of our methods (SVD1, SVD2 and NMF) are given a full matrix $\boldsymbol{Z}$ then they need the missing data to be imputed before performing.

We decided to impute the data in 5 different ways, we replace missing values with:

- 0 ,
- global mean,
- column means,
- row means,
- weighted row and column mean ($\alpha \cdot \mathsf{col\_mean} + (1-\alpha) \cdot \mathsf{row\_mean}$, where $\alpha > 0$ is a parameter).

We may expect that the closer to reality we impute the missing data, the better results we will obtain. tutaj przemyślenia na temat tego czemu niektóre metody działają lepiej i dlaczego

3. Parameters tuning and results

Before performing our methods and obtaining results we have to set some parameters.

First of all, all the methods need a parameter r, which is the rank of matrices in Z decomposition. SGD needs also learning rate and λ. And iteration methods need maximum of possible iterations or a stop condition.

What's more, for all of our methods we want to choose optimal α in the last imputation method data.

SVD1.

Optimizing r. For a start, let's consider only imputation methods that don't need estimation of α, so:

- putting 0 everywhere,
- putting global mean everywhere,
- putting column means,

- putting row means,
- putting weighted row and column mean $\frac{1}{2} \cdot \mathsf{col\_mean} + \frac{1}{2} \cdot \mathsf{row\_mean}$.

The last method is the weighted method for $\alpha = \frac{1}{2}$. The imputation methods mentioned above will be called *basic* in this report.

For these methods we only need to find optimal r. So for every basic imputation method and for every r from 1 to 100 we perform SVD1. Below, we present a graph showing results.

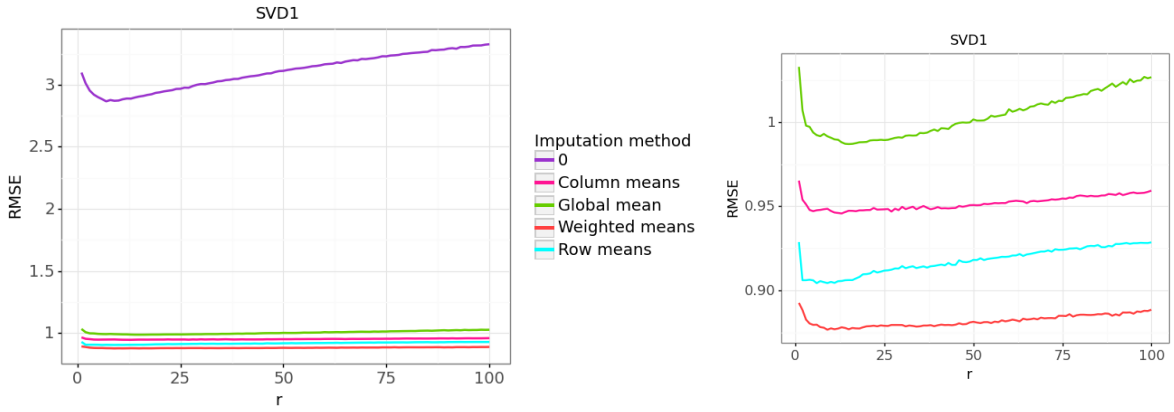

FIGURE 1. RMSE of SVD1 for basic imputation methods and $r = 1, \ldots, 100$

Let's denote that we perform it for only one split of the data into train and test set. This results can be different if we take different split.

jakieś wnioski

Of course we look for the lowest RMSE obtained for each imputation method and the optimal r. So below we present a table containing these information.

	0	column means	global mean	weighted means	row means
r	7	13	15	9	6
RMSE	2.8660	0.9458	0.9870	0.8767	0.9043

TABLE 1. The lowest RMSE and optimal r for SVD1 with basic imputation methods

First of all, we observe that as we expected the choice of the imputation method does matter. It can be most clearly seen on an example of data filled with zeros. For the best r RMSE there is around 2.9 that is, it is about 3 times larger than for other imputation methods. Other methods also differ. The lowest RMSE is obtained for the data filled with

weighted data. But the result for data filled with row means is also quite good. That's why we may suspect that optimizing α can give even better results.

Optimizing α. To get optimal result we perform optimization with respect to two parameters: α and r. As we can see on the picture above only r between 0 and 50 give some reasonable results, so we consider only those (we could use all r, but it is time consuming). Below, we present graph showing results of optimization.

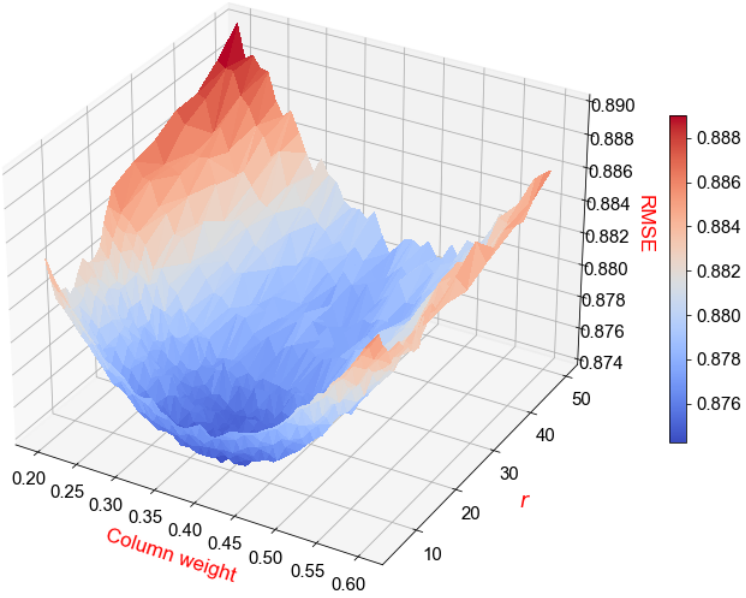

FIGURE 2. RMSE of SVD1 for weigthed imputation method for different α and r

Below we present also table with 5 lowest RMSE and pairs (α, r) that gave them.

α	r	RMSE
0.39	10	0.8740
0.38	10	0.8742
0.42	10	0.8743
0.36	10	0.8744
0.39	11	0.8745

TABLE 2. 5 lowest RMSE of SVD1 for weigthed imputation method and (α, r) that gave them

As we can see the pair $(0.39, 10)$ seems to be optimal in this case. All other pairs are close to it. może coś o tych RMSE

All results above are obtained for only one data split into train and test set. To find the best parameters in our method we have to average those parameters over different splits. So we considered 20 different splits and the results were as follows

- mean value of the best (α, r) is $(0.4055, 12.2)$,
- median of the best (α, r) is $(0.41, 13)$,
- in 15 of 20 cases the best pair is $(0.41, 13)$.

So $\alpha = 0.41$ and $r = 13$ are the parameters we use in our method SVD1 with weighted means as the imputation method.

SVD2. In this case we want to proceed as in SVD1 case. But before we start we have to choose some stop condition for SVD2.

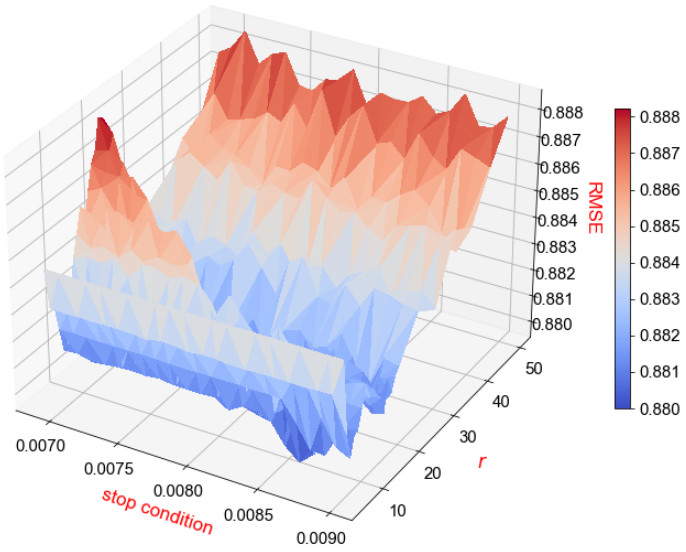

FIGURE 3. tutaj

Stop condition.

Optimizing r. After choosing the stop condition we can proceed exactly as in SVD1 case. So first of all we present a graph showing dependence of RMSE on r and on the imputation method for basic imputation methods.

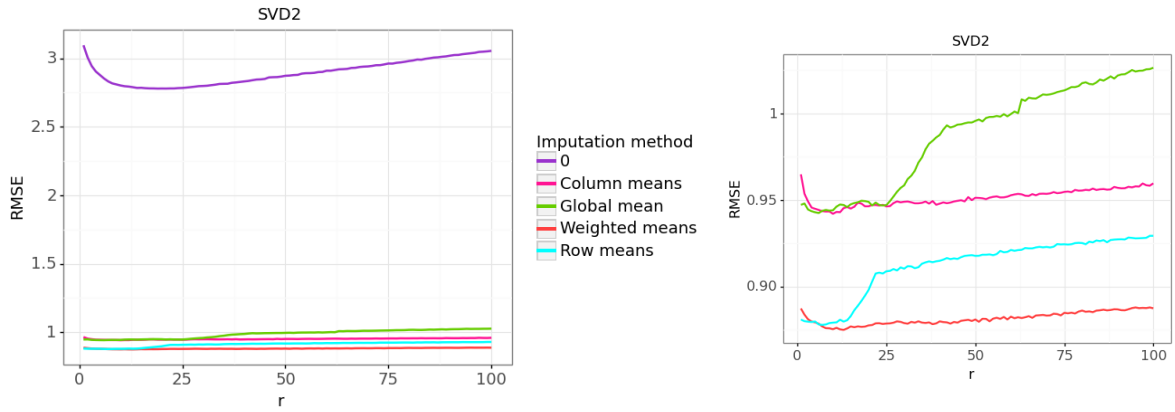

FIGURE 4. RMSE of SVD2 for basic imputation methods and $r = 1, \ldots, 100$

tutaj, że te wykresy są takie bardziej skaczące, ale generalnie trendy są te same

Now we present a table showing the best r and RMSE for every imputation method.

	0	column means	global mean	weighted means	row means
r	19	10	6	13	7
RMSE	2.7789	0.9420	0.9425	0.8749	0.8778

TABLE 3. The lowest RMSE and optimal r for SVD2 with basic imputation methods

Firstly, we can observe that SVD2 improved the results of SVD1. Every result is smaller, but order which methods are better or worse didn't change, maybe differences are a bit smaller.

We can observe again that only r between 0 and 50 gives reasonable results. Although the best r chosen by SVD2 in all cases differ a lot from those chosen by SVD1.

Optimizing α. Moving on to the weighted imputation method, we present a graph showing the results of optimization with respect to α and r.

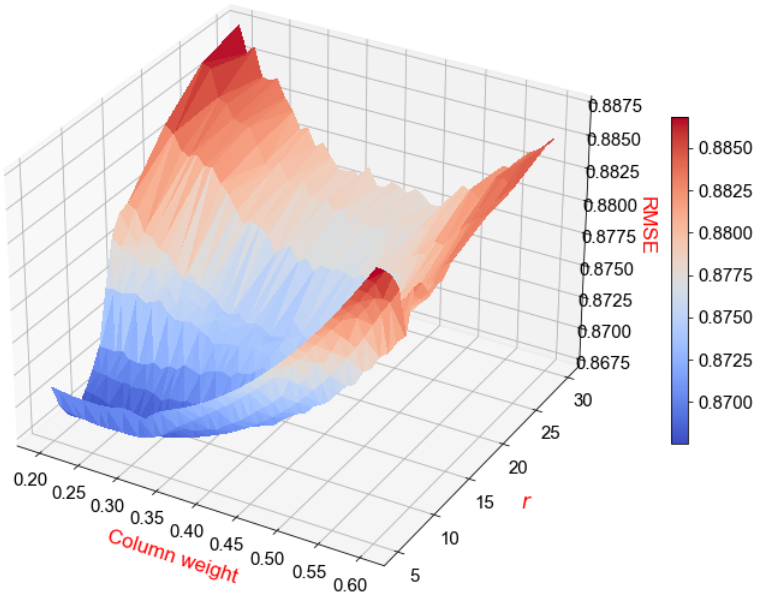

FIGURE 5. RMSE of SVD2 for weigthed imputation method for different α and r

It may look very similar to analogous graph for SVD1, but results actually differ a bit and it can be seen in following table.

α	r	RMSE
0.25	8	0.8674
0.26	8	0.8674
0.24	8	0.8674
0.27	8	0.8674
0.28	8	0.8675

TABLE 4. 5 lowest RMSE of SVD2 for weigthed imputation method and (α, r) that gave them

This time method did the best 5 results for the same r. Also in this case, the result are closer. It may suggest that this method is even more stable. It is intuitive for iterative method because they usually converge to some specific model and that is why we get clear results.

After repeating this optimization for 20 different splits we get that:

- the mean value of the best (α, r) is $(0.259, 8)$,

- the median of the best (α, r) is $(0.255, 8)$.

So $\alpha = 0.26$ and $r = 8$ are parameters we use in our SVD2 with weighted means as the imputation method.

NMF. In this case since we have only r and α to find, we proceed in exactly the same way as in the case of SVD.

Optimizing r. So firstly we present a graph showing dependence of RMSE on r and on the imputation method for basic imputation methods.

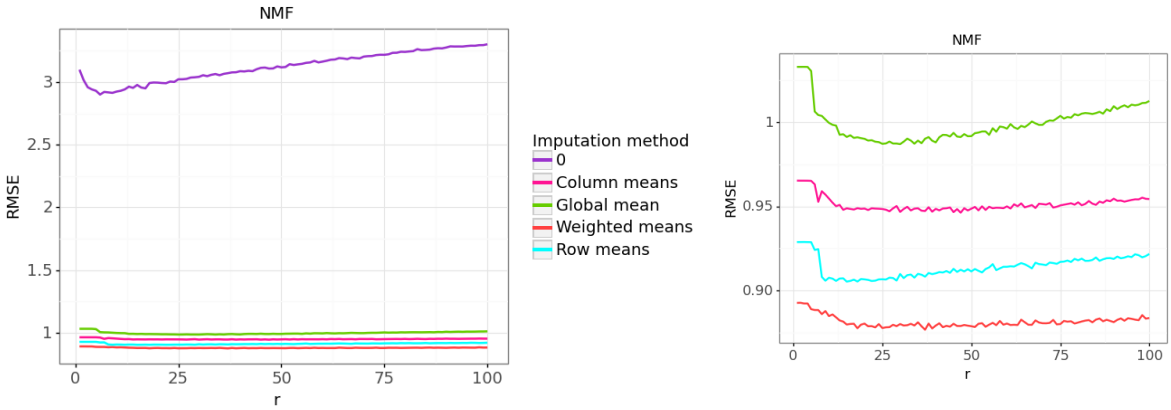

FIGURE 6. RMSE of NMF for basic imputation methods and $r = 1, \ldots, 100$

Comparing this graph to the graph for SVD we can see that it oscillates a lot, which means that for similar r it gives quite different RMSE. What's more, here it is not so obvious where to look for optimal r, because for instance for weighted means RMSE don't grow much with r. przeformułować jeszcze że kolejność kto który jest podobna

Below we also present a table with the lowest RMSE for every imputation method and the parameter r that gave it.

	0	column means	global mean	weighted means	row means
r	6	47	30	37	15
RMSE	2.8997	0.9462	0.9870	0.8766	0.9053

TABLE 5. The lowest RMSE and optimal r for NMF with basic imputation methods

As we can see the parameters r are in general bigger than in previous cases. The RMSEs are very similar to those obtained using SVD1.

Optimizing α. Now we perform the optimization with respect to α and r and present a graph showing the results.

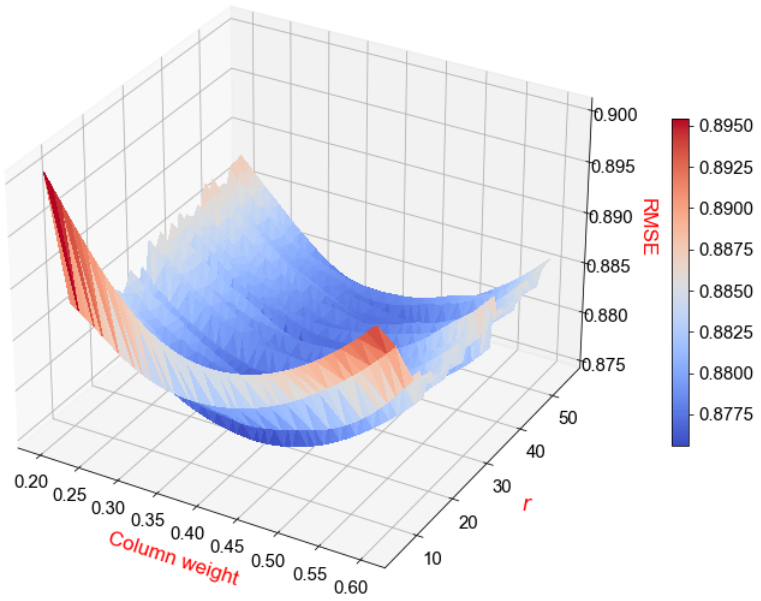

FIGURE 7. RMSE of NMF for weigthed imputation method for different α and r

Similarly as above, this graph is taki postrzępiony looking at the axis of r.

α	r	RMSE
0.40	37	0.8748
0.41	37	0.8748
0.39	18	0.8748
0.39	37	0.8748
0.40	18	0.8748

TABLE 6. 5 lowest RMSE of NMF for weigthed imputation method and (α, r) that gave them

After repeating this optimization for 20 different splits we get that:

- the mean value of the best (α, r) is $(0.393, 32.25)$,
- the median of the best (α, r) is $(0.39, 37)$,
- 15 times the best r is 37 and 5 times the best r is 18.

So $\alpha = 0.39$ and $r = 37$ are parameters we use in our NMF with weighted means as the imputation method.

SGD.

4

5. Results

Since in columns we keep indexes of movies, it means that our filled data take a bit more information from user ratings mean than from the movie ratings mean. That is probably logical ...