BIG DATA MINING AND ANALYTICS
ISSN 2096-0654 02/06 pp192–205
Volume 5, Number 3, September 2022
DOI: 10.26599/BDMA.2021.9020032

Optimal Dependence of Performance and Efficiency of Collaborative Filtering on Random Stratified Subsampling

Samin Poudel* and Marwan Bikdash

Abstract: Dropping fractions of users or items judiciously can reduce the computational cost of Collaborative Filtering (CF) algorithms. The effect of this subsampling on the computing time and accuracy of CF is not fully understood, and clear guidelines for selecting optimal or even appropriate subsampling levels are not available. In this paper, we present a Density-based Random Stratified Subsampling using Clustering (DRSC) algorithm in which the desired Fraction of Users Dropped (FUD) and Fraction of Items Dropped (FID) are specified, and the overall density during subsampling is maintained. Subsequently, we develop simple models of the Training Time Improvement (TTI) and the Accuracy Loss (AL) as functions of FUD and FID, based on extensive simulations of seven standard CF algorithms as applied to various primary matrices from MovieLens, Yahoo Music Rating, and Amazon Automotive data. Simulations show that both TTI and a scaled AL are bi-linear in FID and FUD for all seven methods. The TTI linear regression of a CF method appears to be same for all datasets. Extensive simulations illustrate that TTI can be estimated reliably with FUD and FID only, but AL requires considering additional dataset characteristics. The derived models are then used to optimize the levels of subsampling addressing the tradeoff between TTI and AL. A simple sub-optimal approximation was found, in which the optimal AL is proportional to the optimal Training Time Reduction Factor (TTRF) for higher values of TTRF, and the optimal subsampling levels, like optimal FID/(1–FID), are proportional to the square root of TTRF.

Key words: Collaborative Filtering (CF); subsampling; Training Time Improvement (TTI); performance loss; Recommendation System (RS); collaborative filtering optimal solutions; rating matrix

1 Introduction

A Recommender System (RS)[1,2] presents to a target user a list of possible items to be purchased or procured. The RS is derived based on explicit or implicit feedback of other users regarding various items, and perhaps on additional information such as demographics. RS can be categorized as either content-based filtering or Collaborative Filtering (CF) or hybrid filtering methods[3–6]. The CF-based RS is a widely used process in the area of recommendation studies[7], and it predicts a user's preference toward an item based on a rating matrix in which every user rates few items. The rating matrix is typically very sparse. Despite the sparsity, the computational cost of the CF algorithm is quite significant, considering that millions of users and millions of items can be involved[8,9]. Advancement of web-based e-commerce and other applications has resulted in the enormous growth in the size of the users and items[10].

Empirical studies have not been able to consistently ascertain whether the memory-based approach surpasses the model-based approach in terms of computational efficiency or vice versa[11–16]. In general, the model-

• Samin Poudel and Marwan Bikdash are with the Department of Computational Data Science and Engineering, North Carolina A&T State University, Greensboro, NC 27401, USA. E-mail: spoudel@aggies.ncat.edu; bikdash@ncat.edu.

* To whom correspondence should be addressed.

Manuscript received: 2021-07-27; revised: 2021-12-07; accepted: 2021-12-31

based CF performs better than memory-based CF in terms of accuracy when the rating matrix is highly sparse[11, 16].

In Ref. [13], the authors concluded that the matrix density influences the accuracy of CF algorithms, and that the training time of CF algorithms increases linearly with the size of the rating data used during training[13, 16]. Lee et al.[16] concluded that the performance of CF algorithms improves with increasing numbers of users, items, and density. In the survey in Ref. [3], the authors emphasized that the efficiency of traditional CF algorithms are degraded with the increase in sparsity and the dimension of data resulting from big data. Yang et al.[17] proposed converting user behavior to an implicit rating as a technique to alleviate the effect of reduced density on the performance of algorithms. In general, however, the tradeoff between the accuracy and the computational cost in the presence of sparsity is not well understood.

Subsampling increases the computational efficiency of CF algorithms. The need of computational improvement was strongly argued in Ref. [8]. Subsampling, however, reduces the size of data used and adversely affects accuracy. Subsampling can be performed using probability sampling or non-probability sampling[18]. Simple random sampling, stratified sampling, and cluster sampling are popular techniques of probability-based sampling and are common in the field of RS as well. Researchers have also used judgmental sampling which falls under non-probability sampling to vary the density of rating data under study[13, 19]. The performance of these established sampling methods seems to be comparable. In this paper, we will rely on judgmental sampling, but will modify this algorithm as to allow us to control density, and prevent pathological cases. These constraints are important when trying to derive models of the effect of subsampling on accuracy and efficiency.

Deljoo et al.[20] provided statistical analysis showing that the dataset characteristics (such as size, shape, and density) have a strong effect on the success of the malicious Shilling effect. They also showed that the regression models are significant when studying the effects of data characteristics in Shilling attack, but did not use the coefficients in an optimal analysis. In Ref. [12], the tradeoff between improving the accuracy of RS at the expense of larger datasets (and hence more computations) was discussed. They also conducted extensive simulations involving many models, with each model representing a combination of method and dataset, and they reached the qualitative conclusion that the dataset characteristics affect significantly the performance of 3 CF algorithms. No attempt to show a general trend among the models was made. Moreover, the regression models fitted in Ref. [12], were not used in any subsequent analysis, or in an optimal design of subsampling.

Several studies compare the accuracy of CF algorithms for various combinations of dataset properties like the numbers of users and items, density, etc.[1, 8, 13, 16, 17, 19, 21], but they do not derive or propose explicit models that describe the interdependence of accuracy, efficiency, and dataset characteristics. Moreover, many studies simply derive a model, ascertain that such a model is significant, but rarely use the model to design experiments or propose optimal subsampling. The models are derived mainly for qualitative intuition. This is perhaps partly due to the implicit assumption that the models themselves are not robust or easily interpretable, and a different combination of dataset and algorithm will likely lead to a different model, therefore suggesting a limited use of the interdependence models.

In this paper, we attempt to show that the interdependence models can be robust and interpretable, and one can subsequently develop consistent subsampling guidelines that are likely to be applicable across many methods and datasets. In fact, our simulations have indicated that the models developed so far for the efficiency are robust and constant across many methods and all datasets and depend only on the fractions of users and items used. We show that accuracy models are more complex, but still interpretable. This does not preclude that possibility that, with additional investigations, one can develop accuracy models that are robust or constant across methods and datasets.

The approach followed here is to find explicit models of Accuracy Loss (AL) and Training Time Improvement (TTI) in terms of Fraction of Items Dropped (FID), the Fraction of Users Dropped (FUD), and the dataset characteristics, and to test the constancy of such models across methods and datasets. Next, we check whether knowing the subsampling levels is sufficient to model the effects of subsampling on the efficiency and performance of the CF algorithms, or if additional data properties like density, numbers of users and items, and so forth are required. Subsequently, we derive closed-form expressions for the optimal tradeoff between the accuracy and efficiency induced by subsampling.

The approach here is tested using four well-known datasets (1M MovieLens, 25M MovieLens, Yahoo! Music, and Amazon Automotive dataset), from which many primary datasets are derived. The approach is also based on seven well-established CF approaches:

(1) Regularized Singular Value Decomposition (SVD)[13,22];

(2) SVD with bias terms, denoted as SVD_b[23];

(3) Non-negative Matrix Factorization (NMF)[24];

(4) SlopeOne[25];

(5) CoClustering[10];

(6) User-based Nearest Neighbor (UNN)[20,26];

(7) Item-based Nearest Neighbor (INN)[20,26].

The main contributions of this paper are as follows:

- We developed a Density-based Random Stratified Subsampling using Clustering (DRSC) algorithm for stratified subsampling under various constraints like FID, FUD, and density level. The algorithm combines best practices suggested in the literature, and seeks to maintain the original level of sparsity without introducing pathological cases.
- We derived simple, validated, intuitive, and closed-form estimates for TTI and AL for CF algorithms showing dependence on FID and FUD.
- We derived closed-form expressions for the optimal TTI for a specified AL and the optimal AL for specified TTI with and without constraints on subsampling levels. An elegant accurate sub-optimal expression of the tradeoff was identified.

In Section 2, we define the notation and overview the tools, metrics, and the proposed methodology. In Section 3, we propose and discuss the DRSC algorithm. In Section 4, a simple multi-linear model for CF efficiency is proposed, and in Section 5 a simple model for CF performance is proposed. In Section 6, the optimal subsampling solutions that maximize the efficiency or reduce the AL are derived. In Section 7, the process to use optimal subsampling model is discussed. We put forward conclusions of our study and possible future work in Section 8.

2 Methodology

2.1 Subsampling levels

We use m and n to represent the number of rows (users) and columns (items) in a rating matrix, respectively. Rows and users are used interchangeably. Similarly, items and columns are used interchangeably. R denotes a cluster of rows and C denotes a cluster of columns. r and c denote indices of rows and columns, respectively. δ denotes the density of a matrix, i.e., the ratio of the number of non-zero elements over the total number of elements. The sparsity of a matrix is denoted by $1-\delta$.

We denote the FUD with μ, and the FID with ν. If one interprets μ as the probability of dropping a user, then one can define the Odds Ratio of Dropping a User (ORDU) as

$$\text{ORDU} = \frac{\mu}{1-\mu} = O_\mu \tag{1}$$

Similarly, if one interprets ν as the probability of dropping an item, then one can define the Odds Ratio of Dropping an Item (ORDI) as

$$\text{ORDI} = \frac{\nu}{1-\nu} = O_\nu \tag{2}$$

2.2 Performance metrics

Let $p_{u,i}$ represent the available rating of a user u towards an item i. The corresponding primary matrix is denoted P. One can consider a specified CF algorithm taking a primary matrix P as input and returning 2 measures: The accuracy measure of the rating prediction A and the computation time required to train the CF T. Here we denote $[A, T] = \text{CF}(P)$. The computations involved include: Splitting the primary matrix into a training set and a testing set, learning a prediction formula, and then applying the prediction formula to the testing set. We implemented various CF algorithms using the SURPRISE python library[23]. We made sure that the same training and test sets were used while applying all 7 CF methods considered in this study by defining the specific random state during the train-test split[23].

The error of the CF prediction can be assessed using a variety of predictive accuracy measures[27]. If $\hat{a}$ is an estimate of a, one can use the commonly-used Root Mean Squared Error (RMSE)[13],

$$\text{RMSE}(\hat{a}) = \sqrt{\frac{1}{|K|}\sum_{k\in K}(a_k-\hat{a}_k)^2} \tag{3}$$

or alternatively, the Mean Absolute Error (MAE)[28],

$$\text{MAE}(\hat{a}) = \frac{1}{|K|}\sum_{k\in K}|a_k-\hat{a}_k| \tag{4}$$

as measures of inaccuracy.

In this paper, we measure the AL as

$$\text{AL} = \frac{\text{RMSE}^S - \text{RMSE}^P}{\text{RMSE}^P} \tag{5}$$

where the superscripts P and S refer to the primary and subsampled rating matrices, respectively. The AL becomes larger (more positive) as one increases the FUD and FID subsampling levels, because the RMSE is expected to increase with FUD and FID. We measure the TTI as

$$\text{TTI} = \frac{T^P - T^S}{T^P} \tag{6}$$

which becomes more positive as one increases the FUD and FID subsampling levels. TTI$=0$ means there is no improvement. Equivalently, one can represent the improvement in training time using the Training Time Reduction Factor (TTRF),

$$\text{TTRF} = \frac{T^P}{T^S} = \frac{1}{1-\text{TTI}} \tag{7}$$

For example, if $T^P = 5$ and $T^S = 1$, then TTRF $= T^P/T^S = 5$, which means that the training time is reduced by a factor of 5. The corresponding TTI is TTI $= (5-1)/5 = 0.8$, which is harder to interpret. A TTRF of 50 would correspond to TTI $= (50-1)/50 = 0.98$. In our numerical experiments, it became clear that the regression models for TTI in terms of FID and FUD are more elegant than those for TTRF. Of course, one can deduce the corresponding models for the TTRF using Eq. (7), as will be illustrated in Section 4.

2.3 Overview of the methodology

Figure 1 shows the overall methodology, with the symbol $*$ indicating the contributions in this paper. One starts with a primary matrix P, which represents the original data, or a subset thereof. A subsampled matrix S is then produced through the proposed DRSC algorithm, described in detail in Section 3, which drops a fraction μ of rows and a fraction ν of columns of P while keeping the density of S close to its original value and avoiding pathologies such as zero rows and columns. Subsequently, 7 CF algorithms are applied to the subsampled matrix to generate the metrics $[A^S, T^S] = \text{CF}(S)$ for each of the CF algorithm. The process of subsampling and applying CF algorithms is repeated by varying FID and FUD. Then, the dependence of AL and TTI on FID and FUD is modeled, and the resulting models are validated. Closed-form expressions of the optimal tradeoff are derived using constrained nonlinear optimization and Lagrange multipliers, and simple recommendations are provided.

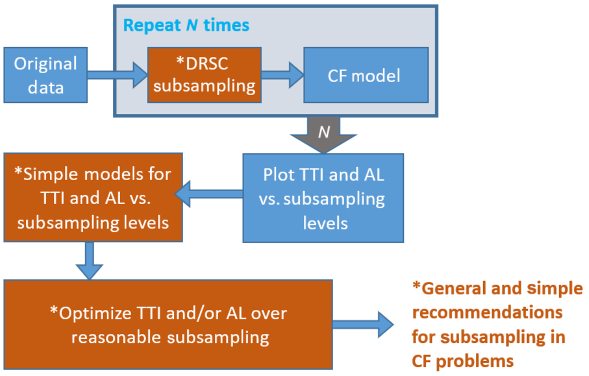
Repeat *N* times
Original data
*DRSC subsampling
CF model
N
*Simple models for TTI and AL vs. subsampling levels
Plot TTI and AL vs. subsampling levels
*Optimize TTI and/or AL over reasonable subsampling
*General and simple recommendations for subsampling in CF problems

Fig. 1 Overview of methodology.

3 Algorithm to Subsample Rating Matrix with Density-Constraints

Many sampling techniques, such as Refs. [12, 20], generally extract data from the rating matrix using simple random sampling. Randomly sampling rows of a rating matrix may, however, lead to dropping a significant number of information-rich rows (users) and information-rich columns (items). Moreover, dropping many rows can produce zero columns, and dropping columns can produce zero rows, thus potentially creating numerical problems. In Ref. [12], it was suggested to drop the resulting empty rows and columns, thus affecting the desired size of users or items in the subsampled rating matrix. To control these characteristics in our experiments, we sought to subsample the primary matrix P as to obtain a subsampled matrix S with the objectives of (1) preventing totally zero rows and columns, (2) preventing a significant deterioration in densities of the rows and columns, and (3) keeping the overall density of the subsampled matrix closed to the density of primary matrix. This is important when the subsampling is heavy and the matrices resulting from subsampling are small compared to those of the original matrix.

The added constraints on FUD, FID, and δ have led us to propose a DRSC algorithm that achieves the above objectives.

The DRSC algorithm is described below:

Step 1: Cluster the rows of P into $R_1, R_2, \ldots,$ based on their densities.

Step 2: Obtain S by randomly subsampling a fraction $1-\mu$ of rows from every row cluster without replacement.

Step 3: If S has an all-zero column c_j, update S by replacing a row r_k with a row closest in density to it but which has a nonzero value in c_j. Care must be exercised that this replacement does not create a new all-zero column in the updated S. If no row replacement introduces a nonzero into c_j, then c_j is dropped. Note that multiple rows can be replaced until the density of the modified c_j approaches a desired value, if possible, but this strategy was not pursued because Steps 4–6 tend to alleviate that problem.

Step 4: Cluster the columns of S into $C_1, C_2, \ldots,$ based on their densities.

Step 5: Update S by randomly subsampling a fraction $1 - \nu$ of columns from every column cluster without replacement.

Step 6: If S has an all-zero row r_k, update S by replacing a column c_j with a column closest in density to it but which has a nonzero value in r_k. Care must be exercised that this replacement does not create a new all-zero row in the updated S. If no column replacement introduces a nonzero into r_k, then r_k is dropped. Note, however, that this situation rarely arises.

Here, subsampling starts with rows and then goes on to subsample columns. The opposite could have been used. One can speed up the subsampling process by dropping c_j in Step 3 and dropping r_k in Step 6, instead of going through every possibility. This reduction in the computational complexity, however, is not really important, as will be illustrated later in Table 1. The computational complexity of the proposed subsampling algorithm is very negligible when compared to the savings obtained in reducing the training time, which is usually by far the most computationally costly step in developing a recommender system.

The DRSC algorithm constrains the density of the subsampled matrix to be close to the primary matrix. It can also be used to extract a rating matrix having the desired number of users, number of items, and density. This can be achieved by sampling from the the clusters with desired density of Steps 1 and 4. In case of subsampling without replacement, if the cluster with desired density does not have enough users in Step 1 or not enough number of items in Step 4, one can sample an equal number of users or items from the clusters having density just below and above the desired density.

To illustrate the savings achieved by subsampling, we used the UNN CF algorithm with P containing 30 000 users and 5500 items derived from the 25M MovieLens dataset. In a first experiment, we dropped 15 000 users and 2750 items using the DRSC algorithm (resulting FID = FUD = 0.5 as shown in Table 1). The required Subsampling Time (ST) was about 3.51 s. The Training Time (TT), however, dropped dramatically from 1542.20 s to 214.90 s with a saving of 1327.30 s. The ratio of subsampling time over the saving in training time is 0.26%, which is very negligible. If one starts with 30 000 users and 5500 items, and drops 70% of the items and 70% of the users, the ratio of ST to the improvement in training time is estimated to be 0.17%. The observations in Table 1 suggest that the subsampling cost is very negligible compared to the learning time, except after extreme subsampling (keeping 1% of the data or less) at which point the saving in training time is several order of magnitude larger than the subsampling time. The larger the datasets, the more pronounced this trend becomes.

Table 1 Subsampling and training time analysis with size of $P = (30\,000, 5500)$ using UNN CF. Here, $\Delta\mathrm{TT} = 1542.2 - \mathrm{TT}$.

FID	FUD	ST (s)	TT (s)	ST/TT (%)	ST/ΔTT (%)
0	0	0	1542.20	0	–
0.3	0.3	4.45	595.10	0.74	0.46
0.5	0.5	3.51	214.90	1.60	0.26
0.7	0.7	2.63	63.20	4.20	0.17
0.9	0.9	1.90	2.55	74.00	0.12

In the subsequent development, ST was not considered while building the TTI models primarily because it is generally negligible. Moreover, the institution making the recommendation needs several recommender systems in different contexts (e.g., for different groups of users or items or using various algorithms and hyper-parameters) and the subsampling can be conducted once for use in many recommender systems.

4 Modeling the Effect of Subsampling on Efficiency

For this study, we used primary rating matrices extracted from the 1M MovieLens dataset[29, 30], the 25M MovieLens dataset[30, 31], the Yahoo! Music dataset[32], and the Amazon Automotive data[33]. The primary rating matrices extracted from 1M MovieLens and the Amazon Automotive datasets have rating data in discrete numerical rating scale of 1 to 5. Primary rating matrices extracted from 25M MovieLens dataset[31] have rating data in discrete numerical rating scale of 0.5 to 5 in steps of 0.5. Primary rating matrices extracted from Yahoo! Music dataset[32] have rating data in discrete numerical rating scale of 1 to 100. Here, P_1 and P_2 are extracted from the 1M MovieLens dataset, P_3 and P_4 are extracted from the 25M MovieLens dataset, P_5 and P_6 are extracted from the Yahoo! Music dataset, and P_7 is extracted from the Amazon Automotive dataset.

Details of primary rating matrices used have been tabulated in Table 2.

A large number of subsampled rating matrices with

Table 2 Details of primary rating matrices used during study.

Primary dataset	Number of users (m)	Number of items (n)	Density (δ)	Rating scale of data	Source
P_1	6040	3706	0.045	1–5	1M MovieLens
P_2	4607	2080	0.095	1–5	1M MovieLens
P_3	8000	4004	0.101	0.5–5	25M MovieLens
P_4	4009	8017	0.151	0.5–5	25M MovieLens
P_5	3500	6000	0.08	1–100	Yahoo! Music
P_6	5006	5011	0.12	1–100	Yahoo! Music
P_7	3000	1301	0.004	1–5	Amazon Automotive

different FUD and FID were subsequently extracted from each of the primary rating matrices using the DRSC algorithm in Section 3. During subsampling we have varied FUD and FID from 0.1 to 0.9 in steps of 0.02. The extensive empirical analysis of the TTI and AL versus the subsampled levels of users and items involved extensive simulation of 10 086 subsampled rating matrices from six different primary rating matrices with different dataset characteristics. Moreover, 1681 rating matrices were subsampled from each of the primary rating matrices.

In Fig. 2 , we depict the variation of the CF efficiency with FID for a specified FUD. Similarly, in Fig. 3 we present the variation of CF efficiency with FUD for a specified FID. Each of the plots shows a fitted linear regression model minimizing a least-squares criterion.

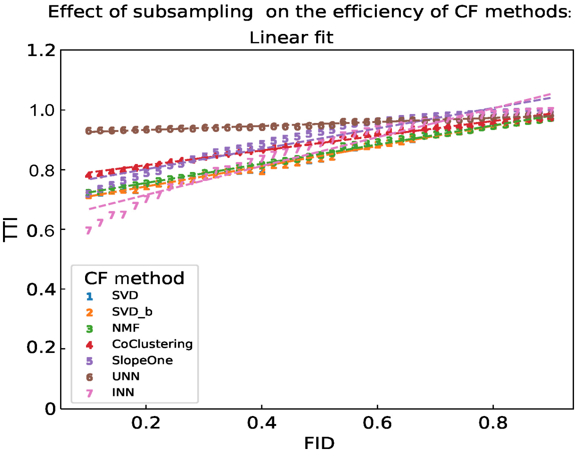

Fig. 2 TTI versus FID using P_1 at constant FUD = 0.7.

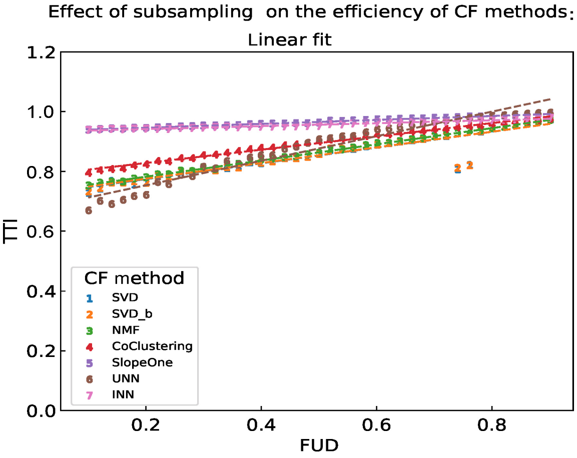

Fig. 3 TTI versus FUD using P_1 at constant FID = 0.7.

The fitting was implemented in SciPy[34] according to Eq. (8), which was suggested by the discussion below.

We can see in Fig. 2 that the SlopeOne and the INN CF recommenders appear to exhibit slight deviations from linearity. To verify that the relationship is indeed linear, we conducted the following experiment. For each of the 6 datasets, the coefficient of determination[35] R^2 is computed and an ANOVA test for linear regression[36] is performed. The tests show that the SlopeOne TTI has strong linear relationship with FID at constant FUD, with $R^2 > 0.9$, and the p-value of the ANOVA tests for linear regression being less than 0.01 in each case.

For instance, using data from P_1, we obtain $R^2 = 0.92$ and $p = 0$ for SlopeOne TTI versus FID at the constant FUD of 0.7. Thus, we conclude that the relationship between SlopeOne TTI and FID is linear at constant FUD. Similarly, $R^2 > 0.9$ and $p < 0.01$ for the linear relationship of INN CF TTI versus FID at constant FUD. For instance, $R^2 = 0.92$ and $p = 0$ for INN CF TTI versus FID at the constant FUD of 0.7 using data from P_1. The recommenders based on SVD, SVD_b, NMF, CoClustering, and UNN exhibit a linear dependence of TTI on the FID at constant FUD as depicted in Fig. 2. In Figs. 2 and 3, plots of SVD and SVD_b recommender TTI are in complete overlap with each other.

Moreover, from Fig. 3, TTI increases at a constant rate with μ for a given ν for the SVD, SVD_b, NMF, CoClustering, SlopeOne, and INN, and the rate of increase is higher for smaller ν. The coefficients of determination R^2 for UNN TTI versus FUD at constant FID are greater than 0.9 and the p-values of the corresponding ANOVA tests for linear regression are less than 0.01, thus indicating that there is linear relationship between UNN TTI and FUD μ at constant FID ν. For instance, $R^2 = 0.95$ and $p = 0$, for UNN TTI versus FUD at the constant FID of 0.7 using data from P_1.

In short, the linearity of the TTI vs. FID at constant FUD (and vice-versa) of the 7 CF methods considered in this study is established. This implies that TTI is multi-linear in μ and ν, thus suggesting the overall model,

$$\text{TTI} = \beta_0\mu + \beta_1\nu - \beta_2\mu\nu \tag{8}$$

where an intercept is not included because, by definition, TTRF = 1 and TTI = 0 when the FUD and FID are zero.

Next, we turn our attention to whether the β coefficients in Eq. (8) are indeed constant or predictable across all models and methods. We proceed as follows:

Step 1: Sample around 25% from the 1681 combinations of TTI, FID, and FUD of P.

Step 2: Confirm that FID and FUD have some values less than and some values greater than 0.5 in the sampled combinations.

Step 3: Compute the the least-squares regression model in Eq. (8) and store values of β coefficients, and the corresponding p-values.

Step 4: Steps 1 to 3 were repeated 50 times.

Step 5: Average the values of β coefficients and compute the standard deviation σ_β for each β. For instance $\sigma_{\beta_1}$is the standard deviation computed from 50 values of β_1. Also, compute combined p-value using Fisher's method[37].

We report the regression analysis of the TTI model in Eq. (8) in Table 3. In Table 3 values of β coefficients indicate their average values.

The values of β_0 in Table 3 are statistically the same for the individual CF approaches regardless of the dataset. Similarly β_1 is similar for a CF algorithm for every dataset used. Moreover, β_2 of a CF approach is similar regardless of the dataset used. Hence, we can infer that the improvements in the efficiency of CF approaches can be reliably estimated using the subsampled levels of users and items, and the type of CF approach being used. Having said that, we also want to emphasize that all β coefficients ≈ 1 for matrix factorization based CF approaches and CoClustering CF approach.

We have evaluated the model in Eq. (8) using $P_1, \ldots, P_7$ and taking TTI as an estimate. Values of β coefficients of a dataset in the Table 3 have been used to estimate the TTI of the respective dataset. The MAE based on TTI using $P_1, \ldots, P_7$ is as shown in Table 4.

Using Eq. (8) and the estimates of β coefficients in Table 3, we propose the final model for predicting the CF efficiency as a function of subsampling for the matrix factorization based methods (SVD, SVD_b, NMF) and the CoClustering CF approach as

$$\text{TTI} = \mu + \nu - \mu\nu \tag{9}$$

The corresponding TTRF model would be

$$\text{TTRF} = \frac{1}{1 - \mu - \nu + \mu\nu} \tag{10}$$

which is harder to guess or fit directly.

Table 3 Results of regression analysis of TTI using model in Eq. (8).

CF method	P_1			P_2			P_3			P_4		
	β_0	β_1	β_2	β_0	β_1	β_2	β_0	β_1	β_2	β_0	β_1	β_2
SVD	0.94″	1.02″	0.96	1.01″	1.01″	1.02″	1.03	1.12″	1.15′	1.0″	1.06″	1.05″
SVD_b	0.94″	1.03″	0.96″	1.01″	1.01″	1.02″	1.05″	1.06″	1.13″	1.03″	1.06″	1.09″
NMF	0.98″	1.024″	0.99″	1.03″	1.03″	1.06″	1.02″	0.98″	0.98″	1.03″	1.05″	1.08″
CoClustering	1.06″	1.07″	1.165″	1.12″	1.10″	1.15″	1.09″	1.07″	1.16″	1.06″	1.02″	1.09″
SlopeOne	1.04″	1.35″	1.44″	1.12″	1.37″	1.52′	1.11″	1.33″	1.50″	1.14″	1.37″	1.54′
UNN	1.32″	0.95″	1.28″	1.36″	1.01″	1.41′	1.34″	0.96″	1.32′	1.37″	1.06″	1.42′
INN	0.9″	1.35″	1.26′	0.93″	1.33″	1.32″	0.99″	1.33″	1.32′	0.99″	1.31″	1.32″

CF method	P_5			P_6			P_7		
	β_0	β_1	β_2	β_0	β_1	β_2	β_0	β_1	β_2
SVD	1.00″	1.01″	1.00″	1.01″	1.02″	1.04″	0.99″	1.02″	1.02″
SVD_b	0.99″	0.99″	0.97″	1.00″	1.01″	1.01″	0.98″	1.00″	0.99″
NMF	0.98″	0.99″	0.97″	1.01″	1.01″	1.01″	0.97″	0.93″	0.98″
CoClustering	1.03″	1.02″	1.06″	1.02″	1.00″	1.02″	0.95″	0.92″	0.97″
SlopeOne	1.06″	1.31″	1.45″	1.11″	1.34″	1.52″	0.82″	1.18″	0.92’
UNN	1.34″	0.93″	1.31′	1.36″	1.00″	1.39′	1.26″	0.91″	1.14″
INN	0.86″	1.29″	1.21″	0.96″	1.31″	1.28″	0.85″	1.25″	0.95′

Note: A double prime ″ indicates small standard deviation $\sigma_{\beta_i} \leqslant 0.02$, while a single prime ′ indicates $0.02 < \sigma_{\beta_i} < 0.04$; moreover, every regression coefficient had a combined p-value < 0.01. Combined p-value is computed using Fisher's method[37].

Table 4 Evaluation of the TTI model in Eq. (8).

CF method	MAE						
	Using P_1	Using P_2	Using P_3	Using P_4	Using P_5	Using P_6	Using P_7
SVD	0.024	0.009	0.031	0.025	0.005	0.005	0.015
SVD_b	0.024	0.009	0.032	0.018	0.007	0.005	0.015
NMF	0.012	0.009	0.025	0.012	0.007	0.005	0.014
CoClustering	0.014	0.015	0.023	0.018	0.007	0.007	0.012
SlopeOne	0.041	0.042	0.048	0.046	0.037	0.042	0.041
UNN	0.044	0.047	0.043	0.048	0.052	0.049	0.036
INN	0.048	0.042	0.042	0.037	0.035	0.038	0.045

We have averaged values of β_0, β_1,and β_2 in Table 3 for SlopeOne, UNN, and INN methods and proposed the TTI model. The TTI model proposed for SlopeOne is

$$\text{TTI} = 1.05\mu + 1.33\nu - 1.42\mu\nu \tag{11}$$

The TTI model proposed for UNN is

$$\text{TTI} = 1.34\mu + 0.99\nu - 1.32\mu\nu \tag{12}$$

The TTI method proposed for INN is

$$\text{TTI} = 0.95\mu + 1.31\nu - 1.23\mu\nu \tag{13}$$

Based on the findings shown in the Table 3, the density of subsampled dataset does not effect the proportion of time saved with changing FID and FUD. Note that changing the density of the rating matrix changes the training time of the CF algorithm using it, but does not change the improvement in efficiency due to subsampling.

In short, the models of TTI for all CF approaches seem to be independent of the shape, size, and density of the matrix. The difference between the first 4 and last 3 models, while obvious, is not drastic, and is unlikely to change the optimal subsampling guidelines developed later.

5 Modeling the Effect of Subsampling on AL

The dependence of the accuracy loss on subsampling turns out to be more complicated. For instance, plotting AL vs. FID at a constant FUD shows a nonlinear variation in Fig. 4. Plotting AL vs. FUD at constant FID shows a similar pattern. For all CF approaches, the nature of plots are similar for AL versus FID at constant FUD as shown in Fig. 4.

After some experimentation, we noted that the Scaled Accuracy Loss (SAL) is

$$\text{SAL} = (1-\mu)(1-\nu)\,\text{AL} \tag{14}$$

and appears to be multi-linear in μ and ν for all the CF approaches. The relationship of SAL with μ at constant ν is illustrated in Fig. 5 and the relationship of SAL with ν at constant μ is shown in Fig. 6. Figures 5 and 6 suggest the use of a simple regression model for SAL as

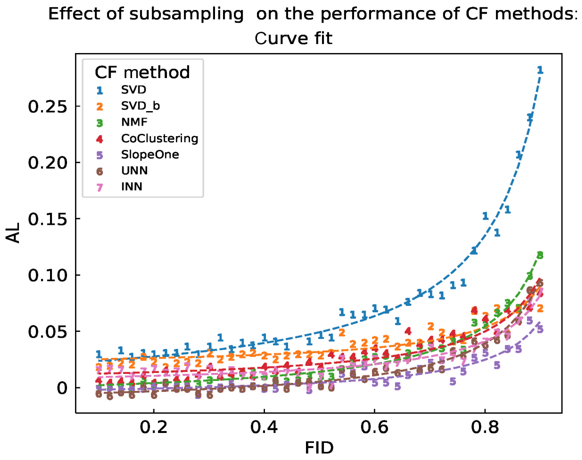

Fig. 4 AL versus FID using P_1 at constant FUD = 0.7.

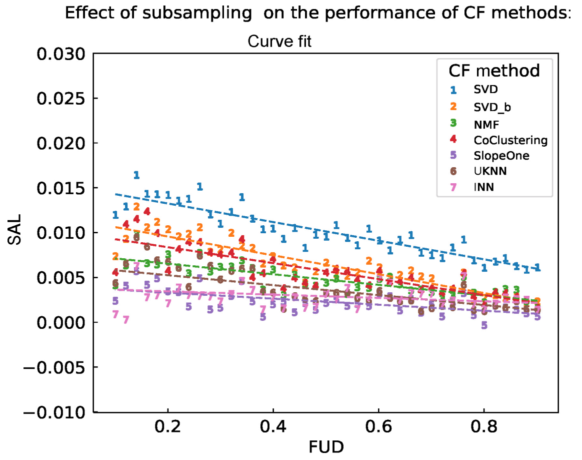

Fig. 5 SAL versus FUD using P_1 at constant FID = 0.7.

$$\text{SAL} = \eta_0\mu + \eta_1\nu - \eta_2\mu\nu \tag{15}$$

where an intercept is not included because, by definition, SAL = 0 if the FUD and FID are zero.

Equations (14) and (15) lead to

$$\text{AL} = \frac{\eta_0\mu + \eta_1\nu - \eta_2\mu\nu}{(1-\mu)(1-\nu)} \tag{16}$$

Next, we use regression analysis to estimate the η coefficients in a least-squares sense, for each of the

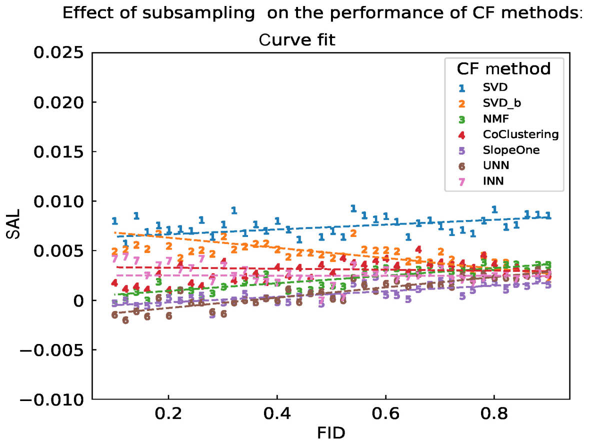

Fig. 6 SAL versus FID using P_1 at constant FUD = 0.7.

datasets $P_1, \ldots, P_7$. A regression analysis similar to that of Section 4 was conducted. Using combination of density, FUD, and FID, the η coefficients for the CF approaches based on the datasets $P_1, \ldots, P_7$ are shown in Table 5. In Table 5, the values of η coefficients for each method and for each dataset are the average values of 50 η coefficients obtained from 50 regression analyses.

We have evaluated the model in Eq. (15) using P_1, $\ldots$, P_7 and taking AL as an estimate. Values of η coefficients were provided as in Table 5 while estimating the AL. The MAE based on AL for $P_1, \ldots, P_7$ is as shown in Table 6.

Hence, The final model for estimating the effect of subsampling on the accuracy performance of CF methods is given by

$$\text{AL} = \frac{\eta_0\mu + \eta_1\nu - \eta_2\mu\nu}{(1-\mu)(1-\nu)} \tag{17}$$

The regression coefficients η are not independent of the characteristics of the datasets subsampled as it was the case for the coefficients in the TTI model. The varying values of η coefficients are based on the dataset

Table 5 Results of regression analysis of AL using model in Eq. (15).

CF method	P_1			P_2			P_3			P_4		
	η_0^*	η_1^*	η_2^*	η_0^*	η_1^*	η_2^*	η_0^*	η_1^*	η_2^*	η_0^*	η_1^*	η_2^*
SVD	$0.91'''$	$2.1'''$	$2.71'''$	$0.8'''$	$1.42'''$	$2.1'''$	$1.53'''$	$1.67'''$	$3.36'''$	$1.49'''$	$1.7'''$	$3.43'''$
SVD_b	$0.88'''$	$1.2'''$	$2.22'''$	$1.12'''$	$1.22'''$	$2.5'''$	$1.66'''$	$1.82'''$	$3.7'''$	$1.65'''$	$1.91'''$	$3.84''$
NMF	$0.29'''$	$1.12'''$	$1.33'''$	$0.47'''$	$1.0'''$	$1.38'''$	$0.23'''$	$0.11'''$	$0.24'''$	$0.03'''$	$0.04'''$	$0.0'''$
CoClustering	$0.52'''$	$1.14'''$	$1.68'''$	$0.28'''$	$0.76'''$	$0.98'''$	$0.37'''$	$0.41'''$	$0.72'''$	$0.25'''$	$0.28'''$	$0.56'''$
SlopeOne	$0.07'''$	$0.57'''$	$0.57'''$	$0.06'''$	$0.3'''$	$0.26'''$	$0.34'''$	$0.07'''$	$0.39'''$	$0.14'''$	$0.25'''$	$0.45'''$
UNN	$0.01'''$	$0.88'''$	$0.83'''$	$0.13'''$	$0.75'''$	$0.84'''$	$0.43'''$	$0.9'''$	$1.38'''$	$0.29'''$	$0.57'''$	$0.94'''$
INN	$0.38'''$	$0.66'''$	$0.97'''$	$0.73'''$	$0.76'''$	$1.48'''$	$1.18'''$	$0.49'''$	$1.76'''$	$1.32'''$	$0.42'''$	$1.95'''$

CF method	P_5			P_6			P_7		
	β_0	β_1	β_2	β_0	β_1	β_2	β_0	β_1	β_2
SVD	$0.37''$	$0.36''$	$1.0'$	$0.08'''$	$0.08'''$	$0.05'''$	$-1.03'''$	$-2.2''$	$3.10'$
SVD_b	$2.40'$	$1.73'$	$3.72'$	$0.71''$	$0.61''$	$2.44'$	$1.40''$	$1.68''$	$3.37'$
NMF	$0.12'''$	$0.07'''$	$0.19'''$	$0.05'''$	$0.10'''$	$0.17'''$	$2.43''$	$1.76''$	$5.20'$
CoClustering	$0.76'''$	$0.38'''$	$1.11'''$	$0.10'''$	$0.19'''$	$0.26'''$	$2.61''$	$2.67''$	$6.26'$
SlopeOne	$0.41'''$	$0.16'''$	$0.56'''$	$0.09'''$	$0.13'''$	$0.23'''$	$0.18''$	$0.80''$	$1.39'$
UNN	$1.24'''$	$1.01'''$	$2.34'''$	$0.77'''$	$1.25'''$	$2.14'''$	$1.42''$	$3.12''$	$5.30'$
INN	$2.04'''$	$0.98'''$	$3.22'''$	$2.59'''$	$1.82'''$	$4.79'''$	$2.83''$	$0.63''$	$4.08'$

Note: $\eta_0^* = \eta_0 \times 10^2, \eta_1^* = \eta_1 \times 10^2$, and $\eta_2^* = \eta_2 \times 10^2$. The triple prime $'''$ indicates standard deviation $\sigma_{\eta_i} \leqslant 0.001$, the double prime $''$ indicates $0.001 < \sigma_{\eta_i} \leqslant 0.002$, and the single prime $'$ indicates $0.002 < \sigma_{\eta_i} \leqslant 0.003$. Moreover, each regression coefficient had a combined p-value < 0.01. Combined p-values are computed using Fisher's method[37].

Table 6 Evaluation of the AL model in Eq. (15).

CF method	MAE						
	Using P_1	Using P_2	Using P_3	Using P_4	Using P_5	Using P_6	Using P_7
SVD	0.0084	0.0067	0.009	0.012	0.0084	0.0038	0.024
SVD_b	0.0082	0.0086	0.012	0.0141	0.0759	0.0132	0.045
NMF	0.0077	0.0069	0.0056	0.0042	0.0016	0.0015	0.021
CoClustering	0.0088	0.0092	0.0053	0.0047	0.0049	0.0034	0.017
SlopeOne	0.0067	0.0073	0.0044	0.0051	0.0035	0.0025	0.032
UNN	0.0071	0.0075	0.0052	0.0063	0.0065	0.0048	0.041
INN	0.0073	0.0065	0.0059	0.0097	0.011	0.0121	0.035

parameters like size of users, size of items, density of rating matrix, rating dispersion[38], Gini coefficients for users and items[12,20,39]. Studying the relationship of η coefficients with the dataset parameters is left for future research. One must, in principle, estimate these regression parameters for a given set before the optimal subsampling analysis in Section 6 can be further pursued. Our experience with modeling TTI and SAL suggests that further analysis may yield constant or common models of SAL that are valid for each method across many if not most datasets.

In this paper, we use the RMSE as a predictive accuracy metric[27]. We expect our insights to carry over to other metrics. In Fig. 7, we show the dependence of another metric, the classification Mean Average Precision (MAP)[27] on the odds ratio FID/(1−FID) or $\nu/(1-\nu)$, when using the proposed subsampling algorithm. The dependence of MAP on $\nu/(1-\nu)$ at a constant FUD was found to be mainly linear which is qualitatively the same as for the RMSE. The MAP metric considers the quality of the top-N recommended items. Figure 7 is based on top 5 items recommended to users based on SVD CF estimated ratings using subsampled matrices from P_1. Moreover, Fig. 7 shows that if one is willing to incur a loss from MAP=0.82 to 0.45, then one can drop as much as 90% of the items. Note that our derived models are valid for FID and FUD as high as 0.9 simultaneously, which is as severe as keeping 1% of the data.

6 Optimal Subsampling of CF Model

In this section, we investigate the use of the proposed simple models of TTI and AL to estimate the best subsampling levels. We are going to initiate this approach based on result for matrix factorization based CF approaches (SVD, SVD_b, and NMF) and CoClustering CF approach. Our proposed simple models for CF efficiency and performance for SVD, SVD_b, NMF, and CoClustering CF approach are summarized below:

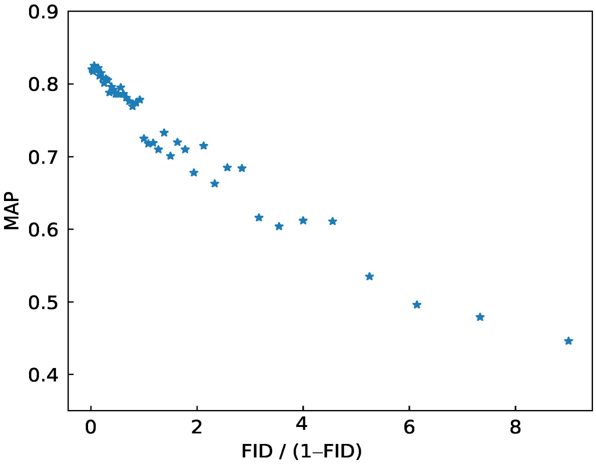

Fig. 7 MAP versus ORDI using P_1 at constant FUD = 0.5 for top 5 recommendations based on SVD estimated ratings.

$$\mathrm{TTI} = \mu + \nu - \mu\nu \tag{18}$$

$$\mathrm{AL} = \frac{\eta_0\mu + \eta_1\nu - \eta_2\mu\nu}{(1-\mu)(1-\nu)} \tag{19}$$

Theorem 1 For a specified accuracy loss AL $= \alpha$, the optimal subsampling levels that maximize the efficiency TTI are given by

$$\begin{aligned} \nu &= \frac{\alpha + \eta_0 - \alpha'}{\alpha + \eta_2}, \\ (\alpha')^2 &= \frac{(\eta_0 - \eta_2)(\alpha\eta_0 + \alpha\eta_1 - \alpha\eta_2 + \eta_0\eta_1)}{(\eta_1 - \eta_2)}, \\ \mu &= \frac{\eta_0 - \eta_1 + (\eta_1 - \eta_2)\nu}{\eta_0 - \eta_2} \end{aligned} \tag{20}$$

and the corresponding TTI is obtained from Eq. (18).

Proof of Theorem 1 is in Appendix A. A dual formulation is shown below:

Theorem 2 For a specified TTI $= \tau$, the optimal subsampling levels that minimize AL are given by

$$\begin{aligned} \nu &= 1 - \left(\frac{(\eta_0 - \eta_2)\ (1-\tau)}{\eta_1 - \eta_2}\right)^{\frac{1}{2}}, \\ \mu &= \frac{\eta_0 - \eta_1 + (\eta_1 - \eta_2)\nu}{\eta_0 - \eta_2} \end{aligned} \tag{21}$$

and the corresponding AL is obtained from Eq. (19). Moreover, the optimal odds ratio for dropping an item, the $O_\nu = \nu/(1-\nu)$ is

$$O_\nu = \sqrt{\frac{\eta_1 - \eta_2}{\eta_0 - \eta_2}}\sqrt{\mathrm{TTRF}} - 1 \tag{22}$$

Proof of Theorem 2 is in Appendix B. Note that Eq. (22) suggests a simple dependence on the square root of TTRF, the training time reduction factor.

We also explored the idea of dropping the same fraction of items and users. This is an intuitively appealing concept, even though it leads to sub-optimal results.

Theorem 3 With a constraint of $\mu = \nu$, and for a specified AL $= \alpha$, the optimal subsampling level that maximizes TTI is given by

$$\begin{aligned} \nu &= \frac{\sigma + \sigma'}{2(\alpha + \eta_2)}, \\ \sigma &= 2\alpha + \eta_0 + \eta_1, \\ (\sigma')^2 &= \eta_0^2 + 2\eta_0\eta_1 + 4\alpha\eta_0 + \eta_1^2 + 4\alpha\eta_1 - 4\alpha\eta_2 \end{aligned} \tag{23}$$

and the corresponding TTI is obtained from Eq. (18).

Proof of Theorem 3 is in Appendix C. A dual optimization problem is shown below. Note that the optimal AL subject to the constraint TTI = τ is equivalent to the optimal AL subject to the constraint TTRF $= 1/(1-\tau)$.

Theorem 4 With a constraint of $\mu = \nu$, and for a specified TTI $= \tau$, the optimal subsampling level that minimizes AL is given by

$$\mu = \nu = 1 - \sqrt{1-\tau} \tag{24}$$

and the corresponding optimum AL is

$$\text{AL} = (\eta_0 + \eta_1)\, O_\nu \sqrt{\text{TTRF}} - \eta_2 O_\nu^2 \propto \text{TTRF} \tag{25}$$

Proof of Theorem 4 is in Appendix D. Note that the optimal values satisfy $\mu = \nu = 1 - 1/\sqrt{\text{TTRF}}$ in this case, and

$$O_\nu = \frac{\nu}{1-\nu} = \sqrt{\text{TTRF}} - 1 \tag{26}$$

which has a particularly appealing interpretation. Moreover, the accuracy loss seems to be proportional to a small fraction of the TTRF for large TTRF as supported by Fig. 8 for regular SVD CF. For different specified levels of AL, we computed the optimal TTRF based on optimal solutions of Theorems 1 and 3. Similarly, for different specified TTRF, we computed the optimal AL based on optimal solutions of Theorems 2 and 4. The results are based on the empirically determined values of η coefficients in Table 5 for SVD CF approach and the plots based on different rating matrices were qualitatively the same. Below, we show only results for the SVD CF approach based on dataset P_1.

The optimal and sub-optimal results are shown in Fig. 8, and they confirm our intuition that constraining the FID and FUD to be equal does not impact the optimality significantly. The closed-form for the sub-optimal results, which shows a strong dependence on $\sqrt{\text{TTRF}}$ is particularly appealing. The optimal TTRF seems to be quadratically dependent on the optimal AL for small AL, and tends to become more linear for large AL.

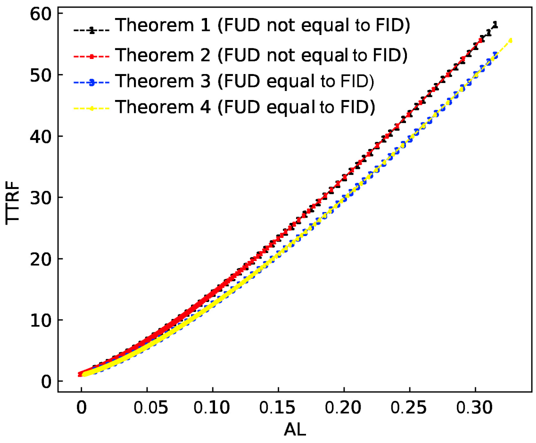

Fig. 8 Plots based on optimal solutions.

As an example, note the that the training time for P_1 was reduced by factor of 10 while losing only about 7% in accuracy. A 40 times improvement in training time can be achieved with about 22% degradation in its accuracy.

7 How Does the Practitioner Use the Proposed Optimal Model

We propose the following guidelines:

(1) Sample the big dataset to derive the coefficients of the AL models.

(2) The TTI model for a CF method does not depend on the dataset, so models proposed in this study can be used directly.

(3) Define the desired level of gain in efficiency.

(4) Derive the optimal FID and FUD using solution in Theorem 2 to obtain desired level of gain in efficiency with minimum possible loss in accuracy for the desired TTI.

(5) Subsample the primary rating matrix with the optimal FID and FUD and apply the training algorithm of interest to the reduced data matrix.

Alternatively, one can define the acceptable accuracy loss in Step 2, and then compute the optimal FID and FUD using the solution in Theorem 1 to get an estimate of the reduction of training time for the tolerated accuracy loss.

Note that as the primary dataset evolves with time, e.g., by adding users and items, it is unlikely that the estimated model coefficients will change appreciably. Hence the optimal subsampling levels are expected to be stable. As the dataset changes, the recommender can be updated by sampling again at the same levels.

Other applications can also be pursued. For instance, in the Movie Lens recommender application, the items can be partitioned per genre, and the users can be partitioned by age or gender, and each partitioned can then be subsampled according to its own optimal rates.

8 Conclusions and Future Work

The proposed DRSC algorithm in Section 3 can be used to subsample a rating matrix while maintaining the density of the subsampled matrix within the tolerance of density of the original matrix. The computational cost of this subsampling algorithm were found to be very negligible compared to the computational savings in the training phase.

Our study showed that the effects of subsampling on the efficiency of CF are constant in the sense that they are largely independent of the CF algorithm and the dataset involved, including the shape of the rating matrix and its density. Indeed the TTI has the same linear regression form for 4 leading CF algorithms and the coefficients are simply 1 in magnitude in terms of the fractions of users and items dropped; namely $\text{TTI} = \mu + \nu - \mu\nu$.

The dependence of AL on subsampling, measured as relative deterioration in the RMSE, is more complicated but it is potentially still reasonably constant. A scaled version of the AL of CF was empirically found to be multi-linear in FUD and FID; but the coefficients of the regression model seem to depend on the characteristics of the rating matrix of the specific dataset considered.

In any case, the simple models developed enabled the development of various optimization problems which were solved in closed-form. The solutions provide guidelines for the optimum level of subsampling balancing the need to save computation time while maintaining accuracy. Ultimately it was shown that dropping the same fraction of users and items that are approximately proportional to the square-root of the desired TTRF is a very good sub-optimal solution.

The above results suggest that there are perhaps some theoretical foundations that make the effects of subsampling predictable by constant (or universal) regression models. We have not attempted to provide such a theoretical development, but the constant model of computational efficiency is rather recognizable. The AL regression models are more opaque, but further study may eventually reveal a constant model.

The applicability of the method to very different rating matrices requires further investigation, but we have already tested our approach to several rating matrices exhibiting a diversity of domain, size, shape, and sparsity, and the results are very promising. It is not clear whether the subsampling recommendations developed here would apply to other machine learning problems beyond recommendation and collaborative filtering. The simplicity of the proposed guidelines for subsampling strongly suggests that these investigations be pursued.

Appendix

A Proof of Theorem 1

Here, we optimize Eq. (18) or $\text{TTI} = \mu + \nu - \mu\nu$ subject to the constraint of

$$\text{AL} = \frac{\eta_0\mu + \eta_1\nu - \eta_2\mu\nu}{(1-\mu)(1-\nu)} = \alpha.$$

Using the Lagrange multipliers method, we define $L = \text{TTI} - \lambda(\text{AL} - \alpha)$. The necessary conditions obtained by taking partial derivatives of L with respect to μ, ν, and λ are as follows:

$$(1-\nu)^2(1-\mu)^2 - \lambda(\eta_0 + \eta_1\nu - \eta_2\nu) = 0 \quad (27)$$

$$(1-\nu)^2(1-\mu)^2 - \lambda(\eta_1 + \eta_0\mu - \eta_2\mu) = 0 \quad (28)$$

$$\eta_0\mu + \eta_1\nu - \eta_2\mu\nu - \alpha(1-\mu)(1-\nu) = 0 \quad (29)$$

Solving Eqs. (27) and (28) leads to

$$\mu = \frac{\eta_0 - \eta_1 + \eta_1\nu - \eta_2\nu}{\eta_0 - \eta_2}.$$

Substituting μ into Eq. (29) will give relation of ν with α leading us to the complete optimal solutions as mentioned in Eq. (20). Substituting μ and ν from Eq. (20) into Eq. (18) will give the optimal .

B Proof of Theorem 2

Here, we optimize Eq. (19) subject to the constraint of $\text{TTI} = \mu + \nu - \mu\nu = \tau$. Using the augmented objective function $L = \text{AL} - \lambda(\text{TTI} - \tau)$, the necessary conditions are

$$\eta_0 + \eta_1\nu - \eta_2\nu - \lambda(1-\nu)^2(1-\mu)^2 = 0 \quad (30)$$

$$\eta_1 + \eta_0\mu - \eta_2\mu - \lambda(1-\nu)^2(1-\mu)^2 = 0 \quad (31)$$

$$\mu + \nu - \mu\nu = \tau \quad (32)$$

Solving Eqs. (30) and (31) leads to

$$\mu = \frac{\eta_0 - \eta_1 + \eta_1\nu - \eta_2\nu}{\eta_0 - \eta_2}.$$

Substituting μ into Eq. (32) will lead to the desired results.

C Proof of Theorem 3

When $\mu = \nu$, Eq. (18) simplifies to

$$\text{TTI} = 2\nu - \nu^2 \quad (33)$$

Also, for $\mu = \nu$, Eq. (19) simplifies to

$$\text{AL} = \frac{\eta_0\nu + \eta_1\nu - \eta_2\nu^2}{(1-\mu)^2} \quad (34)$$

Using the augmented objective function $L = \text{TTI} - \lambda(\text{AL} - \alpha)$, the necessary condition is

$$\frac{\eta_0\nu + \eta_1\nu - \eta_2\nu^2}{(1-\nu)^2} - \alpha = 0 \quad (35)$$

Solving Eq. (35) gives Eq. (23), which is the optimal solution to reach to optimal TTI for specified $\text{AL} = \alpha$ along with the constraint of $\mu = \nu$.

D Proof of Theorem 4

Assuming $\mu = \nu$, we optimize Eq. (34) subject to the constraint TTI$= 2\nu - \nu^2 = \tau$. The augmented objective function is $L = \mathrm{AL} - \lambda(\mathrm{TTI} - \tau)$, and one necessary condition is

$$2\nu - \nu^2 = \tau \tag{36}$$

Solving Eq. (36) gives $\mu = \nu = 1 \pm \sqrt{1-\tau}$. As we expect real roots and μ and ν to be greater than 0 and less than 1, we use $\mu = \nu = 1 - \sqrt{1-\tau}$ as the solution. Equation (24) provides us with the optimal solution to reach optimal AL for specified TTI along with the constraint of $\mu = \nu$. The optimal accuracy loss is then computed as

$$\mathrm{AL} = \frac{\eta_0 \nu + \eta_1 \nu - \eta_2 \nu^2}{(1-\nu)^2}.$$

Substituting $\mu = \nu = 1 - \sqrt{1-\tau}$ will give

$$\mathrm{AL} = \mathrm{TTRF}\left(1 - \frac{1}{\sqrt{\mathrm{TTRF}}}\right) \times \left(\eta_0 + \eta_1 - \eta_2\left(1 - \frac{1}{\sqrt{\mathrm{TTRF}}}\right)\right),$$

$$\mathrm{AL} = O_\nu \sqrt{\mathrm{TTRF}}\left(\eta_0 + \eta_1 - \eta_2 \frac{O_\nu}{\sqrt{\mathrm{TTRF}}}\right).$$

Solving will lead to the optimal AL as given in Eq. (25).

References

[1] L. Sharma and A. Gera, A survey of recommendation system: Research challenges, *Int. J. Eng. Trends Technol.*, vol. 4, no. 5, pp. 1989–1992, 2013.

[2] N. Pereira and S. Varma, Survey on content based recommendation system, *Int. J. Comput. Sci. Inf. Technol*, vol. 7, no. 1, pp. 281–284, 2016.

[3] R. Chen, Q. Y. Hua, Y. S. Chang, B. Wang, L. Zhang, and X. J. Kong, A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks, *IEEE Access*, vol. 6, pp. 64301–64320, 2018.

[4] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, Recommendation systems: Principles, methods and evaluation, *Egypt. Inform. J.*, vol. 16, no. 3, pp. 261–273, 2015.

[5] F. Ortega, J. L. Sánchez, J. Bobadilla, and A. Gutiérrez, Improving collaborative filtering-based recommender systems results using Pareto dominance, *Inform. Sci.*, vol. 239, pp. 50–61, 2013.

[6] C. C. Aggarwal, *Recommender Systems*. Cham, Switzerland: Springer, 2016.

[7] P. B. Thorat, R. M. Goudar, and S. Barve, Survey on collaborative filtering, content-based filtering and hybrid recommendation system, *Int. J. Comput. Appl.*, vol. 110, no. 4, pp. 31–36, 2015.

[8] X. Y. Su and T. M. Khoshgoftaar, A survey of collaborative filtering techniques, *Adv. Artif. Intell.*, vol. 2009, pp. 421–425, 2009.

[9] S. Vucetic and Z. Obradovic, A regression-based approach for scaling-up personalized recommender systems in E-commerce, https://wenku.baidu.com/view/f033010103d8ce2f006623b8.html, 2009.

[10] T. George and S. Merugu, A scalable collaborative filtering framework based on co-clustering, in *Proc. 5th IEEE Int. Conf. on Data Mining*, Houston, TX, USA, 2005, p. 4.

[11] P. H. Aditya, I. Budi, and Q. Munajat, A comparative analysis of memory-based and model-based collaborative filtering on the implementation of recommender system for E-commerce in Indonesia: A case study PT X, in *Proc. 2016 Int. Conf. Advanced Computer Science and Information Systems*, Malang, Indonesia, 2017, pp. 303–308.

[12] G. Adomavicius and J. J. Zhang, Impact of data characteristics on recommender systems performance, *ACM Trans. Manag. Inf. Syst.*, vol. 3, no. 1, p. 3, 2012.

[13] F. Cacheda, V. Carneiro, D. Fernández, and V. Formoso, Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems, *ACM Trans. Web*, vol. 5, no. 1, p. 2, 2011.

[14] Z. Huang, D. Zeng, and H. Chen, A comparison of collaborative-filtering recommendation algorithms for E-commerce, *IEEE Intell. Syst.*, vol. 22, no. 5, pp. 68–78, 2007.

[15] U. Kuelewska, Effect of dataset size on efficiency of collaborative filtering recommender systems with multi-clustering as a neighbourhood identification strategy, in *Proc. 20th Int. Conf. on Computational Science*, Amsterdam, The Netherlands, 2020, pp. 342–354.

[16] J. Lee, M. X. Sun, and G. Lebanon, A comparative study of collaborative filtering algorithms, arXiv preprint arXiv: 1205.3193, 2012.

[17] Z. Yang, B. Wu, K. Zheng, X. B. Wang, and L. Lei, A survey of collaborative filtering-based recommender systems for mobile internet applications, *IEEE Access*, vol. 4, pp. 3273–3287, 2016.

[18] H. Taherdoost, Sampling methods in research methodology; How to choose a sampling technique for research, *Int. J. Acad. Res. Manag.*, vol. 5, no. 2, pp. 18–27, 2016.

[19] J. S. Breese, D. Heckerman, and C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in *Proc. of the Fourteenth Conf. on Uncertainty in Artificial Intelligence*, Madison, WI, USA, 1998, pp. 43–52.

[20] Y. Deldjoo, T. Di Noia, E. Di Sciascio, and F. A. Merra, How dataset characteristics affect the robustness of collaborative recommendation models, in *Proc. of the 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, New York, NY, USA, 2020, pp. 951–960.

[21] A. J. B. Chaney, B. M. Stewart, and B. E. Engelhardt, How algorithmic confounding in recommendation systems increases homogeneity and decreases utility, in *Proc. of the 12th ACM Conf. on Recommender Systems*, Vancouver, Canada, 2017, pp. 224–232.

[22] G. H. Golub and C. Reinsch, Singular value decomposition and least squares solutions, in *Handbook for Automatic Computation*, F. L. Bauer, A. S. Householder, F. W. J. Olver, H. Rutishauser, K. Samelson, and E. Stiefel, eds. Berlin, Germany: Springer, 1971, pp. 134–151.

[23] N. Hug, Surprise: A Python library for recommender systems, *J. Open Source Softw.*, vol. 5, no. 52, p. 2174, 2020.

[24] X. Luo, M. C. Zhou, Y. N. Xia, and Q. S. Zhu, An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems, *IEEE Trans. Ind. Inform.*, vol. 10, no. 2, pp. 1273–1284, 2014.

[25] D. Lemire and A. Maclachlan, SlopeOne predictors for online rating-based collaborative filtering, in *Proc. of the 2005 SIAM Int. Conf. on Data Mining*, Newport Beach, CA, USA: SDM, 2005, p. 5.

[26] Y. Koren, R. Bell, and C. Volinsky, Matrix factorization techniques for recommender systems, *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[27] G. Schröder, M. Thiele, and W. Lehner, Setting goals and choosing metrics for recommender system evaluations, *CEUR Workshop Proc.*, vol. 811, pp. 78–85, 2011.

[28] T. Chai and R. R. Draxler, Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature, *Geosci. Model Dev.*, vol. 7, no. 3, pp. 1247–1250, 2014.

[29] GroupLens, MovieLens 1M Dataset, https://grouplens.org/datasets/movielens/1m/, 2003.

[30] F. M. Harper and J. A. Konstan, The movielens datasets: History and context, *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, p. 19, 2015.

[31] GroupLens, MovieLens 25M Dataset, https://grouplens.org/datasets/movielens/25m/, 2019.

[32] Y. M. Data-set, Webscope-Yahoo Labs, https://webscope.sandbox.yahoo.com/catalog.php?datatype=r&did=1, 2021.

[33] J. M. Ni, J. C. Li, and J. McAuley, Justifying recommendations using distantly-labeled reviews and fine-grained aspects, in *Proc. of the 2019 Conf. on Empirical Methods in Natural Language Proc. and the* 9^{th} *Int. Joint Conf. on Natural Language Proc.*, Hong Kong, China, 2019, pp. 188–197.

[34] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al., SciPy 1.0: Fundamental algorithms for scientific computing in Python, *Nat. Methods*, vol. 17, no. 3, pp. 261–272, 2020.

[35] K. Kumari and S. Yadav, Linear regression analysis study, *J. Pract. Cardiovasc. Sci.*, vol. 4, no. 1, pp. 33–36, 2018.

[36] M. Lacey, ANOVA for Regression, http://www.stat.yale.edu/Courses/1997-98/101/anovareg.htm, 2021.

[37] Fisher's, Fisher's method, https://en.wikipedia.org/wiki/Fisher%27smethod, 2021.

[38] M. Sun, How does the variance of product ratings matter? *Manag. Sci.*, vol. 58, no. 4, pp. 696–707, 2011.

[39] S. Chong and A. Abeliuk, Quantifying the effects of recommendation systems, in *Proc. of the 2019 IEEE Int. Conf. on Big Data*, Los Angeles, CA, USA, 2019, pp. 3008–3015.

Marwan Bikdash received the MEng and PhD degrees in electrical engineering from Virginia Tech in 1990 and 1993, respectively. He is currently a professor and the chair of the Department of Computational Data Science and Engineering, North Carolina A&T State University. He teaches and conducts research in signals and systems, computational intelligence, and modeling and simulations of systems with applications in health, energy, and engineering. He has authored over 140 journal and conference papers. He has supported, advised, and graduated over 50 master and PhD students. His projects have been funded by the Jet Propulsion Laboratory, Defense Threat Reduction Agency, Army Research Lab, NASA, National Science Foundation, the Office of Naval Research, Boeing Inc., Hewlett Packard, National Renewable Energy Laboratories, the Army Construction Engineering Research Laboratory, and others.

Samin Poudel is currently a PhD candidate at the Department of Computational Data Science and Engineering, North Carolina A&T State University. His research interests include but not limited to data analytics, data mining, machine learning, developing models, and optimizing techniques based on data.