Computational Statistics
https://doi.org/10.1007/s00180-022-01242-3

ORIGINAL PAPER

Model aggregation for doubly divided data with large size and large dimension

Baihua He[1] · Yanyan Liu[1] · Guosheng Yin[2] · Yuanshan Wu[3]

Received: 7 April 2021 / Accepted: 23 May 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract

Massive data are often featured with high dimensionality as well as large sample size, which typically cannot be stored in a single machine and thus make both analysis and prediction challenging. We propose a distributed gridding model aggregation (DGMA) approach to predicting the conditional mean of a response variable, which overcomes the storage limitation of a single machine and the curse of high dimensionality. Specifically, on each local machine that stores partial data of relatively moderate sample size, we develop the model aggregation approach by splitting predictors wherein a greedy algorithm is developed. To obtain the optimal weights across all local machines, we further design a distributed and communication-efficient algorithm. Our procedure effectively distributes the workload and dramatically reduces the communication cost. Extensive numerical experiments are carried out on both simulated and real datasets to demonstrate the feasibility of the DGMA method.

Keywords Communication efficiency · Computation complexity · Distributed algorithm · Greedy algorithm · High dimension · One-shot approach · Prediction · Storage ability

✉ Yuanshan Wu
wu@zuel.edu.cn

Baihua He
hebaihua@whu.edu.cn

Yanyan Liu
liuyy@whu.edu.cn

Guosheng Yin
gyin@hku.hk

[1] School of Mathematics and Statistics, Wuhan University, Wuhan, Hubei 430072, China

[2] Department of Statistics and Actuarial Science, University of Hong Kong, Pokfulam Road, Hong Kong, China

[3] School of Statistics and Mathematics, Zhongnan University of Economics and Law, Wuhan, Hubei 430073, China

Published online: 16 June 2022

1 Introduction

Explosive growth in the size and dimensionality of modern datasets has ignited enormous interest in statistical learning such as parameter estimation, inference, and prediction. Typically, the storage and analysis of such big data cannot be conducted on a single machine due to the limitation in the storage and computational capability, which brings great challenges as well as new opportunities. With platforms such as Hadoop (Shvachko et al. 2010) or Spark (Zaharia et al. 2010), the massive dataset can be split into smaller pieces and stored on multiple local machines, where the conventional methods can be applied as usual. The key issue is how to combine the analysis results from each local machine to draw a valid global conclusion. In this regard, both the communication rounds and computational complexity are important factors that should be taken into consideration.

In handling massive data, the divide-and-conquer (DC) method is the most commonly used strategy to alleviate the computational burden, which divides the data into smaller pieces and stores them on multiple local machines with one piece of dataset on each machine, and then a simple average of the local results is taken as the final solution. Relevant work include variable selection (Chen and Xie 2014), nonparametric regression (Zhang et al. 2013; Chen et al. 2016; Zhao et al. 2016), and bootstrap inference (Kleiner et al. 2014). Moreover, Lee et al. (2015) considered the high-dimensional sparse linear regression by combining the local debiased Lasso estimators (van de Geer et al., 2014). Rosenblatt and Nadler (2016) studied the asymptotically exact expression for the averaged estimation error under two large-sample regimes. Shang and Cheng (2017) explored the statistical versus computational trade-off of the DC method for nonparametric smoothing spline. Battey et al. (2018) established the theoretical upper bound on the number of local machines such that the information loss due to the DC algorithm is negligible. However, each local machine must have access to at least the squared root of the total sample size in order to achieve the asymptotic error rate; this requirement alone could exceed the storage capability of a single machine, especially for massive data with huge sample size. Moreover, the DC method takes a simple linear combination of all local results, which may result in efficiency loss when a single machine only has access to relatively small sample size. The overall combination result may further deteriorate for the non-additive structure of the estimation procedure.

From the computational perspective, the distributed alternating direction method of multipliers (ADMM) (Boyd et al. 2011) has been shown to be an effective tool for analyzing massive data. The ADMM recasts the infeasible global optimization to a distributed multi-machine optimization problem. Employing the ADMM algorithm, Mateos et al. (2010) considered the distributed sparse linear regression, and Zhang et al. (2012) studied the distributed classification. The ADMM distributed algorithms typically impose the consensus constraints between the global and local solutions, such that the rate of the local estimator converging to the global one is slower than linearity. As an alternative, the gradient-based optimization method can also be distributed over multiple machines. For example, Johnson and Zhang (2013) introduced an explicit variance reduction method for stochastic gradient descent. Jakovetic et al. (2014) proposed two fast distributed gradient algorithms based on the centralized Nesterov gradient algorithm and established their convergence rates. Shamir et al. (2014) proposed a novel Newton algorithm for the distributed optimization. At every iteration, each local machine leverages the second-order information, resulting in more efficient communication than the first-order approaches. Furthermore, using a surrogate loss function to substitute for the global one, Wang et al. (2017) and Jordan et al. (2019) developed a distributed algorithm for a surrogate loss function using the local gradient information. Their methods can reach the estimation error bounds of the oracle global estimator within several iterations and thus strike a balance between computation and communication.

All the aforementioned methods rely upon correct model specification. However, model misspecification often occurs in practice, especially when dealing with massive data. As a remedy, the model averaging approach is often used to combine the strength of multiple candidate models and mitigate the risk of misspecification. The major focus of model averaging is to determine appropriate weights for candidate models. The Bayesian model averaging (Raftery et al. 1997; Hoeting et al. 1999; Eklund and Karlsson 2007) assigns the posterior model probabilities to the corresponding candidate models. Buckland et al. (1997) constructed model averaging weights based on the AIC or BIC scores, which was further elaborated by Burnham and Anderson (2003). There is also a large body of literature on model averaging from the frequentist perspective, for example, the forecast model averaging (Newbold and Granger 1974), frequentist model averaging (Hjort and Claeskens 2003), Mallows' C_p model averaging (Hansen 2007; Wan et al. 2010), the optimal mean squared error averaging (Liang et al. 2011), the optimal model averaging for linear mixed-effects models (Zhang et al. 2014) and jackknife model averaging (Hansen and Racine 2012).

For the high-dimensional data that can be handled by a single machine, Ando and Li (2014) proposed the delete-one cross-validation model averaging procedure with linear regression, which is further extended to the high-dimensional generalized linear regression models (Ando and Li 2017). Adopting the sure independent screening (SIS) procedure (Fan and Lv 2008), predictors that are less marginally correlated with the response can be initially screened out so as to reduce the dimensionality of optimization. However, such SIS-based dimension reduction methods may also remove some predictors that are truly associated with the response, and it is not clear how to choose the cutpoint for determining the number of predictors to be kept. For massive data of high dimensionality and huge sample size that are distributed and stored on multiple local machines, the model averaging procedure becomes more challenging whether from the perspectives of theoretical development, distributional communication, or computational realization.

On the other hand, the model aggregation approach has been mainly studied in the machine learning community to enhance prediction accuracy. Instead of combining all candidate models under consideration as in the model averaging procedure, the model aggregation approach recruits one model from the candidate set at each iteration by elaborately designing an effective greedy algorithm. As a result, the model aggregation approach can strike a balance between prediction accuracy and computational complexity. We propose a distributed gridding model aggregation (DGMA) procedure for predicting the conditional mean of the response with massive data. Specifically, on each machine with access to relatively small sample size, we develop a greedy model aggregation algorithm to bypass the practical issues arising from the SIS procedure. Instead of ranking predictors as in Ando and Li (2014), Ando and Li (2017), we rank candidate models wherein the fitness of each candidate model is assessed through iterations in the greedy model aggregation algorithm. We further define a surrogate loss function with a penalty and design a distributed algorithm inspired by Wang et al. (2017) and Jordan et al. (2019). As the surrogate loss function only involves data on the master machine and the gradients on the local machines, in each iteration the master machine broadcasts the iteration value to the local ones while the locals transfer the gradients to the master. Using such an iteration-communication procedure, the DGMA approach delivers promising performance which is comparable to the oracle global approach with access to the full data on a single machine.

The remaining of the article is organized as follows. In Section 2, we propose the DGMA approach to handling massive data with large size and large dimension. In Section 3, two algorithms, one for communication and the other for computation, are developed and the counterparts for the oracle global method are also introduced as a benchmark. We conduct extensive simulation studies in Section 4 to assess the performance of the proposed method and make comparisons with the DC and oracle global methods. Our approach is further illustrated with two real examples in Section 5, and Section 6 concludes with some remarks. Some theoretical results are collected in the online supplementary material.

2 Distributed gridding model aggregation

Let $\{\mathbf{x}_i, y_i\}_{i=1}^N$ be N independent and identically distributed (i.i.d.) copies of $\{\mathbf{x}, y\}$, where y is the response and $\mathbf{x} = \left(x_1, \dots, x_p\right)^{\mathrm{T}}$ is the p-dimensional predictor. Assume both the sample size N and the dimension p are too large for a single machine to store such a massive dataset. Conventionally, N samples are evenly divided and stored in J local machines. Without loss of generality, suppose the j-th machine has access to dataset $\{\mathbf{x}_{ij}, y_{ij}\}_{i=1}^n$, where $j = 1, \dots, J$ and $n = N/J$. The first machine is designated as the master and the others as local ones. Our goal is to predict the conditional mean of the response given predictors by utilizing the entire dataset that are stored in such a distributed manner as well as to control the workload of communications among machines at a proper level.

For ease of exposition, we first introduce notation. Let $\{A_k : k = 1, \dots, K_n\}$ be a family of sets with each element A_k being a nonempty subset of $[p]$, where $[p]$

denotes the set $\{1, \dots, p\}$, and K_n is some positive integer depending on n. Furthermore, $|A_k|$, the cardinality of A_k, is assumed to be much smaller than the sample size n for $k = 1, \dots, K_n$. For a p-dimensional vector $\mathbf{a} = (a_1, \dots, a_p)^{\mathrm{T}}$, let $\mathbf{a}_{(k)}$ denote the subvector of $\mathbf{a}$ indexed by set A_k.

We focus on the j-th local machine which only has access to the j-th partial dataset, $\{\mathbf{x}_{ij}, y_{ij}\}_{i=1}^{n}$. For the p-dimensional predictor $\mathbf{x}_{ij}$, we can partition it according to index sets $A_1, \dots, A_{K_n}$ and thus obtain $\mathbf{x}_{ij(1)}, \dots, \mathbf{x}_{ij(K_n)}$ correspondingly. By regressing y_{ij} on $\mathbf{x}_{ij(k)}$, $i = 1, \dots, n$, which underlies the k-th submodel on the j-th local machine, the least squares estimator (LSE) is given by

$$\widehat{\boldsymbol{\beta}}_{j(k)} = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{|A_k|}} \frac{1}{n} \sum_{i=1}^{n} \left(y_{ij} - \mathbf{x}_{ij(k)}^{\mathrm{T}} \boldsymbol{\beta} \right)^2,$$

where it implicitly requires that the dimension of $\mathbf{x}_{ij(k)}$, $|A_k|$, is relatively small compared with sample size n. For simplicity, we denote the k-th submodel on the j-th local machine as the (j, k)-th submodel. Obviously, not only is the massive dataset divided to obtain sub-datasets of smaller sample sizes, but it is further divided according to the dimension of predictors. Essentially, we doubly divide the massive data into a matrix of $J \times K_n$ pieces as illustrated in Fig. 1, so that traditional statistical methods would be directly applicable to each piece. We investigate how to combine the piecewise information together to predict the conditional mean of the

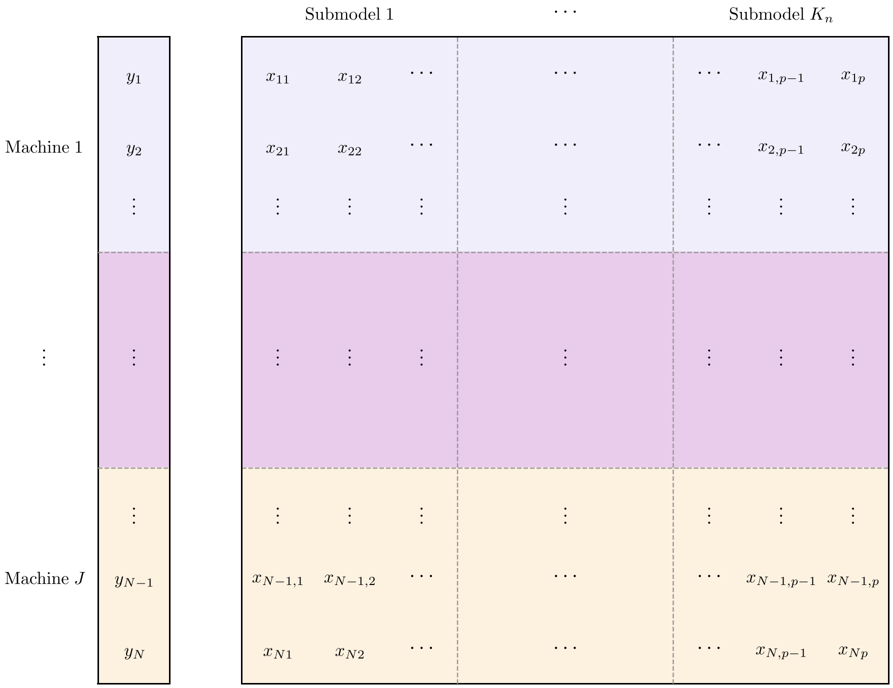
Submodel 1
⋯
Submodel K_n
Machine 1
⋮
Machine J
y_1
y_2
y_{N-1}
y_N
x_{11} x_{12} ⋯ ⋯ ⋯ $x_{1,p-1}$ x_{1p}
x_{21} x_{22} ⋯ ⋯ ⋯ $x_{2,p-1}$ x_{2p}
$x_{N-1,1}$ $x_{N-1,2}$ ⋯ ⋯ ⋯ $x_{N-1,p-1}$ $x_{N-1,p}$
x_{N1} x_{N2} ⋯ ⋯ ⋯ $x_{N,p-1}$ x_{Np}

Fig. 1 Illustration of the gridding approach to doubly dividing the massive data into $J \times K_n$ pieces

response. Instead of naively taking a simple average as in the DC method, we develop a weighted average method to enhance the prediction accuracy. For ease of exposition, we denote $\widehat{y}_{ij(k)} = \mathbf{x}_{ij(k)}^{\mathrm{T}}\widehat{\boldsymbol{\beta}}_{j(k)}$, the estimated response based on the (j, k)-th working submodel, and then $\widehat{\mathbf{y}}_{ij} = \left(\widehat{y}_{ij(1)}, \ldots, \widehat{y}_{ij(K_n)}\right)^{\mathrm{T}}$ represents the vector of estimated responses from the j-th machine. Let $\boldsymbol{\Omega}_n$ be the K_n-dimensional simplex defined as

$$\boldsymbol{\Omega}_n = \left\{ \boldsymbol{\omega} = (\omega_1, \ldots, \omega_{K_n})^{\mathrm{T}} \in [0,1]^{K_n} \,:\, \sum_{k=1}^{K_n} \omega_k = 1 \right\},$$

which collects all the weights assigned to the submodels. For any $\boldsymbol{\omega} \in \boldsymbol{\Omega}_n$, let $\widehat{y}_{ij}(\boldsymbol{\omega}) = \boldsymbol{\omega}^{\mathrm{T}}\widehat{\mathbf{y}}_{ij}$ denote the estimated response based on the weighted combination across the K_n submodels. To choose weights in $\boldsymbol{\Omega}_n$ such that the prediction error is minimized, we define the quadratic loss on the j-th local machine as

$$\mathcal{L}_j(\boldsymbol{\omega}) = \frac{1}{n}\sum_{i=1}^{n}\left\{y_{ij} - \widehat{y}_{ij}(\boldsymbol{\omega})\right\}^2, \quad j = 1, \ldots, J.$$

Therefore, the total loss function is naturally given by

$$\mathcal{T}(\boldsymbol{\omega}) = \frac{1}{J}\sum_{j=1}^{J}\mathcal{L}_j(\boldsymbol{\omega}) = \frac{1}{Jn}\sum_{j=1}^{J}\sum_{i=1}^{n}\left\{y_{ij} - \widehat{y}_{ij}(\boldsymbol{\omega})\right\}^2.$$

Typically, the size of the submodel becomes larger when the dimension of predictors is higher. Thus, we introduce a penalty to accommodate the high-dimensional setting, i.e., the large p case. Let $\boldsymbol{\pi} \in \boldsymbol{\Omega}_n$ be a given prior, and we adopt a special case of the entropy penalty (Dai et al. 2012) as

$$\mathcal{K}(\boldsymbol{\omega}, \boldsymbol{\pi}) = -\sum_{k=1}^{K_n}\omega_k \log\left(\pi_k\right). \tag{2.1}$$

We refer to it as a linear penalty. Intuitively, we give less penalty if we have more confidence on the prior information assigned to the submodels Therefore, the optimal weight is given by

$$\widehat{\boldsymbol{\omega}}_{\mathrm{opt}} = \arg\min_{\boldsymbol{\omega}\in\boldsymbol{\Omega}_n}\{\mathcal{T}(\boldsymbol{\omega}) + \alpha\mathcal{K}(\boldsymbol{\omega}, \boldsymbol{\pi})\}, \tag{2.2}$$

where $\alpha > 0$ is the tuning parameter. However, the optimal weight $\widehat{\boldsymbol{\omega}}_{\mathrm{opt}}$ cannot be obtained under the current setting, because the massive dataset is stored in a distributed manner over multiple machines. We thus develop a distributed algorithm to approximate $\widehat{\boldsymbol{\omega}}_{\mathrm{opt}}$ while controlling the communication cost.

3 Distributed algorithm

Taylor's expansion of $\mathcal{T}(\boldsymbol{\omega})$ around $\boldsymbol{\omega}^\dagger$ yields that

$$\mathcal{T}(\boldsymbol{\omega}) = \mathcal{T}(\boldsymbol{\omega}^\dagger) + \langle \nabla\mathcal{T}(\boldsymbol{\omega}^\dagger), \boldsymbol{\omega} - \boldsymbol{\omega}^\dagger \rangle + \frac{1}{2}\nabla^2\mathcal{T}(\boldsymbol{\omega}^\dagger)(\boldsymbol{\omega} - \boldsymbol{\omega}^\dagger)^{\otimes 2}, \tag{3.1}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product and $\otimes$ denotes the Kronecker product. Apparently, unlike the K_n-dimensional gradient vector $\nabla\mathcal{T}(\boldsymbol{\omega}^\dagger)$, the second-order derivatives require communications of order $O(K_n^2)$ among machines. Motivated by Shamir et al. (2014), Wang et al. (2017) and Jordan et al. (2019), we define a surrogate loss function by replacing the global second-order derivatives with counterparts of the master machine with $j = 1$. Specifically, by noting that

$$\begin{aligned} &\mathcal{T}(\boldsymbol{\omega}^\dagger) + \langle \nabla\mathcal{T}(\boldsymbol{\omega}^\dagger), \boldsymbol{\omega} - \boldsymbol{\omega}^\dagger \rangle + \frac{1}{2}\nabla^2\mathcal{L}_1(\boldsymbol{\omega}^\dagger)(\boldsymbol{\omega} - \boldsymbol{\omega}^\dagger)^{\otimes 2} \\ &\quad = \mathcal{T}(\boldsymbol{\omega}^\dagger) - \mathcal{L}_1(\boldsymbol{\omega}^\dagger) + \mathcal{L}_1(\boldsymbol{\omega}) + \langle \nabla\mathcal{T}(\boldsymbol{\omega}^\dagger) - \nabla\mathcal{L}_1(\boldsymbol{\omega}^\dagger), \boldsymbol{\omega} - \boldsymbol{\omega}^\dagger \rangle, \end{aligned} \tag{3.2}$$

we define the surrogate loss function as

$$\begin{aligned} \mathcal{S}(\boldsymbol{\omega}, \boldsymbol{\omega}^\dagger) =& \mathcal{L}_1(\boldsymbol{\omega}) + \langle \nabla\mathcal{T}(\boldsymbol{\omega}^\dagger) - \nabla\mathcal{L}_1(\boldsymbol{\omega}^\dagger), \boldsymbol{\omega} \rangle \\ =& \mathcal{L}_1(\boldsymbol{\omega}) + \left\langle \frac{1}{J}\sum_{j=1}^{J} \nabla\mathcal{L}_j(\boldsymbol{\omega}^\dagger) - \nabla\mathcal{L}_1(\boldsymbol{\omega}^\dagger), \boldsymbol{\omega} \right\rangle, \end{aligned}$$

which neglects some constants. The difference between the surrogate loss and the total loss can be evaluated by

$$\begin{aligned} \mathcal{S}(\boldsymbol{\omega}, \boldsymbol{\omega}^\dagger) - \mathcal{T}(\boldsymbol{\omega}) =& - \mathcal{T}(\boldsymbol{\omega}^\dagger) + \mathcal{L}_1(\boldsymbol{\omega}^\dagger) + \langle \nabla\mathcal{T}(\boldsymbol{\omega}^\dagger) - \nabla\mathcal{L}_1(\boldsymbol{\omega}^\dagger), \boldsymbol{\omega}^\dagger \rangle \\ &+ O_P\big(n^{-1/2}\|\boldsymbol{\omega} - \boldsymbol{\omega}^\dagger\|_2^2\big), \end{aligned}$$

where $\| \cdot \|_2$ is the Euclidean norm. It is expected that the surrogate loss would produce weights as a reasonable approximation to the optimal weights.

The recursions of our method consist of solving the surrogate loss function on the master machine and exchanging gradient information between the local machines and the master. At iteration $t = 0$, we set an initial value $\widehat{\boldsymbol{\omega}}^{(0)}$ on the master machine and broadcast it to all the local machines. Consequently, each local machine computes $\nabla\mathcal{L}_j(\widehat{\boldsymbol{\omega}}^{(0)})$ and communicates this gradient back to the master machine. After computing its own gradient $\nabla\mathcal{L}_1(\widehat{\boldsymbol{\omega}}^{(0)})$ and summarizing all gradients, the master machine conducts the minimization of the surrogate loss function with a penalty. This constitutes one round of communication, as detailed in Algorithm 1. Repeat the procedure until some convergence criterion is met.

Algorithm 1: Distributed Gridding Model Aggregation (DGMA) Algorithm

```
**Input**: Data $\{\mathbf{x}_{ij}, y_{ij}\}_{i\in[n],j\in[J]}$.
**Initialization**: Obtain the LSE $\widehat{\boldsymbol{\beta}}_{j(k)}$ under each submodel on every machine and $\widehat{\boldsymbol{\omega}}^{(0)}$.
**for** $t = 0, 1, \ldots,$ **do**
    1. Receive $\widehat{\boldsymbol{\omega}}^{(t)}$ from the master, then calculate gradient $\nabla\mathcal{L}_j(\widehat{\boldsymbol{\omega}}^{(t)})$ on every local machine and send it back to the master.
    2. Receive $\{\nabla\mathcal{L}_j(\widehat{\boldsymbol{\omega}}^{(t)})\}_{j=1}^{J}$ from all the local machines, then solve the shifted penalized optimization problem,
       $\widehat{\boldsymbol{\omega}}^{(t+1)} = \arg\min_{\boldsymbol{\omega}\in\boldsymbol{\Omega}_n}\{\mathcal{S}(\boldsymbol{\omega}, \widehat{\boldsymbol{\omega}}^{(t)}) + \alpha_{t+1}\mathcal{K}(\boldsymbol{\omega}, \boldsymbol{\pi})\},$    (3.3)
       on the master machine, and broadcast $\widehat{\boldsymbol{\omega}}^{(t+1)}$ to every local machine.
    If $\|\widehat{\boldsymbol{\omega}}^{(t+1)} - \widehat{\boldsymbol{\omega}}^{(t)}\|_\infty \leq \varepsilon$, then break the loop, where $\|\cdot\|_\infty$ is the supremum norm and $\varepsilon = 0.0001$.
**end for**
```

The tuning parameter α_{t+1} in (3.3) is chosen in such a way that it decreases with the iteration number t. Typically, K_n is larger for higher dimension p, which may cause failure of the classical algorithms for the minimization in (3.3). Motivated by Dai et al. (2012), we develop a greedy algorithm as follows.

Algorithm 2: Greedy Algorithm for (3.3)

```
**Initialization**: $\widehat{\boldsymbol{\omega}}_{(0)}^{(t)} = \widehat{\boldsymbol{\omega}}^{(t)}$,
**for** $\ell = 1, 2, \ldots,$ **do**
    $\lambda_{(\ell)} = 2/(\ell+1)$,
    $\widehat{k}_{(\ell)} \in \arg\min_{k\in[K_n]} \left\{ \mathcal{S}\left(\widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)} + \lambda_{(\ell)}\left(\mathbf{e}_k - \widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)}\right), \widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)}\right) + \alpha_{t+1}\mathcal{K}\left(\widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)} + \lambda_{(\ell)}\left(\mathbf{e}_k - \widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)}\right), \boldsymbol{\pi}\right)\right\}$,
    where $\mathbf{e}_k$ is a $K_n$-dimensional vector of 0's except for the $k$-th element being 1.
    $\widehat{\boldsymbol{\omega}}_{(\ell)}^{(t)} = \widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)} + \lambda_{(\ell)}\left(\mathbf{e}_{\widehat{k}_{(\ell)}} - \widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)}\right)$,
    If $\left\langle -\nabla\mathcal{S}\left(\widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)}, \widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)}\right) - \alpha_{t+1}\nabla\mathcal{K}\left(\widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)}, \boldsymbol{\pi}\right), \mathbf{e}_k - \widehat{\boldsymbol{\omega}}_{(\ell-1)}^{(t)}\right\rangle \le \varepsilon$, then break the loop.
**end for**
```

Based on the greedy algorithm, we can obtain an approximate estimator $\widehat{\boldsymbol{\omega}}_{(\ell)}^{(t)}$ for $\widehat{\boldsymbol{\omega}}^{(t+1)}$ in each round. Furthermore, after ℓ iterations, this algorithm returns a vector $\widehat{\boldsymbol{\omega}}_{(\ell)}^{(t)}$ which has at most ℓ nonzero components. It thus explicitly controls the size of submodels and strikes a balance between prediction accuracy and computation complexity.

As a benchmark, the oracle global method corresponds to the ideal situation where the entire dataset can be stored and analyzed in a single mega-machine and thus can achieve the optimal convergence rate. Despite its infeasibility in practice, the greedy algorithm for (2.2) is described as follows.

Algorithm 3: Greedy Algorithm for (2.2)

```
Initialization: Choose $\widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(0)} \in \boldsymbol{\Omega}_n$,
for $\ell = 1, 2, \ldots,$ do
    $\lambda^{\mathrm{g}}_{(\ell)} = 2/(\ell + 1)$,
    $\widehat{k}^{\mathrm{g}}_{(\ell)} \in \arg\min_{k \in [K_n]} \Big\{ \mathcal{T}\left(\widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)} + \lambda^{\mathrm{g}}_{(\ell)}\left(\mathbf{e}_k - \widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)}\right)\right)$
            $+ \alpha \mathcal{K}\left(\widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)} + \lambda^{\mathrm{g}}_{(\ell)}\left(\mathbf{e}_k - \widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)}\right), \boldsymbol{\pi}\right)\Big\}$,
    $\widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell)} = \widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)} + \lambda^{\mathrm{g}}_{(\ell)}\left(\mathbf{e}_{\widehat{k}^{\mathrm{g}}_{(\ell)}} - \widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)}\right)$,
    If $\left\langle -\mathcal{T}\left(\widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)}\right) - \alpha \nabla \mathcal{K}\left(\widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)}, \boldsymbol{\pi}\right), \mathbf{e}_{\widehat{k}^{\mathrm{g}}_{(\ell)}} - \widehat{\boldsymbol{\omega}}^{\mathrm{g}}_{(\ell-1)}\right\rangle \leq \varepsilon$, then break the
    loop.
end for
```

4 Simulation studies

We conduct extensive simulation studies to evaluate the finite-sample performance of the distributed gridding model aggregation (DGMA) approach. For comparison, we also consider the divide-and-conquer model averaging (DCMA) approach where the optimal weights are obtained by taking an average over all the weights from the local machines. The DCMA approach is new in the context of model averaging for massive data and has its own interest. As a benchmark, we make comparisons with the oracle global method, for which the entire dataset can be stored and analyzed in a single super machine and thus the optimal weights can be obtained by directly optimizing (2.2) using Algorithm 3. The oracle global approach, although infeasible in practice, is expected to deliver the best performance in simulations.

We generate $\mathbf{x}_i, i = 1, \ldots, N$, independently from a p-variate normal distribution with mean zero and covariance matrix $\boldsymbol{\Sigma} = (0.5^{|r-s|})$ for $r, s = 1, \ldots, p$. The response variable y_i is then obtained via the linear model,

$$y_i = \mathbf{x}_i^{\mathrm{T}} \boldsymbol{\beta}^* + \epsilon_i, \tag{4.1}$$

where the error ϵ_i is independently generated from $N(0, 0.5)$ and the first 15 entries of coefficient $\boldsymbol{\beta}^*$ are independently drawn from $N(0, 0.5)$ (and then fixed throughout simulations) and the others are set to be 0. We consider $N = 4000$ and 8000 and $p = 2000$. For each scenario, we randomly divide the data into subsets with an equal sample size $n = 100$ or 400 and allocate them into local machines.

At the initial step, we perform the SIS procedure (Fan and Lv 2008) to rank the 2000 predictors and then group every 10 predictors together to formulate the candidate submodels. This leads to a total of $K_n = 200$ candidate submodels, while none of them contains the true model. Moreover, we also consider the situation in which the true model is contained in the candidate submodels. In particular, we consider $K_n = 100$ and perform the SIS procedure only on the 1985 non-significant predictors to obtain the ordered predictors. The top five predictors and 15 significant ones are grouped together to formulate one candidate submodel. The remaining 1980 predictors are evenly grouped to formulate the other 99 candidate submodels.

We consider three types of priors $\boldsymbol{\pi}$ for the candidate submodels. The first one is an equal constant prior wherein each component of $\boldsymbol{\pi}$ is $1/K_n$. Obviously, the equal noninformative prior leads the regularization term in (2.1) to be a constant and thus makes our method free of the tuning parameter α. The second is a uniform prior, for which we independently generate K_n uniform variates on (0, 1) and set each component of $\boldsymbol{\pi}$ to be a normalized uniform variate (divided by the summation of K_n uniform variates). The third is an informative prior, for which we sort the uniform variates in a descending order, and assign more prior weights to the more effective submodels based on the ranking.

Considering that our datasets naturally consist of J-fold subsets in a distributed framework, we develop a leave-one-machine-out cross validation (CV) procedure to select the tuning parameter α. Specifically, let $\widehat{\boldsymbol{\omega}}^{(-j)}(\alpha)$ denote the resulting estimator based on the DGMA approach by leaving the j-th machine out and define the CV prediction error on the j-th machine as

$$\mathrm{CV}_j(\alpha) = \frac{1}{n}\sum_{i=1}^{n}\left(y_{ij} - \sum_{k=1}^{K_n}\widehat{\omega}_k^{(-j)}(\alpha)\mathbf{x}_{ij(k)}^{\mathrm{T}}\widehat{\boldsymbol{\beta}}_{j(k)}\right)^2.$$

Consequently, the optimal tuning parameter is given by

$$\widehat{\alpha}_{\mathrm{opt}} = \arg\min_{\alpha}\frac{1}{J}\sum_{j=1}^{J}\mathrm{CV}_j(\alpha),$$

where there may exist multiple optimal solutions. Noting that the objective function is separable and to further avoid the out-of-memory issue, we adopt the gridding method to find the optimal $\widehat{\alpha}_{\mathrm{opt}}$. Moreover, the oracle global approach can also use this optimal tuning parameter because it can be considered as the solution of a conventional J-fold CV criterion. The optimal tuning parameter for the DCMA approach can be selected similarly with $\widehat{\boldsymbol{\omega}}^{(-j)}(\alpha)$ obtained using the DCMA approach.

A testing dataset with sample size 1000 is generated from (4.1) to evaluate the performances of three approaches in terms of the mean squared error (MSE) for prediction. Under each configuration, we repeat 500 simulations and present the averaged MSE of prediction and its associated standard deviation (SD). Table 1 shows the simulation results when the equal constant prior is used for $\boldsymbol{\pi}$. The effectiveness of the proposed DGMA algorithm can be fully demonstrated by its quick convergence within only four communications to a stable plateau associated with

Table 1 The mean squared error (MSE) for the prediction of the response averaged over 500 simulations and the corresponding standard deviation (SD) in the testing dataset with the equal constant prior

n	J	K_n		DGMA				DCMA	Oracle
				$t = 1$	$t = 2$	$t = 3$	$t = 4$		
100	40	200	MSE	0.7399	0.6653	0.6477	0.6462	0.6840	0.2556
			SD	0.1362	0.1191	0.1154	0.1161	0.1267	0.0119
		100	MSE	0.3258	0.2718	0.2556	0.2545	0.2643	0.2451
			SD	0.0630	0.0221	0.0139	0.0137	0.0116	0.0116
400	10	200	MSE	0.3364	0.3264	0.3260	0.3261	0.3265	0.2556
			SD	0.1371	0.1304	0.1302	0.1302	0.1301	0.0119
		100	MSE	0.2560	0.2460	0.2454	0.2454	0.2463	0.2451
			SD	0.0178	0.0115	0.0114	0.0114	0.0114	0.0116
100	80	200	MSE	0.7376	0.6652	0.6508	0.6492	0.6865	0.2560
			SD	0.1231	0.1062	0.1033	0.1032	0.1127	0.0115
		100	MSE	0.3228	0.2705	0.2553	0.2540	0.2637	0.2451
			SD	0.0604	0.0211	0.0129	0.0132	0.0112	0.0112
400	20	200	MSE	0.3333	0.3236	0.3232	0.3233	0.3236	0.2560
			SD	0.1362	0.1298	0.1295	0.1295	0.1293	0.0115
		100	MSE	0.2552	0.2460	0.2454	0.2454	0.2461	0.2451
			SD	0.0171	0.0112	0.0111	0.0111	0.0111	0.0112

decreasing standard derivations. In general, DGMA exhibits superior performances over DCMA, while both are outperformed by the oracle global approach. Similar conclusions can be drawn from Tables 2 and 3, which respectively summarize simulation results for the uniform and sorted uniform priors of $\boldsymbol{\pi}$.

Figure 2 shows the MSEs of prediction for scenarios with sample size $N = 4000$. For the case with $K_n = 200$ where the true model is not included in the candidate submodels, DGMA and DCMA exhibit similar performances when the number of local machines is $J = 10$ or, equivalently when each machine stores partial data of adequate size. However, the advantage of DGMA over DCMA is amplified when datasets are distributed over a larger number of local machines ($J = 40$), demonstrating the power of the distributed algorithm. For the case with $K_n = 100$ where the true model is indeed included in the candidate set, the MSEs of prediction under the three approaches decrease dramatically, indicating that all of them can gain greatly from the correct specification of one of the candidate submodels. Furthermore, DCMA delivers almost the same performances as DGMA; both are inferior to the oracle global approach while the gap is narrowed with the use of the sorted uniform prior. Similar conclusions can be drawn from Fig. 3 when N is increased to 8000.

Overall, DGMA generally outperforms DCMA, especially when datasets are distributed across a large number of local machines or a noninformative prior is assigned to candidate submodels; both situations are commonly encountered with massive data, making DGMA a preferable option in practice.

Table 2 The mean squared error (MSE) for the prediction of the response averaged over 500 simulations and the corresponding standard deviation (SD) in the testing dataset with the uniform prior

n	J	K_n		DGMA				DCMA	Oracle
				$t = 1$	$t = 2$	$t = 3$	$t = 4$		
100	40	200	MSE	0.7507	0.6759	0.6585	0.6567	0.6941	0.2556
			SD	0.1308	0.1150	0.1121	0.1124	0.1224	0.0119
		100	MSE	0.3256	0.2715	0.2553	0.2542	0.2596	0.2462
			SD	0.0649	0.0216	0.0132	0.0131	0.0113	0.0114
400	10	200	MSE	0.3329	0.3225	0.3221	0.3221	0.3229	0.2556
			SD	0.1361	0.1288	0.1287	0.1286	0.1293	0.0119
		100	MSE	0.2559	0.2459	0.2453	0.2453	0.2459	0.2462
			SD	0.0175	0.0114	0.0114	0.0114	0.0114	0.0114
100	80	200	MSE	0.7406	0.6671	0.6512	0.6499	0.6862	0.2561
			SD	0.1309	0.1182	0.1153	0.1162	0.1264	0.0111
		100	MSE	0.3248	0.2706	0.2554	0.2535	0.2595	0.2465
			SD	0.0642	0.0223	0.0132	0.0118	0.0106	0.0106
400	20	200	MSE	0.3277	0.3174	0.3170	0.3170	0.3174	0.2561
			SD	0.1299	0.1230	0.1229	0.1230	0.1228	0.0111
		100	MSE	0.2563	0.2462	0.2456	0.2456	0.2462	0.2465
			SD	0.0176	0.0106	0.0105	0.0105	0.0105	0.0106

Table 3 The mean squared error (MSE) for the prediction of the response averaged over 500 simulation runs and the corresponding standard deviation (SD) in the testing dataset with the sorted uniform prior

n	J	K_n		DGMA				DCMA	Oracle
				$t = 1$	$t = 2$	$t = 3$	$t = 4$		
100	40	200	MSE	0.6939	0.6584	0.6533	0.6539	0.6606	0.2553
			SD	0.1214	0.1129	0.1113	0.1116	0.1121	0.0120
		100	MSE	0.2454	0.2474	0.2450	0.2472	0.2468	0.2450
			SD	0.0119	0.0115	0.0114	0.0115	0.0114	0.0115
400	10	200	MSE	0.3253	0.3227	0.3226	0.3226	0.3226	0.2553
			SD	0.1321	0.1296	0.1295	0.1295	0.1296	0.0120
		100	MSE	0.2454	0.2451	0.2450	0.2451	0.2450	0.2450
			SD	0.0115	0.0115	0.0115	0.0115	0.0115	0.0115
100	80	200	MSE	0.7357	0.6657	0.6520	0.6510	0.6595	0.2558
			SD	0.1218	0.1100	0.1082	0.1084	0.1099	0.0111
		100	MSE	0.2458	0.2478	0.2454	0.2477	0.2472	0.2453
			SD	0.0109	0.0104	0.0105	0.0105	0.0104	0.0105
400	20	200	MSE	0.3208	0.3173	0.3172	0.3172	0.3171	0.2558
			SD	0.1279	0.1245	0.1243	0.1243	0.1243	0.0111
		100	MSE	0.2460	0.2455	0.2454	0.2454	0.2454	0.2453
			SD	0.0108	0.0105	0.0105	0.0105	0.0105	0.0105

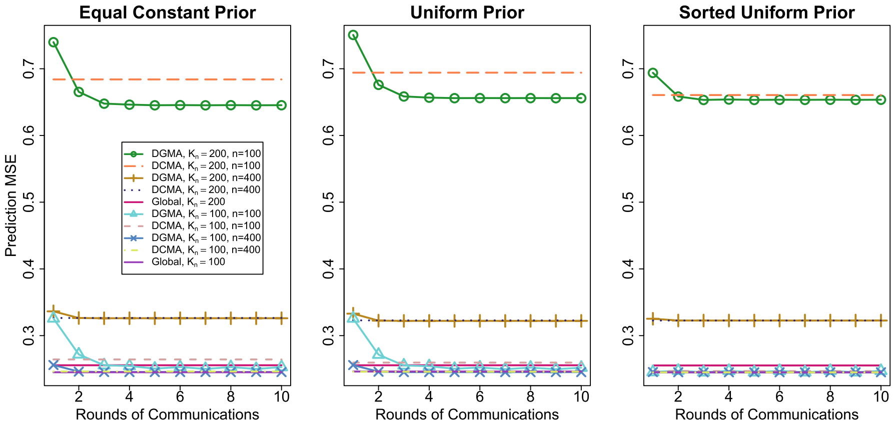

Fig. 2 The mean squared error (MSE) for the prediction of the response in the testing dataset using the DGMA approach within 10 rounds of communications, the DCMA and oracle global approaches for $K_n = 200$ (the true model is not included in the candidate submodels) and for $K_n = 100$ (the true model is included in the candidate submodels) under sample size $N = 4000$

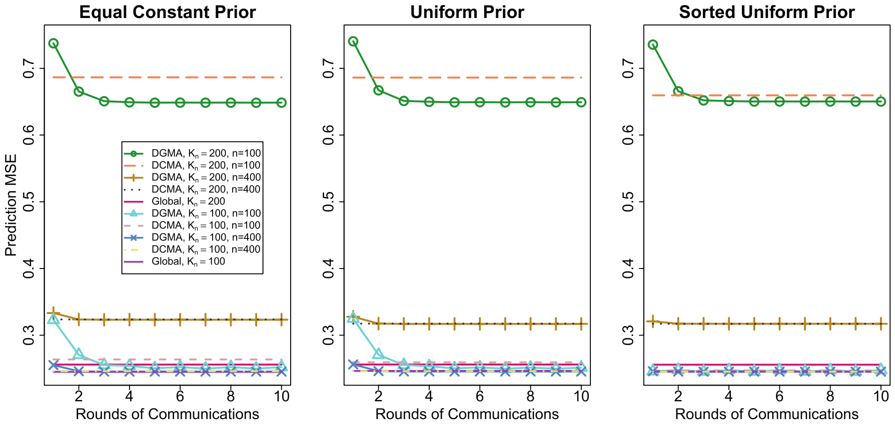

Fig. 3 The mean squared error (MSE) for the prediction of the response in the testing dataset using the DGMA approach within 10 rounds of communications, the DCMA and oracle global approaches for $K_n = 200$ (the true model is not included in the candidate submodels) and for $K_n = 100$ (the true model is included in the candidate submodels) under sample size $N = 8000$

5 Real examples

5.1 Asset prediction

As an illustration, we apply the proposed DGMA, DCMA and the oracle global approaches to analyzing data from the 1991 Survey of Income and Program Participation. In the early 1980s, the United States introduced several tax-deferred savings options and launched 401(k) plans to stimulate individual savings for retirement. In practice, the net financial asset is usually chosen as an evaluation for the individual saving preference. In this study, we are interested in predicting the net financial asset based on several predictors. There are 9915 observations and for each observation 74 predictors are collected, including age, income, family size, education, marriage status, home owner and some other financial indexes, as well as the net financial asset which is considered as the dependent variable. Normalization is adopted for continuous variables. We keep the original 74 predictors as the main effects and construct two-way interactions among all non-dummy predictors, which ends up with a total of 378 predictors after removing terms that are perfectly collinear. Further information on the dataset can be found in Chernozhukov and Hansen (2004).

We randomly select 1600 observations for determining the tuning parameters using the leave-one-machine-out CV procedure, and 7200 observations for training, and the remaining for testing. At the initial step, we apply the SIS procedure to rank all the predictors. Every 10 (or 20) predictors are grouped together to formulate the candidate submodels; accordingly the last candidate submodel has 8 (or 18) predictors and $K_n = 38$ (or 19). We also consider sample size $n = 100$ or 400 on each local machine, leading to the number of local machines being $J = 72$ or 18, respectively. For each configuration, we replicate the experiment 100 times.

Figure 4 exhibits the MSE of prediction for the net financial asset. In general, the oracle global method yields the lowest MSE among all the three approaches. When the number of local machines is $J = 72$, DGMA outperforms DCMA under the equal constant and uniform priors. In contrast, DCMA performs better than DGMA under the sorted uniform prior. Similar to conclusions from the simulation studies, DCMA benefits more from such an informative prior. Nevertheless, a sufficient number of communications among local machines can help DGMA to diminish the gap, which further demonstrates the merit of our distributed approach because an informative prior is typically not available but communications can be readily carried out among machines. When $J = 18$, DGMA and DCMA perform equally well

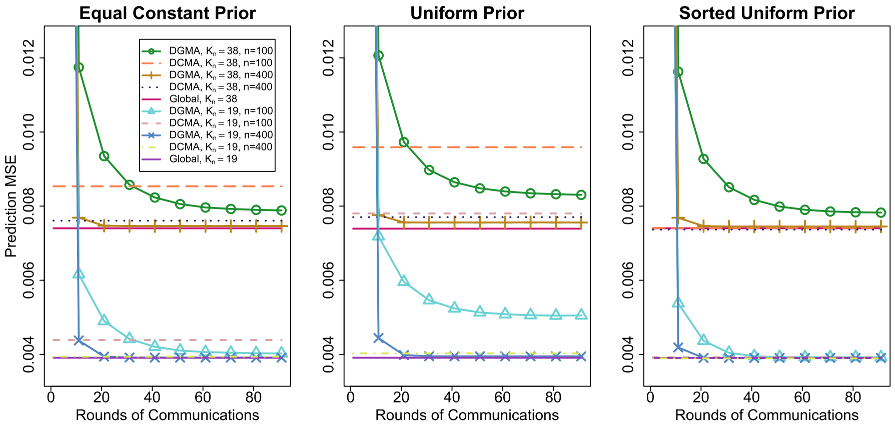

Fig. 4 The mean squared error (MSE) for the prediction of the net financial asset in the testing dataset using the proposed DGMA approach within 100 rounds of communications, the DCMA and oracle global approaches, respectively

and both can even reach the prediction accuracy of the oracle global approach. Thus, the adequate size of the local samples that each machine can access significantly enhances the performances of DGMA and DCMA. In addition, the performances of all the three approaches can also be improved substantially when the number of candidate submodels is reduced from $K_n = 38$ to 19, indicating that the size of submodels is another key factor influencing the performance. We conclude that DGMA generally delivers better performances than DCMA, especially in the typical cases of practice where the local sample size is not adequate, the number of local machines (candidate submodels) is large, or there is no prior information on the weights.

5.2 Age prediction

We also apply the proposed DGMA, DCMA and the oracle global approaches to the human facial image dataset for predicting people's age. The dataset is generated by using the pixel values of facial images of celebrities with ages from 14 to 62 years old. Our goal is to predict the age based on the facial image information. Although there could be multiple but different images for each celebrity, we label them as different samples and are finally end up with a total of 175633 facial images. The histogram of ages is shown in Fig. 5, which indicates a smaller proportion of facial images at the extremely young and elderly ages. The first 512 features are extracted by employing the default "ResNet18" method using the PyTorch from the ImageNet database. Based on the 512 features, we adopt the two hidden layers convolution method (Nair and Hinton 2010) to further generate 256 and 128 features for each layer respectively. We remove 34 features that are perfectly collinear with the others and eventually obtain a total of 862 features. The proportional stratification sampling strategy is utilized to randomly sample from the 175633 facial images; 40000 observations are used to determine the tuning parameter, 120000 observations for training and the remaining for testing. The age labels and all the features are normalized. After the initial SIS screening step to rank the features, we construct $K_n = 87$ (and 44) candidate submodels with sizes of 10 (and 20) (the last submodel is of size 2), respectively. The local sample size n varies from 500 to 8000, indicating the

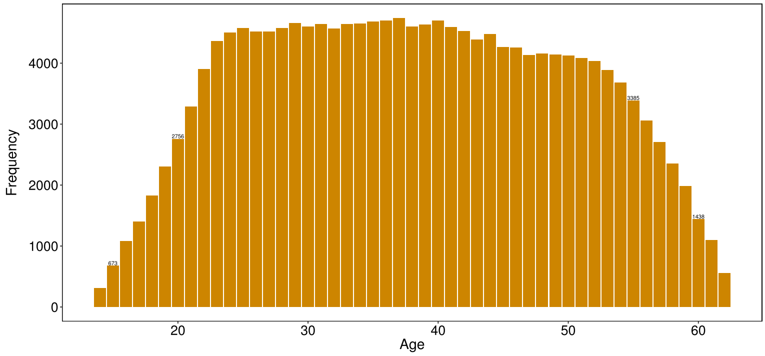

Fig. 5 The histogram of age corresponding to the 175633 facial images

number of local machines J from 240 to 15. Under each configuration, the experiment is repeated for 100 times.

As shown in Fig. 6, the oracle global approach delivers the lowest MSE for age prediction. The MSEs for age prediction using all the three approaches dramatically decrease when K_n is reduced from 87 to 44. It is interesting to see that all the three approaches are robust with respect to the three different types of prior information. The superiority of DGMA over DCMA is remarkable, especially in the scenario of $n = 500$ or $J = 240$, while the performances of DGMA and the oracle approach are comparable when $n = 2000$ or $J = 60$. Figure 7 exhibits the MSEs of age prediction (of DGMA at the 10-th communication round) versus the local sample size n

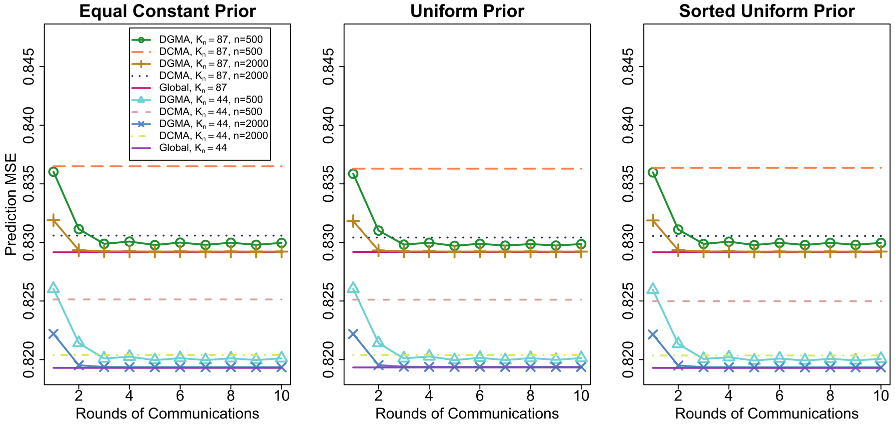

Fig. 6 The mean squared error (MSE) for the prediction of age in the testing dataset using the proposed DGMA approach within 10 rounds of communications, the DCMA and oracle global approaches, respectively

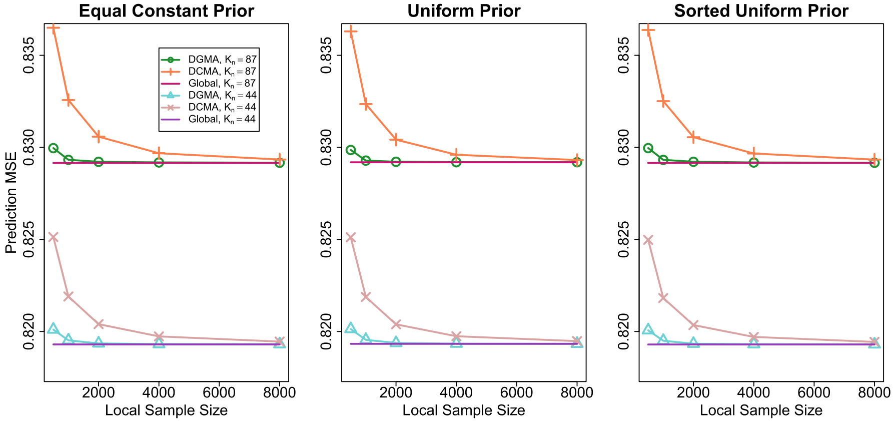

Fig. 7 The mean squared error (MSE) for the prediction of age in the testing dataset using the proposed DGMA approach at the 10th communication round, the DCMA and oracle global approaches against the local sample size, respectively

ranging from 500 to 8000, which further demonstrates that the MSE of age prediction using DGMA decreases at a much faster rate than DCMA and both eventually achieve the benchmark of the oracle. Regardless of whether the massive dataset is stored on a large or a small number of local machines, DGMA is always a promising choice.

For illustration, we select 10 celebrities to demonstrate the prediction of their ages in Fig. 8. The age prediction appears to be over-estimated for extremely young celebrities (e.g., 15 or 20 years), but under-estimated for extremely elderly ones (e.g., 60 years). One possible reason is the data are much more sparse for both extremely young and elderly people. All three methods perform similarly in terms of age prediction under different numbers of candidate submodels.

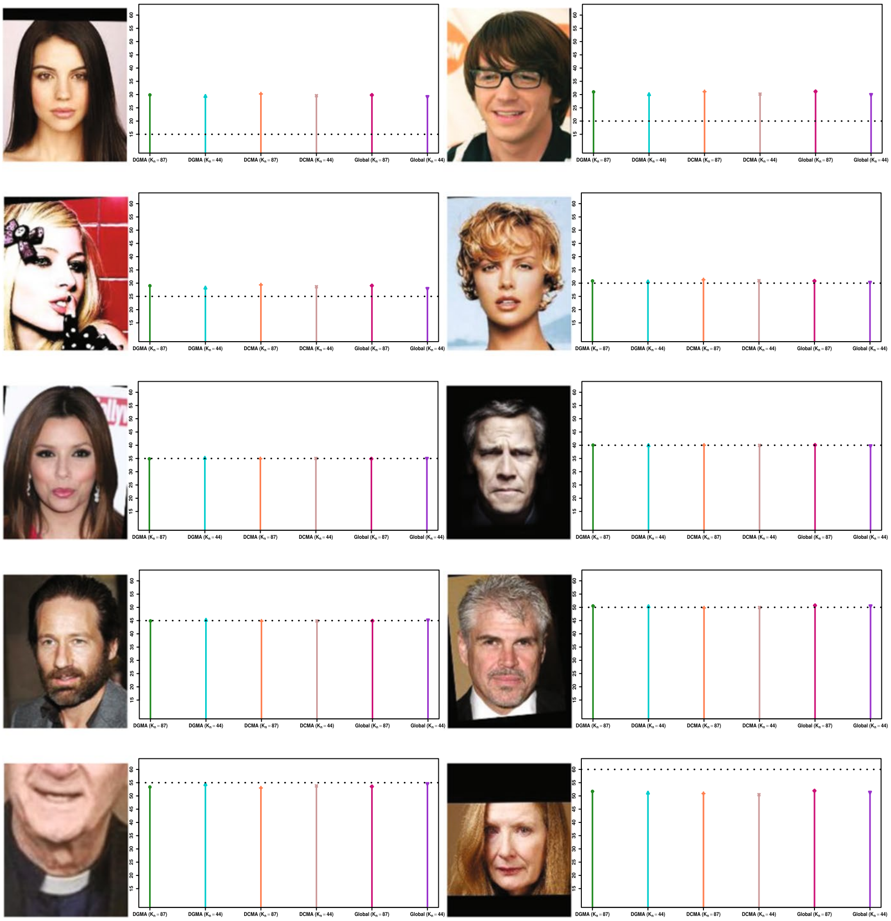

Fig. 8 Estimated age for 10 selected celebrities using the proposed DGMA approach at the 10th communication round, the DCMA and oracle global approaches with the uniform prior and $n = 2000$, respectively

6 Discussion

We have developed a distributed gridding model aggregation approach for predicting the conditional mean of the response given high-dimensional predictors when the datasets are stored across multiple machines. By defining a surrogate loss function for the infeasible oracle global approach, the weights for model aggregation can be obtained by elaborately designing a distributed algorithm. It further demonstrates that our method can strike a balance between statistical accuracy and communication cost. Numerical studies show that the proposed approach generally outperforms the simple divide-and-conquer model averaging approach and is comparable to the oracle global approach.

The regularization-based prediction methods generally require the assumption of correct model specification, which however may be violated, especially when dealing with massive data. The proposed method, aggregating the linear submodel bases as an approximation for the underlying unspecified model, could be considered as a model-free method and thus is more promising in practice. Furthermore, polynomial models can also be adopted as the submodel bases, which may provide a better approximation to the full model space. Yet, the computational burden would increase accordingly. The proposed method can be extended to generalized linear models if the conditional mean of other types of response variables, e.g., binary or ordinal, is of interest. Heterogeneity may be present in the massive dataset and it is thus more sensible to predict the conditional quantile of the response in practice. However, the cusp of the check loss function for quantile regression renders such extension non-trivial.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00180-022-01242-3.

Acknowledgements We would like to thank the two referees and Associate Editor for many insightful comments that greatly improved the paper. Liu's research was partly supported by the National Natural Science Foundation of China (11971362), Yin's research was partly supported by the Research Grants Council of Hong Kong (17308321), and Wu's research was partly supported by the National Natural Science Foundation of China (12071483).

References

Ando T, Li K-C (2014) A model-averaging approach for high-dimensional regression. J Am Stat Assoc 109:254–265

Ando T, Li K-C (2017) A weight-relaxed model averaging approach for high dimensional generalized linear models. Ann Stat 45:2645–2679

Battey H, Fan J, Liu H, Lu J, Zhu Z (2018) Distributed testing and estimation under sparse high dimensional models. Ann Stat 46:1352–1382

Boyd S, Parikh N, Chu E, Peleato B, Eckstein J (2011) Distributed optimization and statistical learning via the alternating direction method of multipliers. Trends Mach Learn 3:1–122

Buckland ST, Burnham KP, Augustin NH (1997) Model selection: an integral part of inference. Biometrics 53:603–618

Burnham KP, Anderson DR (2003) Model selection and multimodel inference: a practical information-theoretic approach. Springer, New York

Chen X, Xie M (2014) A split-and-conquer approach for analysis of extraordinarily large data. Stat Sin 24:1655–1684

Chen X, Zhang Y, Li R, Wu X (2016) On the feasibility of distributed kernel regression for big data. IEEE Trans Knowl Data Eng 28:3041–3052

Chernozhukov V, Hansen C (2004) The impact of 401(k) participation on the wealth distribution: an instrumental quantile regression analysis. Rev Econ Stat 86:735–751

Dai D, Rigollet P, Zhang T (2012) Deviation optimal learning using greedy Q-aggregation. Ann Stat 40:1878–1905

Eklund J, Karlsson S (2007) Forecast combination and model averaging using predictive measures. Econ Rev 26:329–363

Fan J, Lv J (2008) Sure independence screening for ultrahigh dimensional feature space. J R Stat Soc B 70:849–911

Hansen BE (2007) Least squares model averaging. Econometrica 75:1175–1189

Hansen BE, Racine JS (2012) Jackknife model averaging. J Econ 167:38–46

Hjort NL, Claeskens G (2003) Frequentist model average estimators. J Am Stat Assoc 98:879–899

Hoeffding W (1963) Probability inequalities for sums of bounded random variables. J Am Stat Assoc 58:13–30

Hoeting JA, Madigan D, Raftery AE, Volinsky CT (1999) Bayesian model averaging: a turorial. Stat Sci 14:382–401

Jakovetic D, Xavier J, Moura JMF (2014) Fast distributed gradient methods. IEEE Trans Autom Control 59:1131–1146

Johnson R, Zhang T (2013) Accelerating stochastic gradient descent using predictive variance reduction. Adv Neural Inf Process Syst 26:315–323

Jordan MI, Lee JD, Yang Y (2019) Communication-efficient distributed statistical learning. J Am Stat Assoc 114:668–681

Kleiner A, Talwalkar A, Sarkar P, Jordan MI (2014) A scalable bootstrap for massive data. J R Stat Soc B 76:795–816

Lee JD, Liu Q, Sun Y, Taylor JE (2017) Communication-efficient sparse regression. J Mach Learn Res 18:1–30

Liang H, Zou G, Wan ATK, Zhang X (2011) Optimal weight choice for frequentist model average estimators. J Am Stat Assoc 106:1053–1066

Mateos G, Bazerque JA, Giannakis GB (2010) Distributed sparse linear regression. IEEE Trans Signal Process 58:5262–5276

Nair V, Hinton GE (2010) Rectified linear units improve restricted boltzmann machines. In: ICML'10 Proceedings of the 27th international conference on international conference on machine learning, pp 807–814

Newbold P, Granger CWJ (1974) Experience with forecasting univariate time series and the combination of forecast (with discussion). J R Stat Soc Ser A 137:131–165

Raftery AE, Madigan D, Hoeting JA (1997) Bayesian model averaging for linear regression models. J Am Stat Assoc 92:179–191

Rosenblatt J, Nadler B (2016) On the optimality of averaging in distributed learning. Inf Inference 5:379–404

Shamir O, Srebro N, Zhang T (2014) Communication-efficient distributed optimization using an approximate Newton-type method. Proc Int Conf Mach Learn 32:1000–1008

Shang Z, Cheng G (2017) Computational limits of a distributed algorithm for smoothing spline. J Mach Learn Res 18:1–37

Shvachko K, Kuang H, Radia S, Chansler R (2010) The hadoop distributed file system. In: IEEE 26th symposium on mass storage systems and technologies, pp 1–10

van de Geer SA (2008) High-dimensional generalized linear models and the lasso. Ann Stat 36:614–645

Wan ATK, Zhang X, Zou G (2010) Least squares model averaging by Mallows criterion. J Econ 156:277–283

Wang J, Kolar M, Srebro N, Zhang T (2017) Efficient distributed learning with sparsity. Proc Mach Learn Res 70:3636–3645

Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I (2010) Spark: cluster computing with working sets. In: Proceedings of the 2nd USENIX conference on hot topics in cloud computing, pp 10–10

Zhang C, Lee H, Shin K (2012) Efficient distributed linear classification algorithms via the alternating direction method of multipliers. Proc Int Conf Artif Intell Stat 22:1398–1406

Zhang X, Zou G, Liang H (2014) Model averaging and weight choice in linear mixed-effects models. Biometrika 101:205–218

Zhang Y, Duchi J, Jordan JC, Wainwright MJ (2013) Information-theoretic lower bounds for distributed statistical estimation with communication constraints. Adv Neural Inf Process Syst 26:2328–2336

Zhang Y, Duchi J, Wainwright M (2013) Divide and conquer kernel ridge regression. Conf Learn Theory 30:1–26

Zhao T, Cheng G, Liu H (2016) A partially linear framework for massive heterogeneous data. Ann Stat 44:1400–1437

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.