The New England Journal of Statistics in Data Science Volume 0, 1–14 (2020)
DOI: https://doi.org/0000

Subdata selection with a large number of variables

Rakhi SINGH[0] and John STUFKEN[1]

Abstract

Subdata selection from big data is an active area of research that facilitates inferences based on big data with limited computational expense. For linear regression models, the optimal design-inspired Information-Based Optimal Subdata Selection (IBOSS) method is a computationally efficient method for selecting subdata that has excellent statistical properties. But the method can only be used if the subdata size, k, is at last twice the number of regression variables, p. In addition, even when $k \geq 2p$, under the assumption of effect sparsity, one can expect to obtain subdata with better statistical properties by trying to focus on active variables. Inspired by recent efforts to extend the IBOSS method to situations with a large number of variables p, we introduce a method called Combining Lasso And Subdata Selection (CLASS) that, as shown, improves on other proposed methods in terms of variable selection and building a predictive model based on subdata when the full data size n is very large and the number of variables p is large. In terms of computational expense, CLASS is more expensive than recent competitors for moderately large values of n, but the roles reverse under effect sparsity for extremely large values of n.

KEYWORDS AND PHRASES: Effect Sparsity, Optimal Design, Prediction, Subsampling, Variable Selection.

1. INTRODUCTION

Unprecedented advancements in modern information technologies have resulted in an exponential growth of data and massive datasets. Data sizes are now measured in terabytes (TB) or petabytes (PB) and not in mere megabytes (MB) or gigabytes (GB). Big data facilitates and incentivizes data-driven decisions in almost every area of science, industry, and government. Given the challenges that big data presents due to its volume, variety, and complexity, extracting high-quality information from big data is a prerequisite for understanding the data meaningfully [5].

Some statistical methods for analyzing big data include bags of little bootstraps by [21], divide-and-conquer [22, 6, 34, 31, for example] and sequential updating for streaming data [32, 45, for example]. In divide-and-conquer approaches, statistical analyses are performed on multiple parts of the data, and then these results are combined to form overall conclusions. In sequential updating, since the data is made available in streams or large chunks, sequential analysis methods that do not require storing the big data are developed. Interested readers are directed to [38] for a comprehensive review of these approaches. Within the subsampling-based approaches to handle big data, one approach is to work with a carefully selected small representative sample (called *subdata*) of size k from the big data of size n (called *full data*). The sample size k should be chosen so that appropriate statistical tools and methods can be applied to the subdata with sufficiently reduced computational complexity. Methods to identify such subdata are called *subdata selection methods*.

The current literature on subdata selection is rapidly growing. Much of the relevant literature focuses on identifying subdata that yields precise estimates of parameters in a given statistical model, for example, for linear regression [see, 10, 23, 8, 36, 41], logistic regression [42, 39, 7], multinomial logistic regression [46], generalized linear models [13, 36, 1, 50, 53, 16], quantile regression [40, 2, 12, 33], and quasi-likelihood [50]. All of these methods assume a true underlying model. Methods that allow for the misspecification of a linear model [29], a non-parametric regression model [30], a distributed computing environment [52], and model selection [49] also exist. Typically, model-based methods aid in estimating model parameters and, as a byproduct, in the prediction for new test data.

In addition, model-free subdata selection methods also exist. For example, one could mirror the population distribution in the subdata [26, 19, 37], or compress the full data in a small set for prediction [18]. Selective reviews of subdata selection methods are also provided by [48] and [47].

While subdata selection methods focus on data reduction by drastically reducing the number of observations n, they tend to become computationally intensive or statistically inefficient when the number of variables p is moderate to large. In this paper, we consider the situation when $n \gg p$, but p is moderate to large (in the thousands). We assume that the response can be modeled using a linear model and use the Information-Based Optimal Subdata Selecton (IBOSS)

*Corresponding author.
[1]The authors gratefully acknowledge support through NSF grants DMS-1935729 and DMS-2304767.

method [41] in conjunction with LASSO [35] to combine variable selection and subdata selection.

In Section 2, we provide a brief background for the current analysis methods and subdata selection methods. Section 3 describes our method, explores its analytical properties, and discusses its advantages. Section 4 compares the proposed method with competing methods on simulated and real data. Finally, we provide some concluding remarks in Section 5.

2. BACKGROUND

2.1 The model and analysis methods

2.1.1 The model

Let $(\mathbf{X}, \mathbf{Y})$ be the full data, where $\mathbf{X}$ is a $n \times p$ matrix with n observations and p (independent) variables or features and $\mathbf{Y}$ is the corresponding $n \times 1$ response vector. The linear regression model is

$$y_i = \beta_0 + \mathbf{x}_i^T \boldsymbol{\beta}_1 + \epsilon_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \epsilon_i, i = 1, \ldots, n, \quad (2.1)$$

where β_0 is the intercept parameter, $\boldsymbol{\beta}_1 = (\beta_1, \ldots, \beta_p)^T$ is the vector of slope parameters, and $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})^T$, y_i, and ϵ_i are the vector of variable values, response, and error for the ith observation, respectively. Further, we write $\mathbf{z}_i = (1, \mathbf{x}_i^T)^T$, $\boldsymbol{\beta} = (\beta_0, \boldsymbol{\beta}_1^T)^T$, $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^T$, $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_n)^T$, and $\mathbf{Y} = (y_1, \ldots, y_n)^T$. We assume that the ϵ_i's are independent and identically distributed with mean 0 and variance σ^2.

2.1.2 The ordinary least squares (OLS) estimator

When using the full data and model (2.1), the least-squares estimator of $\boldsymbol{\beta}$, which is also its best linear unbiased estimator, is $\hat{\boldsymbol{\beta}}_f = (\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{Z}^T\mathbf{Y}$. The covariance matrix of this unbiased estimator is equal to the inverse of the Fisher information matrix $\mathbf{I}_f$ for $\boldsymbol{\beta}$ from the full data

$$\mathbf{I}_f = \frac{1}{\sigma^2}\mathbf{Z}^T\mathbf{Z}. \quad (2.2)$$

Subdata of size k can be represented by a $k \times (p+1)$ matrix $\mathbf{Z}_s$, which consists of k rows of the full data matrix $\mathbf{Z}$, and the corresponding vector of responses $\mathbf{Y}_s$. The OLS estimator based on the subdata is $\hat{\boldsymbol{\beta}}_s = (\mathbf{Z}_s^T\mathbf{Z}_s)^{-1}\mathbf{Z}_s^T\mathbf{Y}_s$.

2.1.3 The LASSO estimator

For high-dimensional data (i.e., large p), it is common to assume that only a few variables affect the response (*sparsity*). We will refer to these variables as "active" variables. Penalized regression methods are used to analyze data in such situations. One well-known method is LASSO with an ℓ_1-norm regularization. The LASSO estimator $\hat{\boldsymbol{\beta}}_{LASSO}$ is a solution to the following optimization problem [35]

$$\text{argmin}_{\boldsymbol{\beta} \in \mathcal{R}^{p+1}} \frac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{z}_i^T\boldsymbol{\beta})^2 + \lambda||\boldsymbol{\beta}||_1, \quad (2.3)$$

where $||\boldsymbol{\beta}||_1 = |\beta_0| + |\beta_1| + \cdots + |\beta_p|$ is the ℓ_1-norm of $\boldsymbol{\beta}$, and λ is the regularization parameter. If the tuning parameter λ goes to 0 slower than $1/\sqrt{n}$, then, provided that $\mathbf{I}_f$ is non-singular, $\sqrt{n}(\hat{\boldsymbol{\beta}}_{LASSO} - \boldsymbol{\beta})$ converges in distribution [15, 44] to $N(0, \sigma^2\mathbf{I}_f^{-1})$, where $\mathbf{I}_f$ is as in (2.2). In practice, cross-validation is typically used to tune λ. Two solutions, one with the minimum cross-validation error (corresponding to `lambda.min` in R package `glmnet` [14]) and another with the most regularized solution within 1 standard deviation of the minimum cross-validation error (corresponding to `lambda.1se` in R package `glmnet`), are widely used.

2.2 Subdata selection methods for OLS

For a linear regression model, current subdata selection methods can be broadly classified into two categories:

- Probabilistic methods. The k subdata observations are randomly sampled with replacement from the full data, each time using a selection probability π_i for the ith observation in the full data, $i = 1, \ldots, n$, $\sum_{i=1}^{n}\pi_i = 1$. This is often combined with the weighted least squares estimator [cf. 41]

$$\left(\sum_{i=1}^{n} w_i\eta_i\mathbf{z}_i\mathbf{z}_i^T\right)^{-1}\sum_{i=1}^{n} w_i\eta_i\mathbf{z}_i y_i, \quad (2.4)$$

  where η_i denotes the number of times that the ith data point is included in the subsample, and the weight w_i is often taken to be proportional to $1/\pi_i$. Using $\pi_i = 1/n$ and $w_i = 1$ for $i = 1, \ldots, n$ constitutes simple random sampling with replacement, to which we will refer as uniform sampling (UNI), whereas leverage sampling [10, 25, 23, 24, 9] uses $\pi_i = h_{ii}/(p+1)$ and $w_i = 1/\pi_i$ where h_{ii} is the leverage value of the ith observation obtained as $\mathbf{z}_i^T(\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{z}_i$. Another probabilistic approach uses influence functions to find the subsampling probabilities [36]. One major limitation of probabilistic methods [see, 41] is that variances of the estimators in (2.4), given some mild conditions on $\mathbf{X}$, are bounded below by quantities of order $1/k$, and do not approach 0 as $n \to \infty$. Except for UNI, other methods can still be computationally expensive for values of n and p considered in this paper.
- Deterministic methods: These methods, some of which draw inspiration from the optimal design literature, aim to select subdata of size k that optimizes an objective function. Under model (2.1), the information matrix for $\boldsymbol{\beta}$ when using subdata of size k is

$$\mathcal{I}(\boldsymbol{\delta}) = \frac{1}{\sigma^2}\mathbf{Z}^T\boldsymbol{\Delta}\mathbf{Z} = \frac{1}{\sigma^2}\mathbf{Z}_s^T\mathbf{Z}_s, \quad (2.5)$$

where $\boldsymbol{\Delta} = \text{diag}(\boldsymbol{\delta})$, $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_n)^T$, δ_i is the indicator variable indicating whether the ith data point is in the subdata or not, and $\sum_{i=1}^{n} \delta_i = k$. Based on the structure of D-optimal designs [20], [41] devised a deterministic subdata selection method for finding subdata that, approximately, maximizes the determinant of $\mathcal{I}(\boldsymbol{\delta})$ for a give value of k. The method is called D-optimal Information-Based Optimal Subdata Selection (IBOSS). The D-optimality criterion minimizes the expected volume of the joint confidence ellipsoid for $\boldsymbol{\beta}$. When $r = k/(2p)$ is an integer, the simple algorithm proposed in [41], which has computational complexity $O(np)$, selects IBOSS subdata by sequentially considering each column of $\mathbf{X}$ and selecting the observations with the r smallest and r largest values for each of the p variables. Orthogonal subsampling, proposed by [43], is another deterministic approach inspired by experimental design. In addition to having extreme observations for each variable, a better approximate solution to maximizing the determinant can be obtained if the subdata has a structure that mirrors the structure of an orthogonal array of strength 2 [17]. It can provide subdata that has a better spread in p-dimensional space. [41] showed nice theoretical properties of the estimator from the IBOSS sample (discussed in Section 3.3).

In what follows, we use IBOSS as a subdata selection method owing to its computational and statistical superiority.

2.3 Challenges with using IBOSS for large p

Since IBOSS attempts to select at least 2 data points for each variable, it can only be applied if $k \geq 2p$. But even when this condition is satisfied, the subdata that is obtained by applying IBOSS may not be great for large p given that many variables are likely to be inactive. For $k < 2p$, [44] recently proposed first selecting p^+ variables with the largest absolute correlation with the response by the sure independence screening (SIS) method [11] and then only using these p^+ variables for selecting the IBOSS subdata. They call this procedure SIS-IBOSS. They then analyzed the subdata, using all p variables, by using LASSO. They also used this analysis for $k \geq 2p$, in which case they selected the subdata simply by using IBOSS. If the tuning parameter λ of LASSO goes to 0 slower than $1/\sqrt{n}$, then [44] showed that the asymptotic behavior of the $\hat{\boldsymbol{\beta}}$ in (2.3) is the same as that of an OLS estimator. Therefore, both IBOSS and SIS-IBOSS work well with LASSO. Note, however, that SIS only considers the marginal relation of each variable with the response and works best when the variables are independent [11]. SIS-IBOSS also suffers from this problem (see Section 4) and does not work well when the variables are correlated. Another challenge with SIS-IBOSS is that a good value of p^+ is generally unknown and is hard to guess.

We therefore develop a subdata selection method that mitigates these challenges for high-dimensional data.

3. METHODOLOGY

3.1 Overview of our method

With the ultimate goal of prediction, our method first screens variables to identify the active variables, and then performs the subdata selection using only the identified variables. Finally, a linear regression model with only the variables identified as active is fitted using the subdata and OLS estimation. Algorithm 1 provides more details for our method, which we call *CLASS* for Combining Lasso and Subdata Selection. Steps 1 to 8 of Algorithm 1 focus on variable selection. The novel variable selection method runs LASSO multiple times on small randomly selected subsets of the full data. Variables that are consistently selected in different LASSO runs are declared active. Unlike [28] and [3], we use a kmeans-based data-driven approach for deciding which variables are consistently selected. Step 9 of Algorithm 1 focuses on the subdata selection using IBOSS only on the selected variables. Finally, in step 10 of Algorithm 1, we fit a linear regression model to the subdata and selected variables using OLS estimation. This model can then be used to obtain predictions on test data.

Algorithm 1: CLASS

inputs : the number of times LASSO is run *ntimes*, the sample size for each LASSO run *nsample*, the matrix $\mathbf{Z}$, and the response vector $\mathbf{Y}$

1 **for** $j = 1 \rightarrow ntimes$ **do**
2 Let $\mathbf{Z}_{nsample,p+1}$ be a $nsample \times (p+1)$ matrix based on a uniform random sample of size *nsample* from the rows of $\mathbf{Z}$, and $\mathbf{Y}_{nsample}$ be the corresponding response vector;
3 Run LASSO on $\mathbf{Y}_{nsample}$ and $\mathbf{Z}_{nsample,p+1}$, with tuning parameter corresponding to `lambda.min` in 10-fold cross-validation;
4 Save the selected active variables to the list B_j;
5 **end**
6 Let $C = (C_1, ..., C_p)$, where C_ℓ is the number of lists $B_1, B_2, \ldots, B_{ntimes}$ that contain the ℓ-th variable;
7 For the counts in C, form two clusters using kmeans;
8 Variables with counts in the cluster with the largest mean are selected as the *active variables*;
9 Use IBOSS to select subdata of size k only using the *active variables* from step 8;
10 Fit a linear regression model, with intercept and the active variables from step 8, using the subdata from step 9 and OLS estimation;

output: Fitted model from step 10

In the next two subsections, we provide evidence for the superior performance of CLASS.

3.2 Variable selection with large n and large p

We perform repeated applications of LASSO in step 3 of Algorithm 1, each time on a randomly selected subset of

size $nsample \times p$. Therefore, for CLASS to perform well, we need to identify conditions under which LASSO performs well. Regular consistency and the model selection consistency of $\hat{\boldsymbol{\beta}}_{LASSO}$ are well-studied in the literature. Writing the tuning parameter as λ_n, if λ_n tends to zero slower than $n^{-1/2}$, then $\hat{\boldsymbol{\beta}}_{LASSO}$ converges to $\boldsymbol{\beta}$, that is, the solution is consistent [15]. With the same condition on the tuning parameter, $\hat{\boldsymbol{\beta}}_{LASSO}$ is strong sign consistent as $n \to \infty$ if and only if the following condition is satisfied [54, 51]:

$$||\mathbf{Z}_{J^C}^T \mathbf{Z}_J (\mathbf{Z}_J^T \mathbf{Z}_J)^{-1} sign(\boldsymbol{\beta}_J)||_\infty \leq 1, \tag{3.1}$$

where J is the index set for active variables, J^C is the complement of J, and $\boldsymbol{\beta}_J$ and $\mathbf{Z}_J$ are the subvector of $\boldsymbol{\beta}$ and submatrix of $\mathbf{Z}$ with elements and columns, respectively, corresponding to J. Since the condition in (3.1) is not trivial to verify, it is difficult to guarantee strong sign consistentcy of LASSO for any $\mathbf{Z}$. For the special case that the tuning parameter $\lambda_n = \lambda_0 n^{-1/2}$, [3] showed that under regularity conditions, (a) the sign of $\hat{\boldsymbol{\beta}}_{LASSO}$ matches with that of the true $\boldsymbol{\beta}$ for indices in J with probability tending to 1 exponentially fast, and (b) all variables with indices in J^C are selected with a probability strictly between $(0, 1)$. Therefore, [3] proposed *Bolasso* where LASSO is applied multiple times on a different bootstrapped sample of the original data. The variables that appear consistently in either all or at least in 90% of LASSO models are declared to be active variables. The latter version is called the soft-threshold version of the *Bolasso*. Except for the fact that we use samples of much smaller size than that of the original data and that we do not use a fixed cutoff for deciding which variables are active in a LASSO run, the *Bolasso* idea mirrors our approach and our method shares the theoretical properties outlined in [3].

The variable selection component of Algorithm 1 is also closely related to the stability selection method developed in [28]. They proposed using $nsample = n/2$ and obtained good results by declaring variables active if counts exceed a threshold of 20%-80% of $ntimes$. Their simulations suggest that $ntimes = 100$ is a reasonable choice. With Steps 7-8 of Algorithm 1, by using kmeans with two clusters, we provide a data-driven way to soft threshold the value above which a variable is declared active. In Section 4, we will demonstrate that the choices $nsample = 1000$ and $ntimes = 100$ for Algorithm 1 and using kmeans with two clusters gives good results. Other choices for $nsample$ and $ntimes$ are explored in the Supplementary Material. Simulations in Section 4 confirm that CLASS tends to select all active variables and a much smaller number of inactive variables than other competing methods.

3.3 Subdata selection on selected variables

In the previous section, we demonstrated that the selected variables contain (a) true active variables with a very large probability and (b) some inactive variables with a positive probability. In Steps 9–10 of Algorithm 1, we first use IBOSS to select subdata of size k only using the variables selected in step 8. We then fit a linear regression model with the intercept and these variables using the subdata and OLS estimation. This section discusses the asymptotic properties of the OLS estimator of $\boldsymbol{\beta}$ obtained as a result of Algorithm 1. If we select all active variables in Step 8 of Algorithm 1, then the OLS estimators for the slopes of the active variables obtained in Step 10 of Algorithm 1 are unbiased.

IBOSS subdata attempts to maximize the determinant of the information matrix of the model based on the selected variables. Since we apply IBOSS only using the selected variables, IBOSS subdata for CLASS gives a larger determinant of the information matrix corresponding to these selected variables than a method that uses IBOSS on all variables. If the selected variables are precisely the active variables, again indexed by J, then D-optimal IBOSS subdata obtained by only using those variables, minimizes the determinant of the variance-covariance matrix of the estimator $\hat{\boldsymbol{\beta}}_J^D$ of the parameter vector in the linear regression model that only uses the active variables. Provided that the column means for the full data are available, the estimate of the intercept parameter is adjusted in [41] to maintain the predictive power of the model, that is,

$$\hat{\beta}_{J0}^D = \bar{\mathbf{Y}} - \bar{\mathbf{X}}_J^T \hat{\boldsymbol{\beta}}_{1J}^D,$$

where $\bar{\mathbf{Y}}$ and $\bar{\mathbf{X}}_J$ are means based on the full data and $\hat{\boldsymbol{\beta}}_{1J}^D$ are the slope parameter estimates from $\hat{\boldsymbol{\beta}}_J^D$. We will use this adjustment for estimating the intercept parameter.

Theorem 5 in [41] provides a general result for the variance of individual parameter estimators for IBOSS subdata. This result also applies to the individual parameter estimators of the selected variables obtained using IBOSS subdata in Algorithm 1 if all active variables are among those that were selected. As argued in Section 3.2, and as will be demonstrated through simulation, the chance of this happening is high. Theorem 6 in [41] discusses the asymptotic results of these variances when the joint variable distribution is either multivariate normal or lognormal. This result also applies to the subdata of CLASS provided that all active variables are among those that were selected.

For a multivariate normal or lognormal distribution of the variables, this would then imply that for the overall mean, $Var(\hat{\beta}_{s0}^D | X)$, is proportional to $1/k$ and never converges to 0 with a fixed k. However, the variance of the estimators of the slope parameters for the selected variables would converge to 0 as the full data size $n \to \infty$. As shown in [41], this nice property of IBOSS subdata is in contrast with other subsampling-based estimators, such as leverage sampling, where the variances of the slope parameter estimators do not converge to 0 because, under mild assumptions, they are bounded from below by terms that are proportional to $1/k$.

The discussion up to this point has focused on variable selection and parameter estimation rather than prediction

even though our goal is good prediction. However, the strong variable selection properties of our method (see Section 3.2) combined with selecting subdata that optimizes the estimation of the coefficients of the selected variables is bound to result in good prediction properties provided that the model assumptions hold.

3.4 Desirable features of CLASS

First, howsoever large the full data is, variable selection can be done in a feasible time since CLASS runs LASSO on small subsets of the full data. Second, CLASS does better at selecting active variables correctly and in not identifying inactive variables as active than LASSO on full data or than other competing subdata selection methods. The superior performance remains true irrespective of whether the variables are correlated. These claims are validated via simulations in the next section. Third, since CLASS employs IBOSS only on the selected variables for obtaining the subdata and since the active variables are almost always among the selected variables, subdata corresponding to CLASS gives a larger determinant for the information matrix for the active variables than the subdata obtained from competing methods. As a result, CLASS leads to better parameter estimation for the selected variables and prediction.

The computational complexity of LASSO for data of size $n \times p$ is $O(p^3 + p^2 n)$. Therefore, the computational complexity of CLASS for steps 1–8 of Algorithm 1 is $O(ntimes(p^3 + p^2 nsample))$. Step 9's computational complexity is $O(np^*)$, where p^* is the number of selected variables in step 8. Finally, for step 10, it is $O(kp^{*2})$. Assuming that $p^* \ll p$ and $k < nsample \times ntimes$, the overall computational complexity of CLASS is $O(ntimes(p^3 + p^2 nsample) + np^*)$. With $ntimes = 100$ and $nsample < n/100$, CLASS is much faster than LASSO on the full data. CLASS is computationally more expensive than [44] for moderately large n, but with appropriate values of $nsample$ and $ntimes$, it becomes less expensive for very large n and large p.

4. NUMERICAL EXPERIMENTS

In this section, we compare the performance of CLASS with competing methods through simulation studies. The comparison focuses on variable selection and prediction accuracy for test data.

Data are generated from the linear model (2.1) with the value of $p = 500$ and $p = 5000$. Let p_1 be the number of true active variables; without loss of generality, we take the first p_1 variables to be active. The coefficients of the active variables and independent error terms are generated from $N(5, 1)$ and $N(0, 1)$, respectively, for $p = 500$ and $p = 5000$. Similar to [41], we also generate error terms from $N(0, 9)$ with coefficients of active variables equal to 1 for $p = 500$. Let $\mathbf{\Sigma} = \mathbf{\Sigma}_{ij}$ be a $p \times p$ correlation matrix for the p variables. For $p = 500$, we considered uncorrelated variables ($\mathbf{\Sigma}_{ij} = 0$), a constant correlation of 0.5 ($\mathbf{\Sigma}ij = 0.5^{I(i \neq j)}$), and a random correlation matrix generated with the R package `randcorr`. For $p = 5000$, we did not consider a random correlation matrix as, for $p = 500$, the comparisons were not very different from the other two correlation structures. Variables $\mathbf{x}_i$'s were generated according to the following scenarios:

Table 1. Simulation scenarios, with **highlighted** *scenarios presented in the main paper. Results for the remaining cases are relegated to the Supplementary Material*

p	p_1	$\boldsymbol{\beta}_{act}$ and σ^2	$\mathbf{x}_i \sim$	$\mathbf{\Sigma}_{ij}$
500	10,25, **50**	$\boldsymbol{\beta}_{act} \sim N(5,1)$, $\sigma^2 = 1$	**Normal**, **LogNormal**, $\mathbf{t_2}$, **Mixture**	$\mathbf{0, 0.5}$, r_{ij}
500	10,25, 50	$\boldsymbol{\beta}_{act} = 1$, $\sigma^2 = 9$	Normal, LogNormal t_2, Mixture	$0, 0.5$, r_{ij}
5000	**25**,50, 75	$\boldsymbol{\beta}_{act} \sim N(5,1)$, $\sigma^2 = 1$	Normal, LogNormal $\mathbf{t_2}$, Mixture	$0, \mathbf{0.5}$

- **Normal:** $\mathbf{x}_i$'s have the multivariate normal distribution $N(0, \mathbf{\Sigma})$
- **LogNormal:** $\mathbf{x}_i$'s have the multivariate lognormal distribution $LN(0, \mathbf{\Sigma})$
- $\mathbf{t_2}$**:** $\mathbf{x}_i$'s have the multivariate t distribution $t_2(0, \mathbf{\Sigma})$
- **Mixture:** $\mathbf{x}_i$'s have the mixture distribution of four different distributions, $N(0, \mathbf{\Sigma})$, $LN(0, \mathbf{\Sigma})$, $t_2(0, \mathbf{\Sigma})$, and $t_3(0, \mathbf{\Sigma})$, with equal proportions.

The simulation scenarios that we considered are summarized in Table 1. We relegate most results to the Supplementary Material and only present the three highlighted cases in Table 1 in the main paper. Methods are evaluated on two characteristics:

- Variable selection: For variable selection we considered average power and average error. Power is defined as the proportion of active variables being correctly identified as active, whereas error is defined as the proportion of inactive variables that are incorrectly declared as active. High power and low error are desired. Therefore, a method with higher power and lower error is preferred.
- Prediction accuracy: We used the mean squared error (MSE) for test data,

$$MSE = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \left(\mathbf{z}_{i,\, test}^T \boldsymbol{\beta} - \mathbf{z}_{i,\, test}^T \hat{\boldsymbol{\beta}} \right)^2, \qquad (4.1)$$

  where n_{test} is the number of observations in the test data, $\mathbf{z}_{i,\, test}$ corresponds to the ith data point in the test data, and $\hat{\boldsymbol{\beta}}$ consists of OLS estimates from a reduced model with zeros added for parameters of variables that were not selected for the reduced model. We set $n_{test} = 1000$ and use the same joint variable distribution for the full data and test data.

We compare CLASS to four other approaches:

1. Fitting the linear regression model with all p variables using LASSO and the full data (FULL)
2. Fitting the linear regression model with all p variables using LASSO and subdata from a uniform sample of size k (UNI)
3. Fitting the linear regression model with all p variables using LASSO and D-optimal IBOSS subdata obtained by using all p variables (IBOSS) [44]
4. Fitting the linear regression model with all p variables using LASSO and D-optimal IBOSS subdata obtained by using p^+ variables selected by applying SIS (SIS(p^+)-IBOSS) [44]

For all four of the above methods, we obtain improved MSE values by using OLS estimators in a linear regression model that is only based on the variables selected by LASSO. We compare the performance of CLASS with these improved MSEs. Adjusting estimators for a model selected by LASSO is not without precedence; for example, Relaxed LASSO [27] adjusts the LASSO estimators by using OLS estimators with a shrinkage parameter. The `glmnet` function in R is used to apply LASSO. Ten-fold cross-validation is used to find the value of the tuning parameter, and the LASSO solution corresponding to `lambda.1se` is selected. For CLASS, we base variable selection and MSE computation on the fitted model from step 10 of Algorithm 1. [44] showed that their IBOSS and SIS(p^+)-IBOSS methods are better than the leverage-based sampling methods, which are therefore omitted from our comparisons.

For each scenario in Table 1, we repeat generating full data and test data 100 times. For each method considered, the average power, error, and MSE over the 100 replications are reported in the subsequent figures.

Figures 1 (variable selection) and 2 (prediction) compare the five methods when $p = 500$, $p_1 = 50$, and $\mathbf{\Sigma} = I_p$, the identity matrix of order p. The regression coefficients for the active variables were generated from the $N(5, 1)$ distribution and the error variance was $\sigma^2 = 1$. The panels of the figures correspond to the four different joint variable distributions: Normal, logNormal, t_2, and Mixture. The full data sizes n are $10^4, 2 * 10^4, 4 * 10^4, 8 * 10^4, 10^5$, and the subdata size is fixed at $k = 1000$. For SIS(p^+)-IBOSS, we use $p^+ = 100$ and for Algorithm 1, we use $nsample = 1000$ and $ntimes = 100$. All methods have a similar power (close to 100%) for all distributions, but, especially for heavier-tailed distributions such as t_2 and Mixture, CLASS has a smaller error. This is not surprising because all other methods select variables based on a single LASSO run, thereby consistently declaring a large number of inactive variables as active. CLASS also has the smallest MSE among all the subdata selection methods, coming close to that for using the full data for heavier-tailed distributions.

For Figures 3 and 4, the only change is that now $\mathbf{\Sigma} = (0.5^{I(i\neq j)})$. The MSE performance in Figure 4 is similar to that in Figure 2, except that CLASS outperforms prediction using LASSO-OLS on the full data for heavy-tailed distributions. For the variable selection performance, CLASS stands out as the big winner when variables are correlated. In Figure 3, all methods except CLASS select too many variables (high error) as a result of using LASSO for variable selection.

In Figure 5, we keep n fixed at 10^5 and change the sample size k from 1000 to 5000. We set $p = 500$, $p_1 = 50$, the joint variable distribution is t_2, and $\mathbf{\Sigma} = (0.5^{I(i\neq j)})$. As expected, all methods perform better on variable selection and MSE as the sample size k increases. CLASS continues to perform better than LASSO-OLS on the full data because the latter declares too many inactive variables as active.

Finally, Figure 6 demonstrates the results when $p = 5000$, $p_1 = 50$ and the joint variable distribution is t_2. We keep k fixed at 1000, use $\mathbf{\Sigma} = (0.5^{I(i\neq j)})$, and change the full data size n from 10^4 to 10^5. Since doing IBOSS on $p = 5000$ variables is not possible, we use $p^+ = 100$ and $p^+ = 250$ for SIS(p^+)-IBOSS. Similar to Figures 3 and 4, CLASS clearly outperforms other methods on both variable selection and prediction accuracy.

For $p_1 = 50$, $\mathbf{\Sigma} = (0.5^{I(i\neq j)})$, joint variable distribution t_2 and $k = 1000$, Tables 2 and 3 illustrate computing times for different values of n and p averaged over 50 iterations on a Desktop with an AMD Ryzen Threadripper PRO 5955WX @4.00 GHz and 64GB RAM. UNI, IBOSS, and SIS-IBOSS are fastest, with CLASS being slower due to the repeated application of LASSO. However, for larger n, CLASS actually becomes faster than IBOSS and SIS-IBOSS. For example, Table 4 shows that CLASS is faster for $n = 10^7$ with $p = 100, p_1 = 50, \mathbf{\Sigma} = I_p$, joint variable distribution t_2 and $k = 1000$. CLASS is also much faster than LASSO-OLS on the full data. Despite the slightly higher run-time for datasets with smaller n, as seen in Figures 1-6, CLASS is a clear winner in terms of statistical performance. In addition, note that, in contrast to LASSO-OLS on the full data, CLASS can be applied in situations when n is ultra-large.

As suggested by one of the reviewers, since CLASS is computationally slower than the other subdata selection methods in Tables 2 and 3, one should make statistical performance comparisons after allowing the same amount of CPU time for each method. Doing this would allow the other methods to use a larger subdata size than CLASS and, presumably, show a better statistical performance than on subdata of the same size as that used for CLASS. Based on this suggestion, we added Tables 5 and 6, which make a comparison between the different subdata selection methods while keeping CPU time approximately constant. Tables 5 and 6, which report averages over 100 iterations for each method and combinations for n and k, list the performances when $p = 500$, $p_1 = 50$, $\mathbf{\Sigma} = (0.5^{I(i\neq j)})$, and the joint variable distribution is t_2 and Normal, respectively. We consider n to be either 10^5 or 10^6 and use $nsample = 1000$ and $ntimes = 100$ for CLASS. In Table 5, we see that CLASS performs better

Figure 1: Variable selection performance for $k = 1000$, $p = 500$, $p_1 = 50$, and $\mathbf{\Sigma} = I_p$. Average power and error are shown by solid lines and dashed lines, respectively.

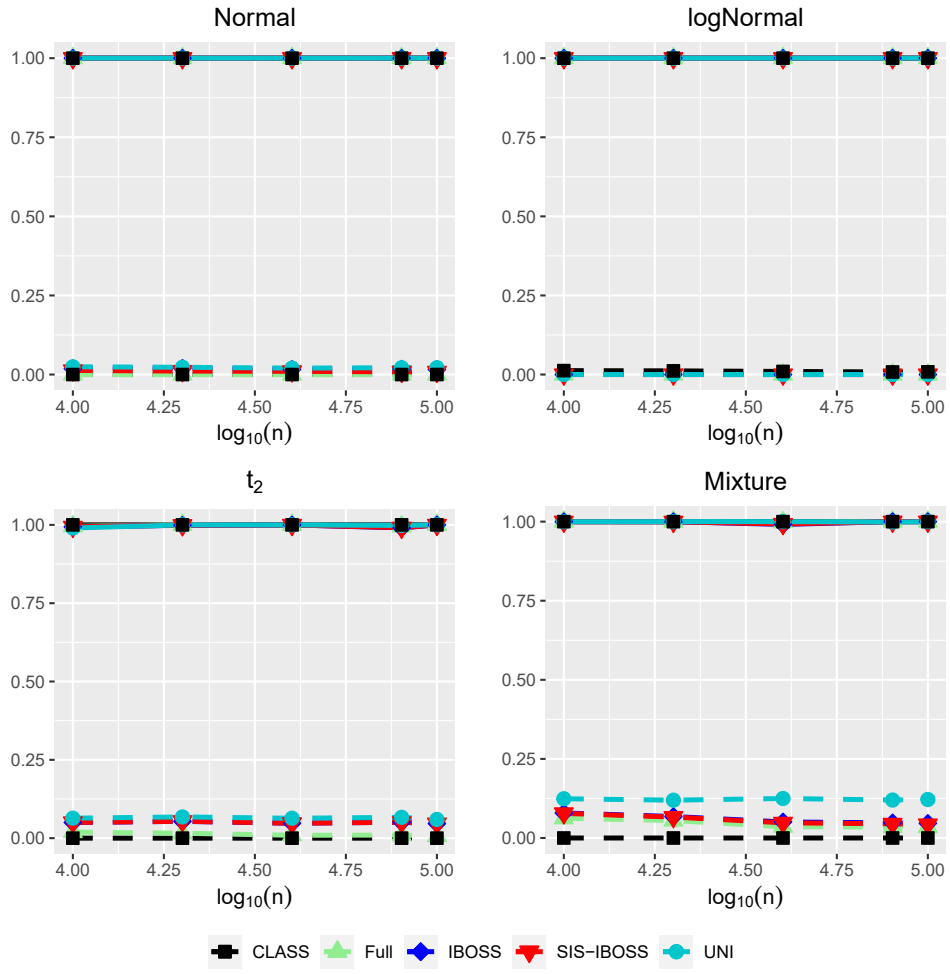

Figure 2: MSE for $k = 1000$, $p = 500$, $p_1 = 50$, and $\mathbf{\Sigma} = I_p$

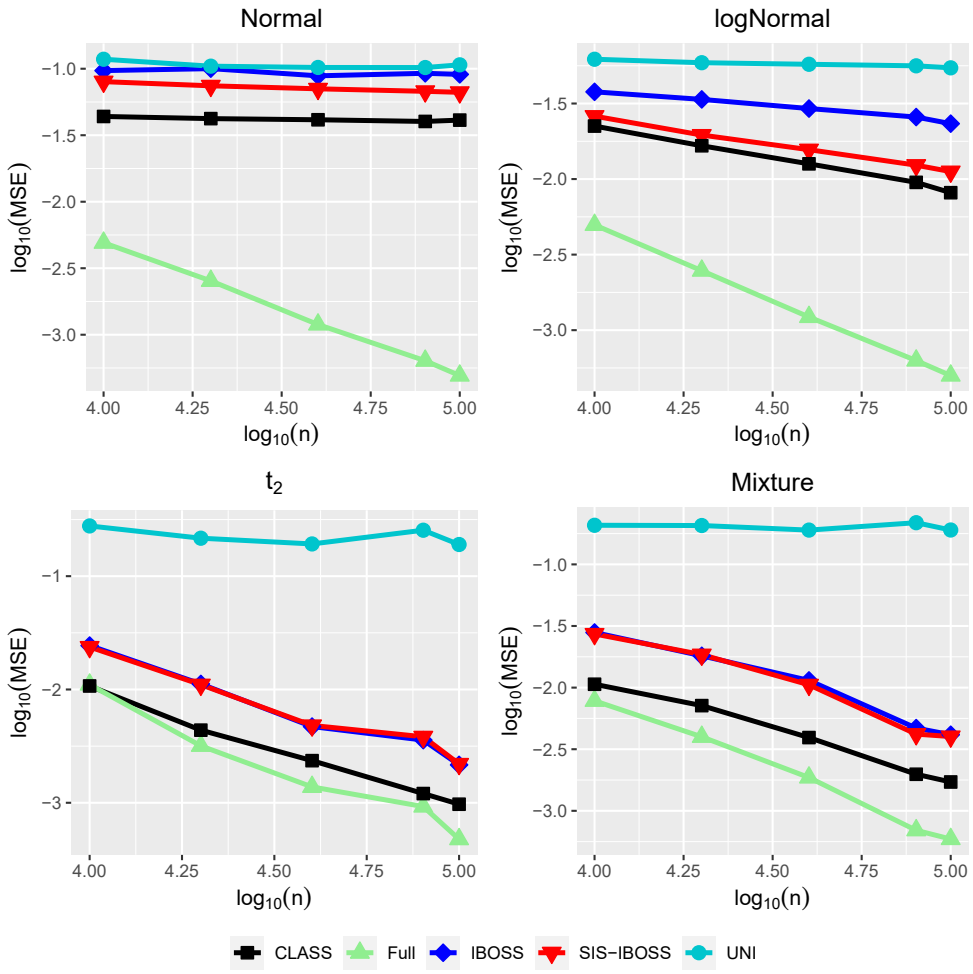

Figure 3: Variable selection performance for $k = 1000$, $p = 500$, $p_1 = 50$, and $\mathbf{\Sigma} = (0.5^{I(i \neq j)})$. The solid lines represent the power whereas the dashed lines represent the error.

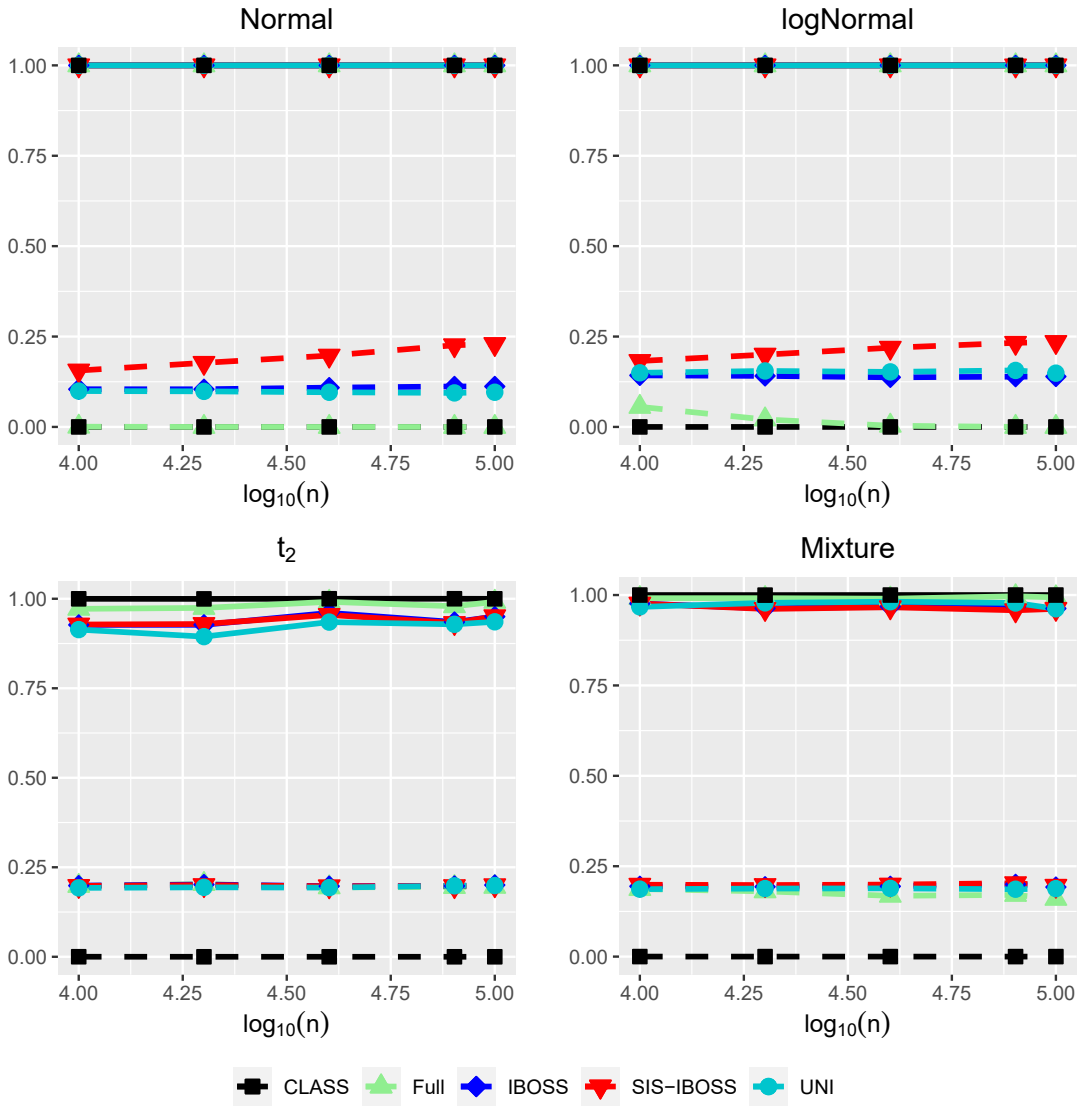

Figure 4: MSE for $k = 1000$, $p = 500$, $p_1 = 50$, and $\mathbf{\Sigma} = (0.5^{I(i \neq j)})$

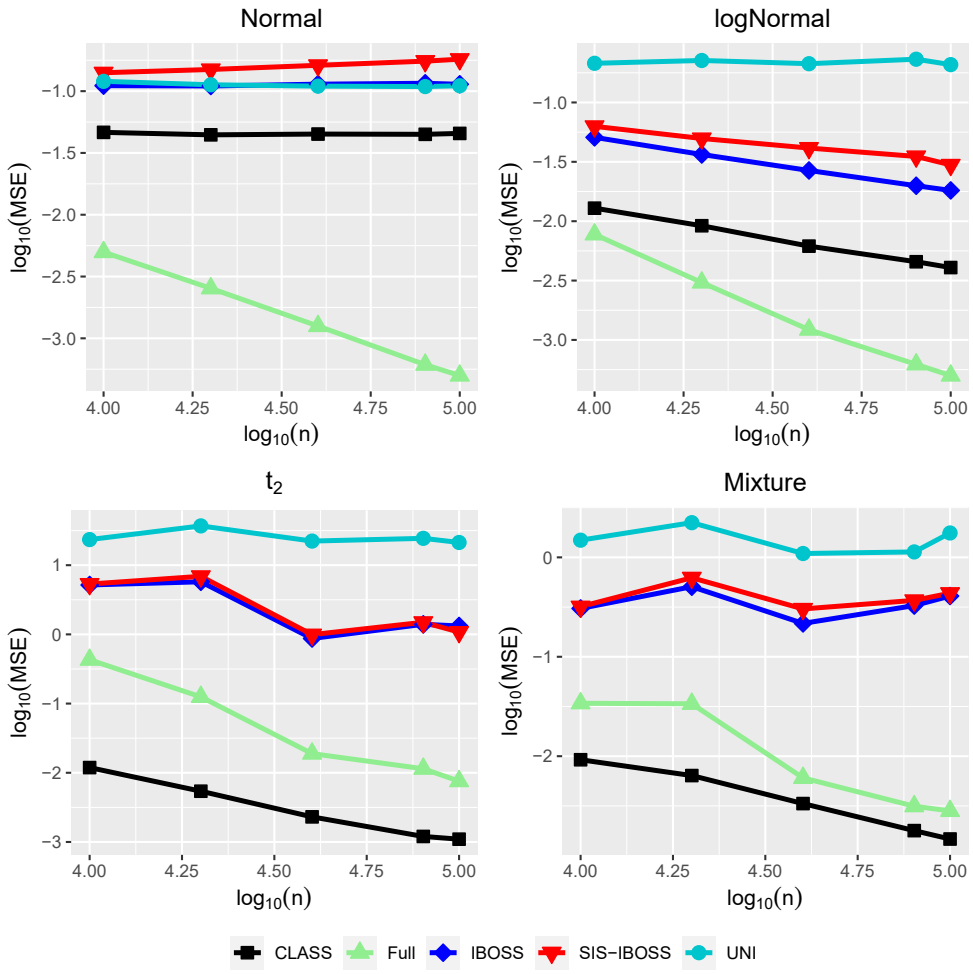

Figure 5: Variable selection and MSE for $n = 10^5$, $p = 500$, t_2, $p_1 = 50$, and $\mathbf{\Sigma} = (0.5^{I(i \neq j)})$

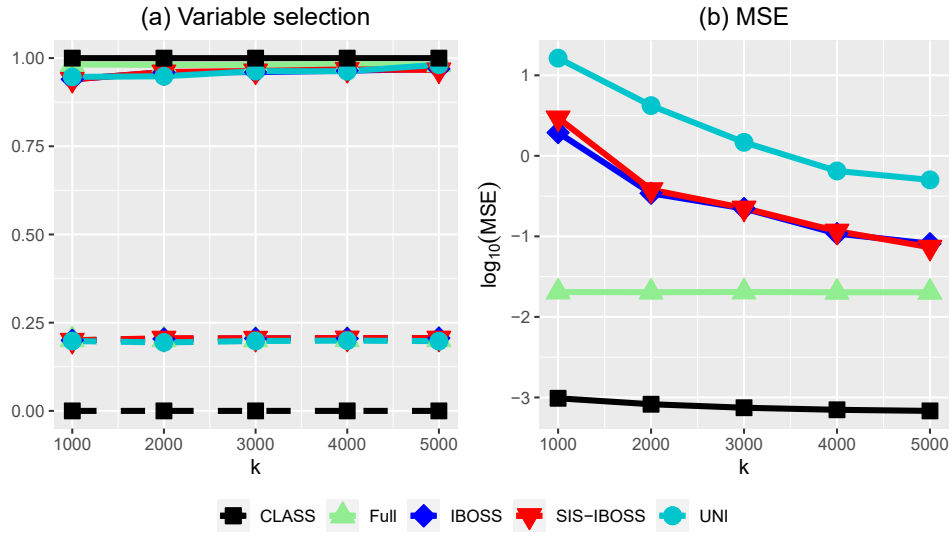

Figure 6: Variable selection and MSE for $k = 1000$, $p = 5000$, t_2, $p_1 = 50$, and $\mathbf{\Sigma} = (0.5^{I(i \neq j)})$

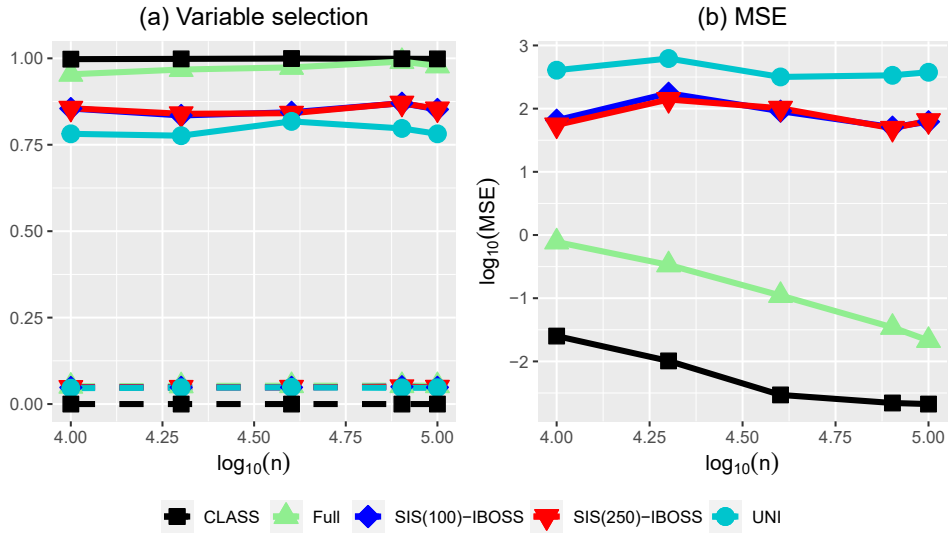

Table 2. CPU times (seconds) for different n with $p = 500$, $p_1 = 50$, $\mathbf{\Sigma} = (0.5^{I(i \neq j)})$, joint variable distribution t_2 and $k = 1000$

n	Full	UNI	IBOSS	SIS(100)--IBOSS	CLASS
5×10^3	2.35	0.54	0.98	0.58	51.82
5×10^4	33.96	0.53	1.81	0.91	50.39
5×10^5	361.66	0.54	9.21	3.88	51.36

Table 3. CPU times (seconds) for different p with $n = 5 \times 10^5$, $p_1 = 50$, $\mathbf{\Sigma} = (0.5^{I(i \neq j)})$, joint variable distribution t_2 and $k = 1000$

p	Full	UNI	IBOSS	SIS(100)--IBOSS	CLASS
100	16.78	0.19	1.95	2.31	18.31
250	55.03	0.42	4.79	2.95	43.11
500	361.66	0.54	9.21	3.88	51.36

Table 4. CPU times (seconds) for $n = 10^7$, $p = 100$, $p_1 = 50$, $\mathbf{\Sigma} = I_p$, joint variable distribution t_2 and $k = 1000$

n	UNI	IBOSS	SIS(80)--IBOSS	CLASS
10^7	0.10	32.22	32.83	25.86

on all criteria despite utilizing a far smaller subdata size of $k = 1000$ compared to other subdata selection methods. The differences in MSE are large which we attribute to CLASS being more successful at identifying virtually all of the active variables and collecting subdata that facilitates precise estimation of the corresponding parameters. The results in Table 5 are not entirely surprising. When CLASS performs better than FULL in terms of the MSE for prediction, it is expected to perform better than the other subdata methods since they are not expected to beat FULL. These other subdata methods perform the same analysis as FULL, but on fewer data points. Also, irrespective of the subdata size, the error rates for these methods tend to be no better than those for FULL, and can thus be much worse than those for CLASS. Similar results are seen for the mixture distribution (see the Supplementary Material). In addition, as demonstrated in Table 4, CLASS will become faster than IBOSS and SIS-IBOSS for very large n while continuing to perform better statistically. Table 6 for the Normal distribution shows a different picture. With this variable distribution, the difference between CLASS and other subdata selection methods was smaller when all the methods used subdata of the same size, and now that other methods can use more subdata (in some cases almost all of the data), they start to outperform CLASS in terms of MSE. However, in contrast to Table 5, the differences in MSE for Table 6 are very small. Also, CLASS is not outperformed in terms of variable selection. Similar observations as for the Normal distribution also apply for the logNormal distribution (see the Supplementary Material). In conclusion, with the same amount of CPU time, CLASS is for all cases studied here highly competitive, and in two of the four cases the clear winner in terms of both MSE and variable selection.

Table 5. MSE and variable selection performance for different subdata methods with approximately equal CPU times, different n and k, $p = 500$, $p_1 = 50$, $\mathbf{\Sigma} = (0.5^{I(i \neq j)})$ and joint variable distribution t_2

Method	k	time (s)	MSE	Power	Error
	For $n = 10^5$				
UNI	90000	63.13	93.00642	0.9892	0.1957
IBOSS	60000	54.53	53.00148	0.9932	0.2010
SIS(100)-IBOSS	75000	53.73	49.58365	0.9936	0.2005
CLASS	1000	50.13	0.084747	0.9998	0.0000
	For $n = 10^6$				
UNI	80000	55.58	110.0874	0.9906	0.1932
IBOSS	50000	59.72	103.6516	0.9800	0.1970
SIS(100)-IBOSS	70000	56.03	96.07541	0.9808	0.1963
CLASS	1000	52.47	0.000104	1	0.0000

Table 6. MSE and variable selection performance for different subdata methods with approximately equal CPU times, different n and k, $p = 500$, $p_1 = 50$, $\mathbf{\Sigma} = (0.5^{I(i \neq j)})$ and joint variable distribution Normal

Method	k	time (s)	MSE	Power	Error
	For $n = 10^5$				
UNI	90000	53.74	0.001053	1	0
IBOSS	60000	51.60	0.000843	1	0
SIS(100)-IBOSS	75000	46.89	0.000663	1	0
CLASS	1000	43.56	0.044859	1	0
	For $n = 10^6$				
UNI	80000	48.97	0.000687	1	0.0000
IBOSS	50000	69.27	0.001016	1	0.0000
SIS(100)-IBOSS	70000	64.64	0.000679	1	0.0002
CLASS	1000	44.90	0.045218	1	0.0000

4.1 Real Data

Similar to [44], we consider the Blog Feedback data as a real case study. This data is obtained from blog posts, with the ultimate goal of predicting the number of comments to a future blog post. The data is available at the UCI repository at https://archive.ics.uci.edu/ml/datasets/BlogFeedback and is described in [4]. The raw HTML documents of the blog posts are crawled and processed to select the blog posts that were published no earlier than 72 hours before the selected base date/time. Counting both the training and test datasets together, this data has $n = 52,397$ and $p = 280$. The 280 variables in the data include average, standard deviation, min, max, and median of the length of time between the publication of the blog post and "current" time, the length of the blog post, the number of comments in the last 24 hours before the base time and so on. We use a random 80-20% split of this dataset as training and test data and with $k = 5600$, we compute the mean squared prediction error of the test set, where

$$MSPE = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (y_i - \hat{y}_i)^2$$

and $\hat{y}_i$ is the predicted value. For the full data (Full) and IBOSS subdata on $p = 280$ variables (IBOSS), predicted values are obtained by first fitting a LASSO model followed by OLS. For CLASS, predicted values come from the model obtained by Algorithm 1. The choice of $k = 5600$ is inspired by previous studies such as [44] so that IBOSS can select 20 observations per variable. In Table 7, we present the average mean squared prediction errors for these three methods computed over 100 random splits of the full data in training and test data. CLASS produces a smaller MSPE than IBOSS, and the full data does better than the two subsampling methods.

Table 7. MSPE over 100 random training-test splits on the Blog Feedback data

Full	IBOSS	CLASS
902.04	1043.65	1003.25

5. CONCLUDING REMARKS

With a very large number of observations n and a large number of variables p, it can be computationally challenging to identify the active variables and build a good model for prediction. Subdata selection methods are one way to deal with this problem. If a linear regression model is a reasonable model for the data, then IBOSS is known to be an excellent method for subdata selection. However, as discussed in previous sections, IBOSS subdata can only be found when the sample size $k > 2p$. Moreover, even when $k \leq 2p$, the

subdata tends to be better when selecting IBOSS subdata by only using the active variables.

In this work, under the assumption of effect sparsity, we propose a method, CLASS, that attempts to do just that. We first devise a variable selection method that uses small uniform random samples of the full data to conduct multiple LASSO runs. As demonstrated, our variable selection approach is better than applying Lasso to IBOSS subdata, SIS(p^+)-IBOSS subdata, or the full data. We then obtain IBOSS subdata using only these selected variables, and fit a linear model to the subdata using OLS estimation. CLASS results in much smaller mean squared errors than IBOSS and SIS-IBOSS, even after adding OLS estimation at the end of these methods. For heavy-tailed joint distributions of the variables, CLASS can also improve on using the full data.

Due to the repeated applications of LASSO, CLASS takes a larger computing time than the competing subdata selection methods. However, if n is very large, CLASS becomes computationally less expensive than IBOSS and, with values of p^+ that are not too small, than SIS-IBOSS. The superior statistical performance of CLASS, both in terms of variable selection and prediction accuracy, makes a strong case for its use over the competing methods. CLASS is faster than analyzing the full data and is applicable in situations where full data analysis may not be possible.

FUNDING

JS gratefully acknowledges support through NSF grants DMS-1935729 and DMS-2304767.

SUPPLEMENTARY MATERIAL

The Supplementary Material is available online and contains more performance results corresponding to the cases in Table 1.

Received October 2020

REFERENCES

[1] Ai, M., Yu, J., Zhang, H. and Wang, H. (2019). Optimal subsampling algorithms for big data regressions. *Statistica Sinica*. https://doi.org/DOI:10.5705/ss.202018.0439.

[2] Ai, M., Wang, F., Yu, J. and Zhang, H. (2021). Optimal subsampling for large-scale quantile regression. *Journal of Complexity* **62** 101512.

[3] Bach, F. R. (2008). Bolasso: model consistent lasso estimation through the bootstrap. In *Proceedings of the 25th international conference on Machine learning* 33–40.

[4] Buza, K. (2014). Feedback prediction for blogs. In *Data analysis, machine learning and knowledge discovery* 145–152 Springer.

[5] Cai, L. and Zhu, Y. (2015). The challenges of data quality and data quality assessment in the big data era. *Data science journal* **14**.

[6] Chen, X. and Xie, M. -G. (2014). A split-and-conquer approach for analysis of extraordinarily large data. *Statistica Sinica* **24**(4) 1655–1684.

[7] Cheng, Q., Wang, H. and Yang, M. (2020). Information-based optimal subdata selection for big data logistic regression. *Journal of Statistical Planning and Inference* **209** 112–122.

[8] Derezinski, M., Warmuth, M. K. and Hsu, D. J. (2018). Leveraged volume sampling for linear regression. *Advances in Neural Information Processing Systems* **31**.

[9] Drineas, P. and Mahoney, M. W. (2016). RandNLA: randomized numerical linear algebra. *Communications of the ACM* **59**(6) 80–90.

[10] Drineas, P., Mahoney, M. W. and Muthukrishnan, S. (2006). Sampling algorithms for l2 regression and applications. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm* 1127–1136.

[11] Fan, J. and Lv, J. (2008). Sure independence screening for ultra-high dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **70**(5) 849–911.

[12] Fan, Y., Liu, Y. and Zhu, L. (2021). Optimal subsampling for linear quantile regression models. *Canadian Journal of Statistics* **49**(4) 1039–1057.

[13] Fithian, W. and Hastie, T. (2014). Local case-control sampling: Efficient subsampling in imbalanced data sets. *Annals of statistics* **42**(5) 1693.

[14] Friedman, J., Hastie, T. and Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software* **33**(1) 1–22. https://doi.org/10.18637/jss.v033.i01.

[15] Fu, W. and Knight, K. (2000). Asymptotics for lasso-type estimators. *The Annals of Statistics* **28**(5) 1356–1378.

[16] Han, L., Tan, K. M., Yang, T. and Zhang, T. (2020). Local uncertainty sampling for large-scale multiclass logistic regression. *The Annals of Statistics* **48**(3) 1770–1788.

[17] Hedayat, A. S., Sloane, N. J. A. and Stufken, J. (1999) *Orthogonal arrays: theory and applications*. Springer Science & Business Media.

[18] Joseph, V. R. and Mak, S. (2021). Supervised compression of big data. *Statistical Analysis and Data Mining: The ASA Data Science Journal* **14**(3) 217–229.

[19] Joseph, V. R. and Vakayil, A. (2022). Split: An optimal method for data splitting. *Technometrics* **64**(2) 166–176.

[20] Kiefer, J. and Wolfowitz, J. (1960). The equivalence of two extremum problems. *Canadian Journal of Mathematics* **12** 363–366.

[21] Kleiner, A., Talwalkar, A., Sarkar, P. and Jordan, M. I. (2014). A scalable bootstrap for massive data. *Journal of the Royal Statistical Society: Series B: Statistical Methodology* **76**(4) 795–816.

[22] Lin, N. and Xi, R. (2011). Aggregated estimating equation estimation. *Statistics and Its Interface* **4**(1) 73–83.

[23] Ma, P. and Sun, X. (2015). Leveraging for big data regression. *Wiley Interdisciplinary Reviews: Computational Statistics* **7**(1) 70–76.

[24] Ma, P., Mahoney, M. W. and Yu, B. (2015). A statistical perspective on algorithmic leveraging. *The Journal of Machine Learning Research* **16**(1) 861–911.

[25] Mahoney, M. W. (2011). Randomized algorithms for matrices and data. *arXiv preprint arXiv:1104.5557*.

[26] Mak, S. and Joseph, V. R. (2018). Support points. *The Annals of Statistics* **46**(6A) 2562–2592.

[27] Meinshausen, N. (2007). Relaxed lasso. *Computational Statistics & Data Analysis* **52**(1) 374–393.

[28] Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **72**(4) 417–473.

[29] Meng, C., Xie, R., Mandal, A., Zhang, X., Zhong, W. and Ma, P. (2020). LowCon: A design-based subsampling approach in a misspecified linear model. *Journal of Computational and Graphical Statistics* 1–32.

[30] Meng, C., Zhang, X., Zhang, J., Zhong, W. and Ma, P. (2020). More efficient approximation of smoothing splines via space-filling

basis selection. *Biometrika* **107**(3) 723–735.
[31] Muecke, N., Reiss, E., Rungenhagen, J. and Klein, M. (2022). Data-splitting improves statistical performance in overparameterized regimes. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics* (G. Camps-Valls, F. J. R. Ruiz and I. Valera, eds.). *Proceedings of Machine Learning Research* **151** 10322–10350. PMLR. https://proceedings.mlr.press/v151/muecke22a.html.
[32] Schifano, E. D., Wu, J., Wang, C., Yan, J. and Chen, M. -H. (2016). Online updating of statistical inference in the big data setting. *Technometrics* **58**(3) 393–403.
[33] Shao, L., Song, S. and Zhou, Y. (2022). Optimal subsampling for large-sample quantile regression with massive data. *Canadian Journal of Statistics*.
[34] Song, Q. and Liang, F. (2015). A split-and-merge Bayesian variable selection approach for ultrahigh dimensional regression. *Journal of the Royal Statistical Society: Series B: Statistical Methodology* **77** 947–972.
[35] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* **58**(1) 267–288.
[36] Ting, D. and Brochu, E. (2018). Optimal subsampling with influence functions. In *Advances in neural information processing systems* 3650–3659.
[37] Vakayil, A. and Joseph, V. R. (2021). Data Twinning. *arXiv preprint arXiv:2110.02927*.
[38] Wang, C., Chen, M. -H., Schifano, E., Wu, J. and Yan, J. (2016). Statistical methods and computing for big data. *Statistics and its interface* **9**(4) 399.
[39] Wang, H. (2019). More Efficient Estimation for Logistic Regression with Optimal Subsamples. *Journal of Machine Learning Research* **20**(132) 1–59.
[40] Wang, H. and Ma, Y. (2020). Optimal subsampling for quantile regression in big data. *Biometrika*.
[41] Wang, H., Yang, M. and Stufken, J. (2019). Information-based optimal subdata selection for big data linear regression. *Journal of the American Statistical Association* **114**(525) 393–405.
[42] Wang, H., Zhu, R. and Ma, P. (2018). Optimal subsampling for large sample logistic regression. *Journal of the American Statistical Association* **113**(522) 829–844.
[43] Wang, L., Elmstedt, J., Wong, W. K. and Xu, H. (2021). Orthogonal subsampling for big data linear regression. *The Annals of Applied Statistics* **15**(3) 1273–1290.
[44] Wang, X., Yang, M. and Li, W. (2022). Efficient Data Reduction Strategies for Big Data and High-Dimensional LASSO Regressions. *in preparation*.
[45] Xue, Y., Wang, H., Yan, J. and Schifano, E. D. (2020). An online updating approach for testing the proportional hazards assumption with streams of survival data. *Biometrics* **76**(1) 171–182.
[46] Yao, Y. and Wang, H. (2019). Optimal subsampling for softmax regression. *Statistical Papers* **60**(2) 235–249.
[47] Yao, Y. and Wang, H. (2021). A review on optimal subsampling methods for massive datasets. *Journal of Data Science* **19**(1) 151–172.
[48] Yao, Y. and Wang, H. (2021). A Selective Review on Statistical Techniques for Big Data. *Modern Statistical Methods for Health Research* 223–245.
[49] Yu, J. and Wang, H. (2022). Subdata selection algorithm for linear model discrimination. *Statistical Papers* 1–24.
[50] Yu, J., Wang, H., Ai, M. and Zhang, H. (2020). Optimal distributed subsampling for maximum quasi-likelihood estimators with massive data. *Journal of the American Statistical Association* 1–12.
[51] Yuan, M. and Lin, Y. (2007). On the non-negative garrotte estimator. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **69**(2) 143–161.
[52] Zhang, H. and Wang, H. (2021). Distributed subdata selection for big data via sampling-based approach. *Computational Statistics & Data Analysis* **153**.
[53] Zhang, T., Ning, Y. and Ruppert, D. (2021). Optimal sampling for generalized linear models under measurement constraints. *Journal of Computational and Graphical Statistics* **30**(1) 106–114.
[54] Zhao, P. and Yu, B. (2006). On model selection consistency of Lasso. *The Journal of Machine Learning Research* **7** 2541–2563.

Rakhi Singh. Department of Mathematics and Statistics, Binghamton University, USA.
E-mail address: rsingh@binghamton.edu

John Stufken. Department of Statistics, George Mason University, USA.
E-mail address: jstufken@gmu.edu