Leveraging Methods for Subsampling: Towards a Realistic Evaluation

Changrui (Charlie) Liu
University of Kentucky
Lexington, KY
cli322@g.uky.edu

Byran J. Smucker
Miami University
Oxford, OH
smuckerb@miamioh.edu

1 Introduction

In the context of linear regression, we compare several subsampling algorithms in terms of their actual, rather than theoretical, computational time. The goal is to assess whether relatively sophisticated leverage-based subsampling methods provide as much information as random subsampling, when accounting for the time it takes to both form and analyze the subsample. For a small simulation, we find that analyzing a small fraction of the data using leverage-based methods takes as long or longer than analyzing the entire dataset. Work is ongoing, but this provides initial evidence that leverage-based subsampling is not currently practically viable.

Society is producing increasingly massive datasets.One of the most important questions in the big data era is whether the computational infrastructures and analysis methods used to investigate these large datasets can keep up with the increasing data volume. Computational efficiency continues to improve, but is being outpaced by the increase in the size of data. One intuitive way to handle this disparity is to take a tractable subsample from large data and analyze this sample. A uniform random sample is the most straightforward approach, but under certain conditions it performs poorly (Ma and Sun, 2015; Ma et al., 2015). Despite its problems, the uniform sampling approach has a tremendous advantage over many other sampling methods: it is very fast.

Other subsampling methods can be used, including those based upon information criteria (Wang et al., 2019) and leverages (Ma et al., 2015; Drineas et al., 2012). Here, we investigate two leverage-based methods and compare them to the time it takes to compute the full dataset. If the leverage-based methods are not faster than the full data analysis, we conclude that with current subsampling technology, random sampling will provide more information per time unit than these more intricate subsampling approaches.

2 Methods

In this section, we briefly describe the subsampling methods that we compare, and discuss the measures we've taken to ensure a fair comparison between the methods.

Throughout, we assume an underlying linear regression model $\mathbf{Y} = X\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\mathbf{Y}$ is the $n \times 1$ response vector, X is the $n \times p$ model matrix, $\boldsymbol{\beta}$ is a $p \times 1$ vector of regression parameters, and $\boldsymbol{\epsilon}$ is a vector of error terms with distribution $N(\mathbf{0}, \sigma^2 I)$. Of interest is the least squares estimate of $\boldsymbol{\beta}$, which is $\hat{\boldsymbol{\beta}}_{\text{OLS}} = (X^T X)^{-1} X^T \mathbf{Y}$. The vector of predictions is $\hat{\mathbf{Y}} = X\hat{\boldsymbol{\beta}}_{\text{OLS}} = H\mathbf{Y}$ where $H = X(X^T X)^{-1} X^T$ and is called the hat matrix with leverages, denoted as $h_{11}, \ldots, h_{nn}$, on its diagonals. For a diagonal weight matrix W with weights $w_1, \ldots, w_n$, the weighted least squares (WLS) estimates are $\hat{\boldsymbol{\beta}}_{\text{WLS}} = (X^T W X)^{-1} X^T W \mathbf{Y}$.

2.1 Uniform Subsampling (UNIF)

The uniform subsampling algorithm simply subsamples r observations such that each observation is chosen with probability $\pi_i = \frac{1}{n}$ and the OLS estimates, computed in $O(rp^2)$ time, are obtained based on the subsampled dataset. When leverages are highly skewed and subsample sizes are relatively small, this method can have high variance.

2.2 Shrinkage Leverage-Based Subsamples

The basic leveraging subsampling approach constructs a subsampling distribution based on the h_{ii}, the statistical leverages. Ma et al. (2015) proposed a method that improves upon the basic leverage-based algorithm by shrinking the probabilities of high and low leverage points toward the uniform probabilities. This shrinkage leverage approach (SLEV) is described as below:

1. Compute the leverages of the full design matrix, $h_{11}, \ldots, h_{nn}$.
2. Use a convex combination of normalized statistical leverages and uniform probabilities as the subsampling probability distribution: $\pi_i = \alpha \frac{h_{ii}}{p} + (1-\alpha)\frac{1}{n}$. Obtain a subsample of size r from the full dataset.
3. Compute the WLS estimates on the subsample with weights $w_i = \frac{1}{\sqrt{r\pi_i}}$.

Intuitively, this method is more informative than sampling uniformly, because it increases the likelihood of obtaining high-leverage points in the sample. However, the leverages themselves are of the same order to compute—$O(np^2)$—as obtaining the least squares estimates *of the full data*.

2.3 Approximate Leverage-Based Subsamples

Drineas et al. (2012) obtained an approximation to the leverages in $o(np^2)$ time. The approximate leverage-based subsampling algorithm, called BFSLEV, is:

1. Compute the approximate leverage scores of the design matrix, denoted by $\tilde{h}_{11}, \ldots \tilde{h}_{nn}$.

   (a) Construct a random $p \times n$ matrix Π_1 as well as a random $p \times r_2$ matrix Π_2, each element of each matrix a 1 or -1 with probability 0.5.

   (b) Let R be from a QR decomposition of $\Pi_1 X$.

   (c) Calculate and report $\tilde{h}_{ii}$, the leverages of $U = XR^{-1}\Pi_2$.

2. Same as SLEV step 2, except $\pi_i = \alpha \frac{\tilde{h}_{ii}}{p} + (1-\alpha)\frac{1}{n}$.

3. Same as SLEV step 3.

2.4 A Fair Comparison

We make no claims that our implementation is fast; indeed, we are using the statistical software **R** to perform the calculation instead of `C`, `C++`, or `Fortran`. However, we have endeavored to make the comparisons fair, in the sense that we are using the same type of calculations across the methods we are comparing. For instance, we are using the `qr.solve()` function for matrix inversions, and QR-based methods are used throughout the calculation of exact as well as approximate leverage scores. In none of the methods do we use optimized `C`-based functions like `lm()`, since this would unfairly privilege the full data and exact-leverage methods.

3 Results

In this section we provide results from a small simulation study to compare runtimes for full data OLS, SLEV, and BFSLEV.

3.1 Simulation Design

In the simulation study, we tried combinations of $n =$ 10,000, 100,000, and 1,000,000, and $p = 10$ and 100. For each pair of (n, p), we generated X from the standard multivariate normal distribution, which results in leverage scores that are relatively uniform. $\mathbf{Y}$ is taken as a sample from the standard normal distribution. For all SLEV and BFSLEV algorithms, we took $\alpha = 0.9$ as suggested in Ma et al. (2015). We performed 100 simulations for each method and scenario.

To our end of comparing the running time, we treat $\hat{\boldsymbol{\beta}}_{\text{OLS}}$ calculated from the full sample as the conditionally true beta. The simulated model here still follows from section 2, except that we do not need to know the unconditionally true $\boldsymbol{\beta}$.

3.2 Simulation Results

Figure 1 demonstrates that for the range of simulations we performed, the full analysis consistently takes less time than either of the leverage-based methods. Also, BFSLEV appears to generally outpace the method based upon exact leverages.

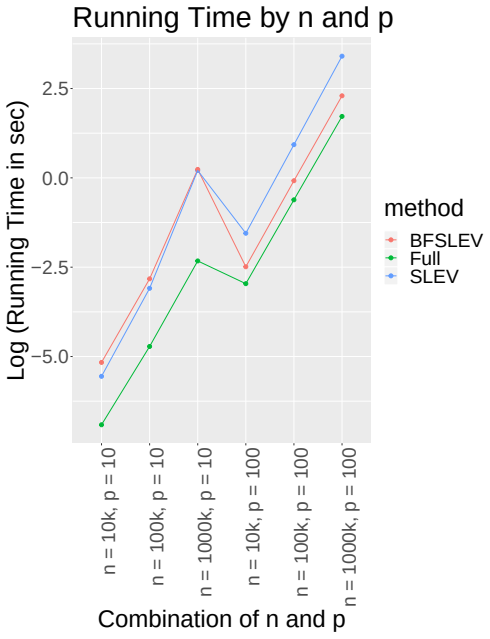

Figure 1: A comparison of running times using different methods

4 Discussion and Future Work

This work is ongoing and has built upon the contributions of several other people[1]. We will be increasing the number of simulation repetitions, expanding the number of scenarios we consider, and considering other subsampling methods for comparison. For instance, IBOSS (Wang et al., 2019) subsamples based upon an information-based criterion, though it is limited to only simple, first-order regression models.

Based on this work, we find the leverage-based methods computationally impractical. For large n, uniform subsampling seems to be better than leverage-based approaches.

References

Petros Drineas, Malik Magdon-Ismail, Michael W Mahoney, and David P Woodruff. 2012. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13(Dec):3475–3506.

Ping Ma and Xiaoxiao Sun. 2015. Leveraging for big data regression. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(1):70–76.

Ping Ma, Michael W Mahoney, and Bin Yu. 2015. A statistical perspective on algorithmic leveraging. *The Journal of Machine Learning Research*, 16(1):861–911.

HaiYing Wang, Min Yang, and John Stufken. 2019. Information-based optimal subdata selection for big data linear regression. *Journal of the American Statistical Association*, 114(525):393–405.

[1]We acknowledge the contributions of Yuexi Wang, Xinyuan Liu, Greg Keslin, and Karsten Maurer.