

This item is likely protected under Title 17 of the U.S. Copyright Law. Unless on a Creative Commons license, for uses protected by Copyright Law, contact the copyright holder or the author.

Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing scholarworks-group@umbc.edu and telling us what having access to this work means to you and why it's important to you. Thank you.

A Coreset Learning Reality Check

Fred Lu^{1, 2, 3}, Edward Raff^{1, 2, 3}, James Holt³

¹ Booz Allen Hamilton

² University of Maryland, Baltimore County

³ Laboratory for Physical Sciences

Lu.Fred@bah.com, Raff.Edward@bah.com, holt@lps.umd.edu

Abstract

Subsampling algorithms are a natural approach to reduce data size before fitting models on massive datasets. In recent years, several works have proposed methods for subsampling rows from a data matrix while maintaining relevant information for classification. While these works are supported by theory and limited experiments, to date there has not been a comprehensive evaluation of these methods. In our work, we directly compare multiple methods for logistic regression drawn from the coreset and optimal subsampling literature and discover inconsistencies in their effectiveness. In many cases, methods do not outperform simple uniform subsampling.

1 Introduction

It has become common, even routine, for researchers to have massive datasets in reach for modeling and prediction. Using standard machine learning tools for datasets with over millions of observations n or thousands of features d is challenging when many algorithms have superlinear scaling with the data dimensions. In terms of time and resources, even training a simple linear model becomes an expensive endeavor.

A natural workaround for excessively large data is to subsample a set of rows m without losing significant information. The specific form of information to be preserved, and hence the best strategy to accomplish this, depends on the task. In the logistic regression setting, we have a data matrix $X \in \mathbb{R}^{n \times d}$, labels $y \in \{-1, 1\}^n$, and a loss function $f(x) = \log(1 + e^{-x})$, and the goal is to find the weights β^* which minimizes the objective function $L(\beta; X, y) = \sum_{i=1}^n f(y_i x_i^\top \beta)$. There are three natural quantities which a subsampling method may choose to approximate:

1. The model fit, measured as the sum of validation losses
2. The maximum likelihood estimate (MLE) $\hat{\beta}_{MLE}$
3. The model's performance on unseen data

By subsampling, a practitioner will fit a logistic regression over only m points instead of n . Each of the m points is permitted an instance weight w_i for the fitting procedure,

which indicates the level of representation of that point in the linear model.

Recently a number of works proposed coresets for logistic regression which target quantity (1). A coreset consists of a set of $m \ll n$ weighted points whose objective function approximates that of the full dataset for any β . More precisely, if there is a subset C of the data consisting of x_{c_1}, \dots, x_{c_m} and weights w_1, \dots, w_m , such that for all $\beta \in \mathbb{R}^d$,

$$\left| \sum_{i=1}^m w_i f(y_{c_i} x_{c_i}^\top \beta) - L(\beta; X, y) \right| \leq \epsilon \cdot L(\beta; X, y)$$

then the points form an $(1 + \epsilon)$ -coreset of X .

The coreset literature suggests sampling strategies, known as *sensitivity sampling*, over the rows in X so that with probability $1 - \delta$ the resulting subsample is a $(1 + \epsilon)$ -coreset. The sensitivity can be viewed intuitively as the worst-case influence of a given data point on the objective over all values of β . Points which are highly influential should be more likely to be sampled into the coreset. While the sensitivity is intractable because it involves optimizing over all values of β , various strategies are used to approximate (and more specifically, upper-bound) the sensitivity in order to sample data. These include k -means clustering (Huggins, Campbell, and Broderick 2016), leverage scores (Munteanu et al. 2018), and row norms (Tolochinsky and Feldman 2018). Using theoretical frameworks based on (Feldman and Langberg 2011; Braverman et al. 2016), probabilistic bounds are derived to relate a given subsample size m to ϵ and δ (for example, the minimum m needed to achieve some ϵ).

Because the primary quantity of interest has been approximating the objective function, attention has not been given to how the minimizer $\hat{\beta}_C$ of the coreset objective relates to $\hat{\beta}_{MLE}$. In particular, the standard $(1 + \epsilon)$ -coreset needs to hold for all β , whereas we may only be interested in the approximation performance of the coreset in a neighborhood of $\hat{\beta}_{MLE}$. Recently the statistics literature on optimal subsampling has investigated this area by focusing on target (2). Wang, Zhu, and Ma (2018) proposes a subsampling procedure with conditions where $\hat{\beta}_C$ converges to $\hat{\beta}_{MLE}$. While the asymptotic performance of such procedures has been studied (Wang, Zhu, and Ma 2018; Ting and Brochu 2018),

finite sample bounds like those in the coreset literature have not yet been developed.

Finally, while some of the above works have empirically examined coreset performance on (3), this has not been a main focus. Since large-scale logistic regression models are often trained in order to get predictive power over unseen data, it is crucial that prediction quality does not greatly deteriorate when subsampling is applied.

Based on these frameworks, numerous sampling strategies have been presented in the literature with theoretical support. While each respective method was supported by experimental results, a comprehensive evaluation has been missing. Some limitations of previous studies are:

- Each method is benchmarked with at most one or two competitors on a small selection of real-world datasets. The datasets vary greatly in size depending on the study, from thousands of rows to millions.
- The evaluation metric has not consistent across these works, with some assessing objective function error, some measuring the error of estimating $\hat{\beta}$ itself, etc.
- Subsample size ranges: Some works examine coresets with only hundreds of samples even on large datasets, while others do not consider very small coreset sizes. In our work we find that some methods are more effective in certain size regimes.

In our experiment design we account for these limitations by benchmarking most known subsampling methods for logistic regression over a large variety of realistic datasets. We are the first to present a thorough empirical comparison of these approaches, over a range of important metrics. In the process we discover that most existing algorithms show inconsistencies in performance across datasets, with many of them not outperforming the uniform sampling baseline. Furthermore, we find that one of the earliest coreset methods still performs among the best, despite being ignored in all recent evaluations. We additionally show that our results are robust across regularization settings.

In section 2 we give context on the importance of reproducibility research. We then detail the coreset and optimal subsampling algorithms in section 3 and our evaluation procedure in section 4. Our results are shown and discussed in section 5. Finally we conclude in section 6.

2 Related Work

The primary focus of our study is on the matter of reproducible research. Considerable work has been done in replicating research as described by the original manuscripts (Raff 2019) and in developing repeatable infrastructure so that a manuscript’s code can be easily re-run (Claerbout and Karrenbach 1992; Forde et al. 2018a; Zaharia et al. 2018; Forde et al. 2018b). While such research is valuable for understanding the nature of replication and its incentives (Raff 2021, 2022; Raff and Farris 2022), it also repeats any methodological errors from the original manuscript

where conclusions could change under different evaluation (Bouthillier, Laurent, and Vincent 2019).

This methodological concern has grown in recent years, with many sub-fields in machine learning discovering different results when a more thorough evaluation is performed. Musgrave, Belongie, and Lim (2020) showed that metric learning was not improving with new triplet mining and losses, but rather more general neural architectures, data augmentation, and optimization strategies. Blalock et al. (2020) found inconsistencies in neural architecture search and built a new benchmarking framework to standardize. Dacrema, Cremonesi, and Jannach (2019) and Sun et al. (2020) also found that in recommender systems many results could be replicated as described in the original works, but when a simple baseline was properly tuned the advantage of proposed techniques disappeared. This is similar to our own work, except our baseline is naive uniform random sampling. Eggensperger et al. (2021) similarly showed hyperparameter optimization conclusions could change in a thorough benchmark, though most methods did improve upon naive baselines. Our work follows in performing a through evaluation of coreset methods, where we find that most prior seminal methods often fail to improve upon the naive baseline of uniform random sampling.

A sub-concern of methodological evaluation is the method by which conclusions about one method being “better” than another are drawn. Early work in the machine learning field proposed a number of alternative schemes on how to balance valid statistical testing with the computational limits of performing the necessary experiments (Kubat and Matwin 1997; Bradley 1997; Dietterich 1998; Alpaydin 1999), but multiple later results (Demšar 2006; Benavoli, Corani, and Mangili 2016) showed that using a non-parametric test was a highly effective means of drawing a valid conclusion, with computational limits mostly unnecessary given advancements in computing speed. This is not true in all cases. In particular deep learning can be highly compute intensive and needs alternative strategies (Bouthillier et al. 2021), but our work is focused on logistic regression coresets and so we can use the non-parametric test for robust conclusions. While other approaches to robust conclusions have been proposed recently we consider them beyond our intended scope (Soboroff 2018; Berrar 2016).

3 Subsampling Methods

We identified and implemented several subsampling methods for logistic regression from the literature. All methods follow the same general procedure, as described in Algorithm 1. The final weights are designed so that the sum of losses over the subsample is on the scale of the loss over all n points, which enables the coreset objective to approximate the full data objective. The subsample is then used to fit a weighted logistic regression model, i.e. $\hat{\beta}_C$ minimizing $\sum_{j=1}^m w_j f(y_{c_j} x_{c_j}^T \beta)$. This subsampled estimate is finally compared with the full data MLE. We next briefly describe the methods being evaluated along with implementation details that we found were necessary in order to repro-

Coreset	Weights	Coreset size	Details
k -means	$s_i = \frac{n}{1 + \sum_{j=1}^k G_j^{(-i)} e^{-R \ G_j^{(-i)} - Z_i\ }}$	$O\left(\frac{\bar{s}}{\epsilon^2} (d+1) \log \bar{s} + \log \frac{1}{\delta}\right)$	G_j are Z_i assigned to cluster j , $Z_i = X_i y_i$
Leverage	$s_i = \ Q_i\ _2 + \frac{1}{n}$, where $Z = QR$	$\tilde{O}\left(\frac{\mu_y(X) \sqrt{n} d^{3/2}}{\epsilon^2}\right)$	$Z_i = -X_i y_i$, X is μ -complex
Monotonic	$s_i = \frac{132\sqrt{k} \ p_i\ + 2}{i}$	$O\left(\frac{t}{\epsilon^2} (d \log t + \log \frac{1}{\delta})\right)$	$p_i = \text{sorted}(x_i)$, $k = \frac{1}{2\lambda}$, $t = \sum s_i$
Lewis	$s_i = \text{Lewis}(X)$	$\tilde{O}\left(\frac{\mu_y(X)^2 d}{\epsilon^2}\right)$	μ -complex, <code>Lewis</code> in (Cohen and Peng 2015)
OSMAC (vc)	$s_i = y_i - p_i(\hat{\beta}) \cdot \ x_i\ $	–	p_i estimated probs. using pilot $\hat{\beta}$
OSMAC (mse)	$s_i = y_i - p_i(\hat{\beta}) \cdot \ M_X^{-1} x_i\ $	–	$M_X = \frac{1}{n} \sum p_i(\hat{\beta})(1 - p_i(\hat{\beta})) x_i x_i^\top$
uniform	$s_i = 1$	$\tilde{O}\left(\frac{n^{1-\kappa} d}{\epsilon^2}\right)$	$\ x_i\ _2 \leq 1$, regularization $\lambda \propto n^\kappa$

Table 1: Computational expressions (corresponding to line 1 of Algorithm 1) and theoretical size of a $(1 + \epsilon)$ -coreset for each coreset method evaluated. The notation \tilde{O} suppresses logarithmic factors. This is used in some methods like `Leverage` where the $\log 1/\delta$ is re-expressed as a multiplicative logarithmic constant, or the expression is otherwise very lengthy. Throughout $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$, λ is the regularization parameter, and δ is the failure probability.

duce the original results. We refer the reader to the original publications for the nuanced details of each method. This six prior methods, plus our naive baseline are detailed below. Furthermore, mathematical expressions for the coreset weights and theoretical sizes are shown in Table 1.

Algorithm 1 Coreset sampling procedure

Require: Datasets $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$, desired subsample size m , method to compute importance weights `CoresetMethod`

- 1: Compute $\{s_i\}_{i=1}^n \leftarrow \text{CoresetMethod}(X, y)$
- 2: **for** i in $1 : n$ **do**
- 3: $p_i \leftarrow s_i / \sum_{i=1}^n s_i$
- 4: Sample m rows from X, y using probabilities $\{p_i\}$ to get X_c, y_c
- 5: **for** j in $1 : m$ **do**
- 6: $w_j \leftarrow 1/(p_{c_j} \cdot n)$
- 7: **return** coreset $(X_c, y_c, w) = \{x_{c_j}, y_{c_j}, w_j\}_{j=1}^m$

k -means coreset (Huggins, Campbell, and Broderick 2016). This was one of the first coresets proposed for logistic regression. The theoretical analysis is based on the framework developed in (Braverman et al. 2016; Feldman and Langberg 2011), where the key step is to upper bound the sensitivity score of each data point. If the upper bound is too loose, then a larger m is needed to achieve the same ϵ coreset. This work used k -means clustering on a uniform subsample of the data to compute the upper bound importance weights based on distance from cluster centers. The method has two hyperparameters, the number of clusters k and the radius R of the ball containing β . We found that the results were robust to k and chose $k = 6$. While the theory requires β to be bounded for the results to hold, in practice we found $R = 1$ usually performed the best even if $\max_d |\hat{\beta}_{MLE}|$ exceeded 1.

The only other work to evaluate the k -means coreset is (Munteanu et al. 2018), where R was set to be unbounded. However, as $R \rightarrow \infty$ the importance weights converge to the uniform distribution, so their results are not representative of the effectiveness of the k -means method. No other

works have benchmarked this method, so we felt it was important to give it a fair chance.

Leverage coreset (Munteanu et al. 2018). The ℓ_2 -leverage scores have been used to approximate $\|X\theta\|_2$ with $\|X'\theta\|_2$ for a row-wise reduced matrix X' up to $(1 + \epsilon)$ relative error (Cohen and Peng 2015). In (Munteanu et al. 2018), the root leverage scores are shown to be an effective upper bound on the sensitivities. The scores are computed by taking the row-wise L2-norm of an orthonormal basis of X . Such a basis was obtained with the QR-decomposition. A further step bins the sampling probabilities into multiples of 2 to reduce the complexity of the procedure.

While an approximate QR-decomposition algorithm was used in the paper, it was faster for us to use the optimized `numpy.linalg.qr` routine for an exact solution. Another important detail for the method to give good results was to make sure any columns of 1s used to include the intercept were removed prior to computing Q .

Monotonic coreset (Tolochinsky and Feldman 2018). Unlike the previous coreset works, this work develops bounds on the sensitivity of logistic regression when regularization is used. The sampling weights are computed to be proportional to the lengths of the data points, scaled by the relative index of each point in sorted order. The weights are further scaled by \sqrt{k} where k is the regularization parameter (which in our version is equivalent to $\frac{1}{2\lambda}$). Thus their framework requires the use of regularization.

Lewis coreset (Mai, Musco, and Rao 2021). An ℓ_1 analog of the leverage scores, the `Lewis` scores approximate the quantity $\|X\theta\|_1$ up to $(1 + \epsilon)$ relative error. Using their similarity to hinge-like loss functions including the logistic loss, this work derives coreset results for such functions. While intuitively similar to the root leverage coreset, they are the first to show a linear rather than polynomial dependence on the feature dimension d via their proof technique. In practice, computing the `Lewis` weights requires iteratively computing the leverage scores t times using the algorithm of (Cohen and Peng 2015), thus giving an approximate runtime of t times that of the `Leverage` coreset. As reported

in their paper, we found that the method fully converged for $t = 20$ while being acceptable for $t = 5$, so we used $t = 5$ to avoid excessive run times.

OSMAC (Wang, Zhu, and Ma 2018). This work shows that under relatively mild conditions, the MLE $\hat{\beta}$ under a subsampling distribution converges to the full data MLE as the coreset size increases. The `osmac_mse` is proposed as such a subsampling method. Furthermore the distribution of $\hat{\theta} - \hat{\theta}_{MLE}$ is shown to be asymptotically normal, with the resulting covariance matrix being minimized by the `OSMAC_MSE` weights. The scores are computed by taking into account the residual $(y - \hat{y})$, normalized by a covariance matrix weighted by the expected binomial variance at each point. This has a similar form to the leverage while using label and model fit information, and can also be interpreted as the influence function as in (Ting and Brochu 2018). The authors additionally propose a second method `OSMAC_VC` which replaces the covariance matrix with the identity for computational efficiency.

Both methods require the actual MLE θ to estimate the residuals and weighted Hessian (in the first case) accurately, so an initial pilot estimate using a uniform subsample is required. In our experiments we used a subsample of $m/2$ for the pilot estimate. We found that results were unreliable at small sizes. Checking with the source code, we identified that the authors weighted the pilot sample to have evenly balanced classes. Implementing this ourselves led to more stable pilot estimates.

Influence subsampling (Ting and Brochu 2018). The influence function uses a second-order approximation of the loss to estimate the effect of infinitesimally upweighting each point in the dataset, as motivated in (Hampel 1974; Koh and Liang 2017). For the logistic regression case, the form of the influence is equivalent to the score used in `OSMAC_MSE`. Thus we do not show this method separately. Like the previous, this work also proves the asymptotic optimality of influence sampling, albeit with a different framework.

Uniform coreset. It was shown in (Munteanu et al. 2018) that in unconstrained logistic regression there exist general datasets for which no sublinear coreset exists (Munteanu et al. 2018). In order to bound the sensitivity, they rely on the notion of μ -complexity, which measures for a dataset, over all β , the worst-case ratio of sum-log-odds between incorrectly and correctly classified points. The smaller this quantity is, the more effective the `Leverage` coreset is. (Curtin et al. 2019) pointed out that when sufficient regularization is used, the sensitivity of logistic regression is controlled over all datasets. Therefore, uniform subsampling is an effective coreset for all datasets and is in fact nearly optimal under sufficient regularization.

In our experiments we use very weak regularization and will show that even in nearly unconstrained settings that the uniform coreset is competitive with other methods.

Combining coresets. Coreset methods can be composed, e.g. by recursively applying the subsampling procedure. For

example, (Munteanu et al. 2018) give additional theoretical bounds for a recursive coreset. Due to the added complexity when stacking results, we consider these methods to be outside the scope of our evaluation. Thus in all methods we limit the coreset generation to a single round of subsampling.

4 Evaluation procedure

Dataset	n	d	d (num.)	% pos
chemreact	24059	100	100	3.00
census	30932	100	6	24.1
bank	39128	51	8	11.3
webspam	126185	127	127	60.7
kddcup	469319	41	35	80.3
covtype	551961	54	10	51.2
bitcoin	2770862	24	6	1.42
SUSY	4500000	18	18	45.7

Table 2: Characteristics of datasets used in the study, with rows n , features d , numeric features (i.e. not categorical columns which are one-hot encoded), and percent positive class. Additional details on dataset preprocessing and sources are in the Appendix.

Our aim is to unify previous experimental methods into a single comprehensive procedure. In this procedure we aim to resolve the limitations from previous evaluations:

Datasets. While previous work used at most 3 datasets, we evaluate on 8 datasets which include previously used ones as well as new ones. The sizes range from 24000 to nearly 5 million (Table 2).

Evaluation metric. In accordance with the three aims of subsampling as described in the Introduction, we assess the following metrics:

- Relative negative log-likelihood (NLL) error to assess the model fit:

$$|L(\hat{\beta}_C; X, y) - L(\hat{\beta}_{MLE}; X, y)| / L(\hat{\beta}_{MLE}; X, y)$$

- The mean squared error (MSE) of the MLE:

$$\|\hat{\beta}_C - \hat{\beta}_{MLE}\|_2^2$$

- Relative ROC of the subsampled model on validation data: $ROC(\hat{\beta}_C, X, y) / ROC(\hat{\beta}_{MLE}, X, y)$

Subsample sizes. We consider 25 subsample sizes on a logarithmically spaced grid from 200 to 100000. (For datasets with fewer rows we limit the upper end of the evaluation range.) We replicate each procedure 50 times and report the medians and inter-quartile intervals. This is a much broader range of evaluation than in previous work.

Regularization. Most coreset methods directly handle regularization by adding the penalty to the loss function, giving for example $f(x; \beta) = \log(1 + e^{-x}) + \lambda \|\beta\|_2^2$. Some previous works, including (Munteanu et al. 2018; Wang, Zhu,

and Ma 2018) only evaluate their methods on unconstrained logistic regression, while certain methods require the use of regularization (Tolochinsky and Feldman 2018). Furthermore, recent theoretical work has shown that the uniform subsampling method produces an effective coreset in the presence of regularization (Curtin et al. 2019). In our main experiments, we use weak L2 regularization at $\lambda = 10^{-5}$ as some of the datasets produce unstable results without any regularization. Coefficient estimates generally did not further change when further weakening λ . We conduct additional analysis at varying λ values and our main conclusions did not change.

Our λ is defined with respect to the objective $\frac{1}{2n} \sum f(y_i x_i^\top \beta) + \lambda \|\beta\|_2^2$. Because of the normalizing n , when we fit a subsampled logistic regression, we normalize the modeling weights w_i to maintain the same level of regularization with respect to the data loss term. For loss parameterizations without the normalizing n , the original w_i should be used.

5 Results

Our primary comparison is shown in Figure 1, with remaining datasets in Figure 4. The relative performance of subsampling algorithms vary by dataset but are generally consistent across metrics. This shows that accurately estimating $\hat{\beta}_{MLE}$ with $\hat{\beta}_C$ leads to effective logistic regression subsampling regardless of the metric of interest.

Methods such as Leverage, Lewis, and OSMAC_MSE are very effective on KDD cup relative to the *uniform* baseline, but not so much on others. In fact, the uniform baseline frequently outperforms other coreset methods across datasets. In addition, while previous works which had stopped showing the *k-means* coreset due to perceived lack of performance, our results find that despite being the earliest method, it turned out to be one of the most competitive when properly tuned. The *Monotonic* method was formulated to work with regularization, so we may expect it to perform weaker. However, increasing the L2 penalty still does not improve it over the uniform baseline (Figure 3). The OSMAC methods show large variability, with OSMAC_MSE giving strong performances on Webb spam and bitcoin but worse performances on other datasets, especially at small subsampled sizes. This may be due to challenges in obtaining a reliable inverse Hessian matrix with small sample sizes.

We follow recommendations by (Demšar 2006; Benavoli, Corani, and Mangili 2016) and use a non-parametric test to determine if there is a significant difference between all coreset methods. We compute the non-parametric Dunn test using the JASP software. This software performs a multiple test correction per metric, but not across metrics. Since we have three metrics, this means the proper threshold for significance is $\alpha \leq 0.05/3 \approx 0.0167$. We first look at the results in Table 3, where each method is compared against the uniform sampling baseline. This result shows that when accounting for the number of tests performed, only the OSMAC methods beat the uniform baseline and only when measuring by MSE of β . We remind the reader this is a somewhat

Pyrrhic victory, as Figures 1 and 4 show that both OSMAC variants perform worse than uniform sampling in different regimes of the coreset size.

Table 3: Comparisons with uniform baseline, with multiple test corrected p -value in the three right most columns. We can see that none of the coreset methods are significantly different than a uniform random sampling baseline by the NLL or ROC metrics. Only OSMAC outperforms uniform sampling for the MSE metric.

Comparison	NLL	β MSE	ROC
k-means - uniform	0.123	0.08	0.318
Leverage - uniform	1.000	0.527	1.000
Lewis - uniform	1.000	0.249	1.000
Monotonic - uniform	0.055	0.29	0.23
OSMAC_MSE - uniform	0.758	0.004	1.000
OSMAC_VC - uniform	0.057	0.006	0.075

Given the apparent failure of these methods to beat the naive baseline, it is natural to wonder how such a situation came to be. First, we remind the reader that in Figure 1 the behavior of the coreset algorithms varies from dataset to dataset, and our work is testing more datasets than prior works have done. By chance and happenstance, testing on fewer datasets may lead one to make conclusions of superior results, a common issue in most sciences today (Ioannidis 2005) and in machine learning competitions (Elkan 2001). For example, Webb spam and KDD cup were two of the most widely used datasets in the coreset literature, and coreset methods happen to perform well on those.

Furthermore, most studies only included one or two baselines. It only takes one paper to draw conclusions of improvement against a baseline for later works to follow suit and exclude the “beaten” baseline in favor of newer methods. We highlighted an example of this where the *k-means* method turned out as one of the best performers but was never evaluated in later works. Through this lens it becomes clear how subsequent papers can, following reasonable practices of replicating prior methods as their baseline, result in an erroneous conclusion of improvement. This erroneous conclusion can still occur when even when testing on more datasets once a less ideal baseline model enters evaluation. Our experiment is in fact the first comprehensive comparison among several prior methods. As shown in Table 4, there may be a statistically significant difference between two methods while each has no statistical difference with the uniform random baseline.

Finally one may ask if the overall performance does not differ, does the distribution of the performances differ? We show that they do in Figure 2, but that the uniform baseline is still competitive in all cases. Instead we see that the *Monotonic* and OSMAC methods have poorly behaved outliers in their distribution of scores, depending on the method of evaluation being used. From visual inspection, we note that *k-means*, *Leverage*, and *Lewis* appear the most effective of the coreset methods.

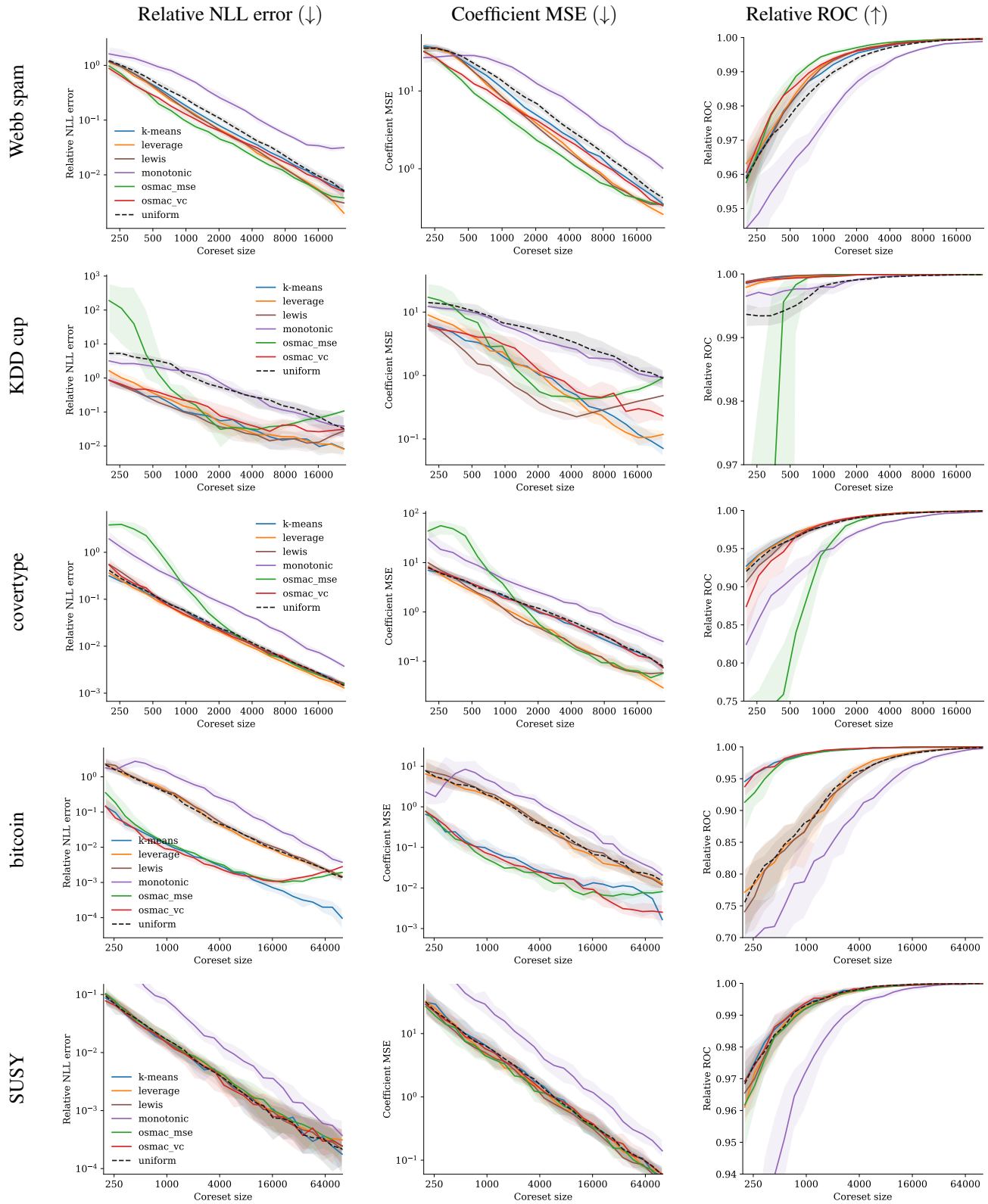


Figure 1: All subsampling methods are compared over a range of 25 subsample sizes. Each row represents one dataset and each column represents a different evaluation metric. Each point represents a median of 50 replications, with shaded areas representing interquartile range. The five largest datasets are shown here, with the remaining deferred to Figure 4.

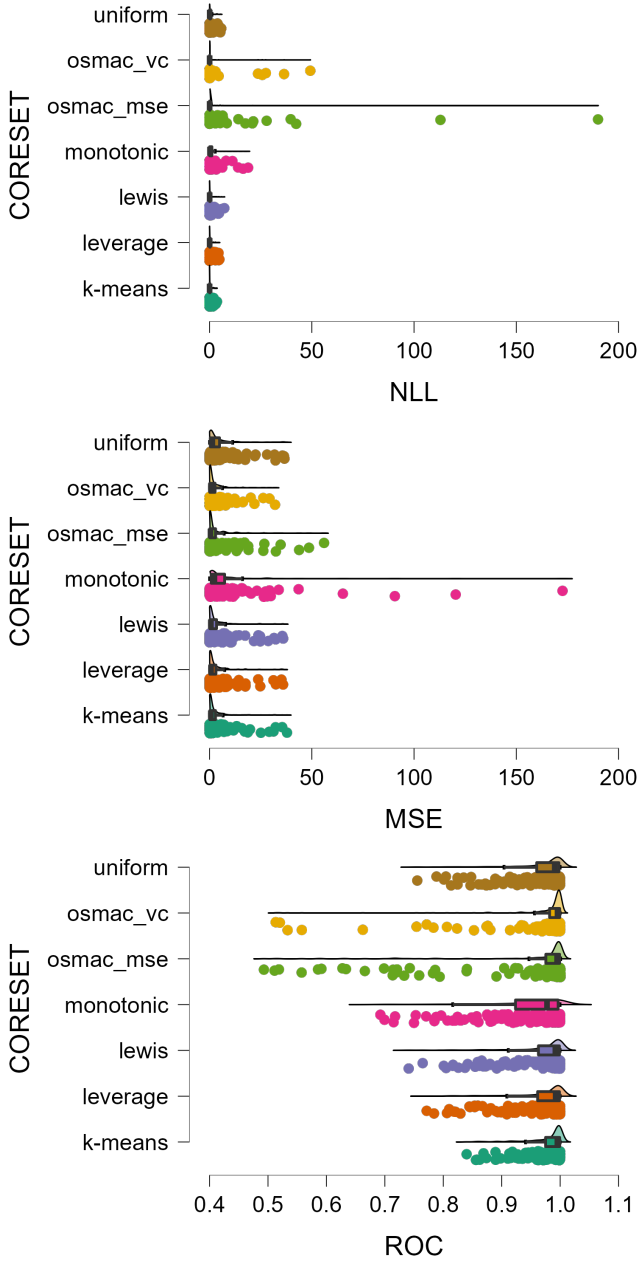


Figure 2: Comparing the distribution of scores between each method for all three metrics. In all cases the uniform baseline is as competitive as any other approach. We note the *Monotonic* method has worse outliers when measured by NLL or MSE, and that the *OSMAC* methods have a worse tail when measured by ROC.

5.1 Effect of regularization

As discussed in section 3, regularization impacts the effectiveness of coresets methods in logistic regression. In particular, although results from (Munteanu et al. 2018) indicate that the availability of coresets depends on the complexity of the data, the uniform baseline is known to be effective on all

Table 4: Comparison of coresets methods with each other. In many cases one coreset method is significantly better than another, but not better than uniform random sampling as shown in Table 3. This helps explain why many works have drawn conclusions of significance by comparing against other methods, but failing to include the competitive but naive baseline.

Comparison	NLL	β MSE	ROC
k-means - Leverage	1	1	1
k-means - Lewis	1	1	1
k-means - Monotonic	$<1e-4$	$<1e-4$	$<1e-4$
k-means - OSMAC_MSE	1	1	1
k-means - OSMAC_vc	1	1	1
Leverage - Lewis	1	1	1
Leverage - Monotonic	$<1e-4$	$<1e-4$	0.009
Leverage - OSMAC_MSE	1	1	1
Leverage - OSMAC_vc	1	1	1
Lewis - Monotonic	$<1e-4$	$<1e-4$	0.02
Lewis - OSMAC_MSE	1	1	1
Lewis - OSMAC_vc	1	1	0.645
Monotonic - OSMAC_MSE	$<1e-4$	$<1e-4$	$<1e-4$
Monotonic - OSMAC_vc	$<1e-4$	$<1e-4$	$<1e-4$
OSMAC_MSE - OSMAC_vc	1	1	1

datasets when the model is regularized (Curtin et al. 2019). We run a followup experiment where we vary the L2 penalty of the classifiers beyond that of our main experiment. Our findings in Figure 3 confirm experimentally that increasing regularization makes the gap between uniform sampling and the best-performing methods vanish.

One may ask whether further decreasing regularization strength in our main experiment would impact our findings by weakening the uniform subsample. We replicate our experiment with $\lambda = 10^{-7}$ with results in Table 5 and Appendix Table 6. We find that the situation becomes worse, and no method outperforms uniform random sampling. Further lowering λ hardly changed the regression coefficients, so we expect similar results for any $\lambda < 10^{-7}$. Similarly, as we showed earlier in accordance with recent theoretical work that uniform sampling only improves with stronger regularization, we conclude that our results are likely to hold true over all λ .

6 Conclusions

In this work we have performed a thorough evaluation against many seminal methods used in the coreset and sub-sampling literature, used with respect to logistic regression. When compared against a naive baseline of uniform random selection of a subset, and measured by three possible metrics of improvement, we find that almost none of these classic approaches improves upon the naive baseline. Our results call into question the need for larger diversity of evaluation sets and benchmarks to be used in coreset research.

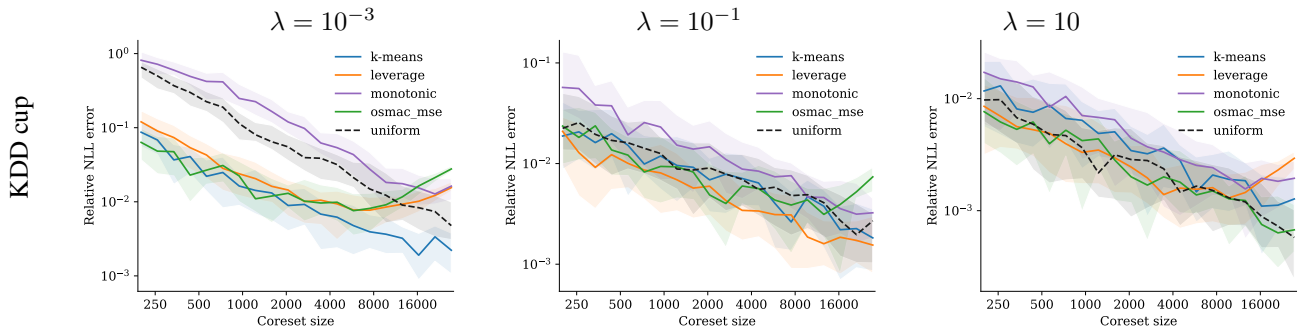


Figure 3: On the KDD Cup 99 dataset, increasing L2 regularization noticeably improves the efficiency of uniform subsampling relative to other methods. At high regularization, no advantage is observed from any other method. This shows concordance with the theoretical results in (Curtin et al. 2019). Similar effects can be seen on other datasets in Figure 5.

Table 5: Comparisons with uniform baseline for logistic regression coresets, with multiple test corrected p -value in the three right most columns. Regularization is further weakened to $\lambda = 10^{-7}$ compared to the main experiment. Note the final column shows that OSMAC- ν C is significant worse than uniform, so none of the methods are statistically improving over the naive baseline.

Comparison	NLL	β MSE	ROC
k-means - uniform	0.140	0.482	0.288
Leverage - uniform	1.000	0.733	1.000
Lewis - uniform	1.000	0.18	1.000
Monotonic - uniform	0.215	1.000	0.548
OSMAC_MSE - uniform	1.000	0.433	1.000
OSMAC- ν C - uniform	1.000	0.152	1.000

References

- Alpaydin, E. 1999. Combined 5×2 cv F Test for Comparing Supervised Classification Learning Algorithms. *Neural Comput.*, 11(9): 1885–1892.
- Benavoli, A.; Corani, G.; and Mangili, F. 2016. Should We Really Use Post-Hoc Tests Based on Mean-Ranks? *Journal of Machine Learning Research*, 17(5): 1–10.
- Berrar, D. 2016. Confidence curves: an alternative to null hypothesis significance testing for the comparison of classifiers. *Machine Learning*, 1–39.
- Blalock, D.; Gonzalez Ortiz, J. J.; Frankle, J.; and Gutttag, J. 2020. What is the State of Neural Network Pruning? In *Proceedings of Machine Learning and Systems 2020*, 129–146.
- Bouthillier, X.; Delaunay, P.; Bronzi, M.; Trofimov, A.; Nichyporuk, B.; Szeto, J.; Sepah, N.; Raff, E.; Madan, K.; Voleti, V.; Kahou, S. E.; Michalski, V.; Serdyuk, D.; Arbel, T.; Pal, C.; Varoquaux, G.; and Vincent, P. 2021. Accounting for Variance in Machine Learning Benchmarks. In *Machine Learning and Systems (MLSys)*.
- Bouthillier, X.; Laurent, C.; and Vincent, P. 2019. Unreproducible Research is Reproducible. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 725–734. Long Beach, California, USA: PMLR.
- Bradley, A. P. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7): 1145–1159.
- Braverman, V.; Feldman, D.; Lang, H.; Statman, A.; and Zhou, S. 2016. New frameworks for offline and streaming coreset constructions. *arXiv preprint arXiv:1612.00889*.
- Claerbout, J. F.; and Karrenbach, M. 1992. Electronic documents give reproducible research a new meaning. In *SEG Technical Program Expanded Abstracts 1992*, 601–604. Society of Exploration Geophysicists.
- Cohen, M. B.; and Peng, R. 2015. Lp row sampling by lewis weights. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, 183–192.
- Curtin, R. R.; Im, S.; Moseley, B.; Pruhs, K.; and Samadian, A. 2019. On coresets for regularized loss minimization. *arXiv preprint arXiv:1905.10845*.
- Dacrema, M. F.; Cremonesi, P.; and Jannach, D. 2019. Are we really making much progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems - RecSys '19*, 101–109. New York, New York, USA: ACM Press. ISBN 9781450362436.
- Demšar, J. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7: 1–30.
- Dietterich, T. G. 1998. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Comput.*, 10(7): 1895–1923.
- Eggensperger, K.; Müller, P.; Mallik, N.; Feurer, M.; Sass, R.; Klein, A.; Awad, N.; Lindauer, M.; and Hutter, F. 2021. HPOBench: A Collection of Reproducible Multi-Fidelity Benchmark Problems for HPO. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, NeurIPS, 1–36.
- Elkan, C. 2001. Magical Thinking in Data Mining: Lessons from CoIL Challenge 2000. In *Proceedings of the Sev-*

- enth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '01, 426–431. New York, NY, USA: Association for Computing Machinery. ISBN 158113391X.
- Feldman, D.; and Langberg, M. 2011. A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, 569–578.
- Forde, J.; Head, T.; Holdgraf, C.; Panda, Y.; Perez, F.; Nalvarte, G.; Ragan-kelley, B.; and Sundell, E. 2018a. Reproducible Research Environments with repo2docker. In *Reproducibility in ML Workshop, ICML'18*.
- Forde, J. Z.; Bussonnier, M.; Fortin, F.-A.; Granger, B. E.; Head, T. D.; Holdgraf, C.; Ivanov, P.; Kelley, K.; Pacer, M. D.; Panda, Y.; Pérez, F.; Nalvarte, G.; Ragan-Kelley, B.; Sailer, Z. R.; Silvester, S.; Sundell, E.; and Willing, C. 2018b. Reproducing Machine Learning Research on Binder. In *Machine Learning Open Source Software 2018: Sustainable communities*.
- Hampel, F. R. 1974. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346): 383–393.
- Huggins, J.; Campbell, T.; and Broderick, T. 2016. Coresets for scalable Bayesian logistic regression. *Advances in Neural Information Processing Systems*, 29.
- Ioannidis, J. P. 2005. Contradicted and initially stronger effects in highly cited clinical research. *Journal of the American Medical Association*, 294(2): 218–228.
- Koh, P. W.; and Liang, P. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, 1885–1894. PMLR.
- Kubat, M.; and Matwin, S. 1997. Addressing the Curse of Imbalanced Training Sets: One Sided Selection. In *Proceedings of the Fourteenth International Conference on Machine Learning*, volume 97, 179–186.
- Mai, T.; Musco, C.; and Rao, A. 2021. Coresets for classification—simplified and strengthened. *Advances in Neural Information Processing Systems*, 34: 11643–11654.
- Munteanu, A.; Schwiegelshohn, C.; Sohler, C.; and Woodruff, D. 2018. On coresets for logistic regression. *Advances in Neural Information Processing Systems*, 31.
- Musgrave, K.; Belongie, S.; and Lim, S.-N. 2020. A Metric Learning Reality Check. In *ECCV*.
- Raff, E. 2019. A Step Toward Quantifying Independently Reproducible Machine Learning Research. In *NeurIPS*, volume 32. Curran Associates, Inc.
- Raff, E. 2021. Research Reproducibility as a Survival Analysis. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*, volume 35, 469–478.
- Raff, E. 2022. Does the Market of Citations Reward Reproducible Work? In *ML Evaluation Standards Workshop at ICLR 2022*.
- Raff, E.; and Farris, A. L. 2022. A Siren Song of Open Source Reproducibility. In *ML Evaluation Standards Workshop at ICLR 2022*.
- Soboroff, I. 2018. Meta-Analysis for Retrieval Experiments Involving Multiple Test Collections. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM '18*, 713–722. New York, NY, USA: ACM. ISBN 978-1-4503-6014-2.
- Sun, Z.; Yu, D.; Fang, H.; Yang, J.; Qu, X.; Zhang, J.; and Geng, C. 2020. Are We Evaluating Rigorously? Benchmarking Recommendation for Reproducible Evaluation and Fair Comparison. In *Fourteenth ACM Conference on Recommender Systems, RecSys '20*, 23–32. New York, NY, USA: Association for Computing Machinery. ISBN 9781450375832.
- Ting, D.; and Brochu, E. 2018. Optimal subsampling with influence functions. *Advances in neural information processing systems*, 31.
- Tolochinsky, E.; and Feldman, D. 2018. Coresets for monotonic functions with applications to deep learning. *CoRR*, abs/1802.07382.
- 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.
- Zaharia, M. A.; Chen, A.; Davidson, A.; Ghodsi, A.; Hong, S. A.; Konwinski, A.; Murching, S.; Nykodym, T.; Ogilvie, P.; Parkhe, M.; Xie, F.; and Zumar, C. 2018. Accelerating the Machine Learning Lifecycle with MLflow. *IEEE Data Eng. Bull.*, 41: 39–45.

A Remaining figures for main experiment

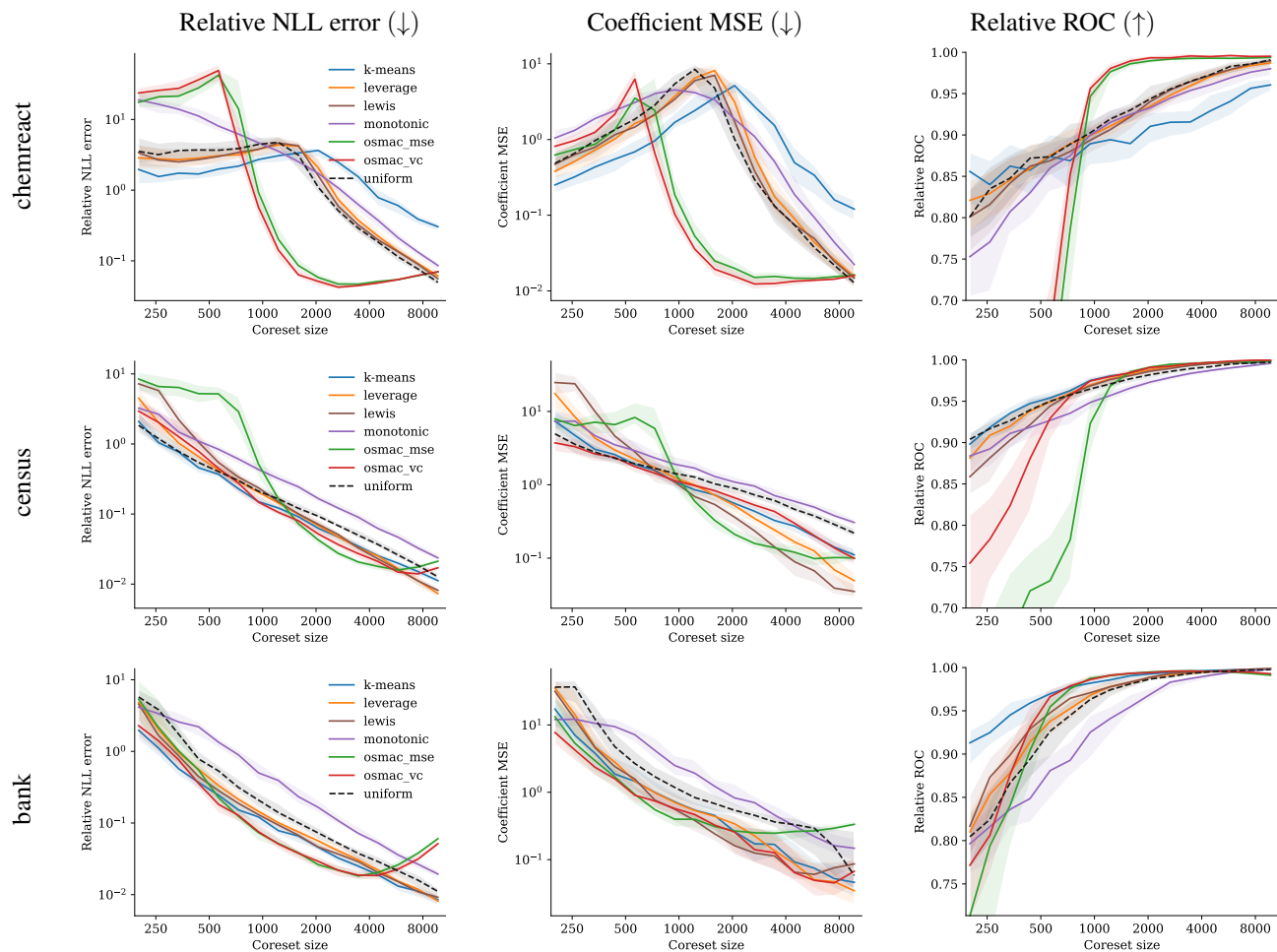


Figure 4: All subsampling methods are compared over a range of 25 subsample sizes. Each row represents one dataset and each column represents a different evaluation metric. Each point represents a median of 50 replications, with shaded areas representing interquartile range. The three smaller datasets are shown here, with the remaining in the main paper.

B Effect of L2 regularization

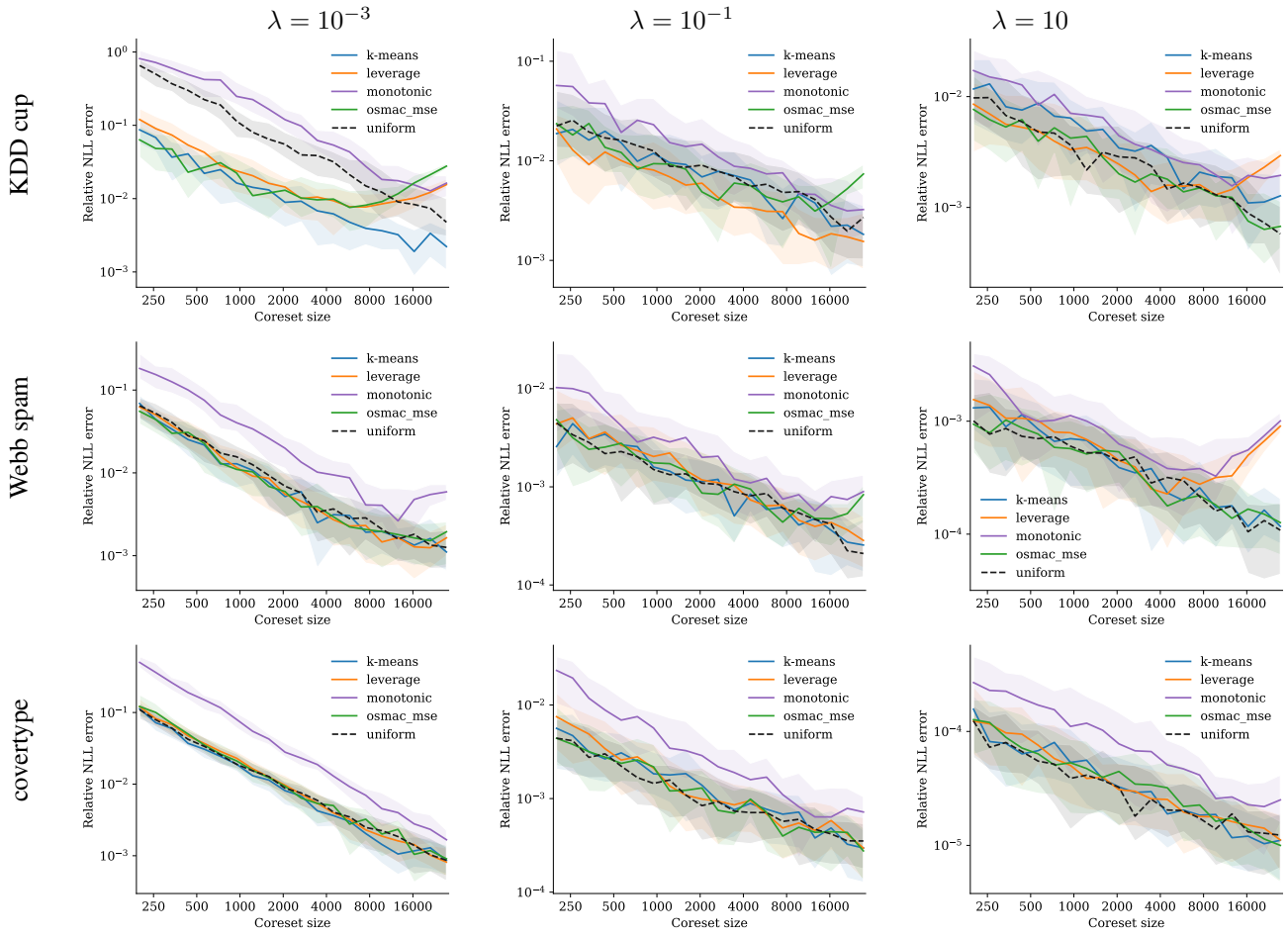


Figure 5: Additional results. Increasing L2 regularization noticeably improves the efficiency of uniform subsampling relative to other strong-performing methods. This shows concordance with the theoretical results in (Curtin et al. 2019).

C Replicated analysis for weakened regularization

The final table was not shown in main paper due to space, and is a comparison among coreset methods with regularization $\lambda = 10^{-7}$.

Table 6: Comparison of coreset methods with each other for logistic regression with further weakened regularization. In many cases one coreset method is significantly better than another, but not better than uniform random sampling as shown in Table 5.

Comparison	NLL	MSE	ROC
k-means - leverage	1	1	1
k-means - lewis	1	1	1
k-means - monotonic	2.710E-6	4.969E-4	5.764E-5
k-means - osmac_mse	0.769	1	1
k-means - osmac_vc	1	1	1
leverage - lewis	1	1	1
leverage - monotonic	0.001	0.001	0.013
leverage - osmac_mse	1	1	1
leverage - osmac_vc	1	1	1
lewis - monotonic	5.800E-4	9.681E-5	0.016
lewis - osmac_mse	1	1	1
lewis - osmac_vc	1	1	1
monotonic - osmac_mse	0.03	4.139E-4	0.09
monotonic - osmac_vc	0.002	7.351E-5	0.008
osmac_mse - osmac_vc	1	1	1

D L1 regularization

One may further ask how the subsampling methods perform with L1 regularization, which induces sparsity in the estimated coefficients. To study this question, we selected the three datasets with most features d , which are census, chemreact, and Webb spam, and repeated our experiment with the L1 penalty equal to $\lambda = 0.001$. We measured the support accuracy, which refers to the average agreement between $\hat{\beta}_C$ and $\hat{\beta}_{MSE}$ on whether a feature should be in the model. All methods improve as the coreset size increases, and methods like osmac_mse and leverage appear to be effective performers. However, it appears that feature selection with high accuracy requires subsampling at the high end of our tested range. As all prior theory for subsampling target quantities such as the coefficient MSE which are agnostic to sparsity, we think this area is of interest for future study.

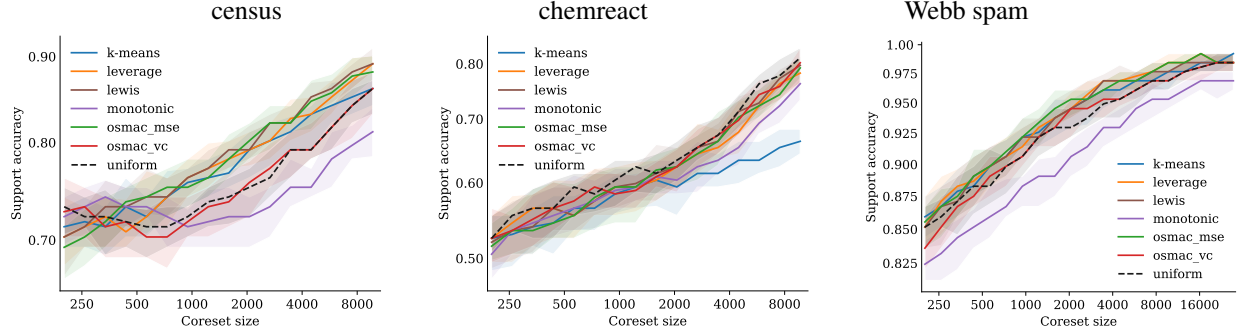


Figure 6: Support accuracy measures the average agreement between $\hat{\beta}_C$ and the $\hat{\beta}_{MSE}$ on whether a feature should be in the model, when L1 regularization is used to promote sparsity. The three datasets with the most features are shown here. $\lambda = 0.001$ is used in all three figures. Higher is better.

E Experiment details

E.1 Datasets

We detail our dataset sources and preprocessing steps here and in our code. Most datasets have previously been used in coreset papers, which was our primary motivation for including them. However, we include more features where possible, including categorical. In many cases, prior works only used a few numerical columns, for which we sometimes observed worse performance on the same datasets when including more features. We believe this makes our evaluation more challenging and realistic.

chemreact, Webb spam, covtype. For these datasets we directly used the preprocessed versions and code from (Huggins, Campbell, and Broderick 2016) (<https://bitbucket.org/jhhuggins/lrcoresets/src/master/>). These match standard usage of the datasets, except for Webb spam. This dataset contained 126185 entries instead of 350000 which was reported in their paper. We used the version that was available.

KDD cup 99. We obtained the 10% subset from <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>. Columns 2-4 are categorical which we one-hot encoded. We applied min-max scaling to the range $[-1, 1]$ and reserved 5% for the test set.

bank. We obtained the dataset from the UCI repository <https://archive.ics.uci.edu/ml/datasets/bank+marketing>. We kept the numerical variables age, duration, campaign, previous, emp.var.rate, cons.conf.idx, euribor3m, nr.employed and the categorical variables job, marital, education, default, housing, loan, contact, month, day_of_week, poutcome which we one-hot encoded. We applied min-max scaling and reserved 5% for the test set.

census. We obtained the dataset from the UCI repository <https://archive.ics.uci.edu/ml/datasets/census+income>. We identified and kept columns 1, 3, 5, 11, 12, 13 as numeric, and columns 2, 4, 6, 7, 8, 9, 10, 14 as categorical, which we one-hot encoded. We min-max scaled the dataset and reserved 5% as the test set.

SUSY. This is the most large-scale dataset we used in order to benchmark coresets on a more realistic application scale. We obtained it from <https://archive.ics.uci.edu/ml/datasets/SUSY>. All features are numeric so we applied min-max-scaling. As described in their readme, we reserved the final 500k rows as the test set, leaving 4.5 million for training.

bitcoin. This is a dataset of bitcoin ransomware attacks vs benign transactions, obtained from <https://archive.ics.uci.edu/ml/datasets/BitcoinHeistRansomwareAddressDataset>. We also selected this dataset because it was one of the larger non-simulated datasets available. We kept the numeric columns length, weight, count, looped, neighbors, income. We also used the year and day columns as categorical: We one-hot encoded the year directly. For day, we binned the days into 12 groups roughly corresponding to months, and one-hot encoded that. We applied min-max scaling and reserved 5% as the test set.

E.2 Coreset code sources

For k -means coreset we used their public code at <https://bitbucket.org/jhhuggins/lrcoresets/src/master/>. We modified their sampling function to return the entire (X, y, w) coreset so we could fit the subsampled model.

We found the R package for OSMAC from <https://rdrr.io/github/Ossifragus/OSMAC/>. We re-implemented the method in Python and referred to their code to resolve ambiguities.

To the best of our knowledge, no code was available for any other coreset method so we re-implemented them based on their papers.

E.3 Computational details

We distributed the coreset experiments over a cluster of 250 CPUs. A single experiment with 50 replications runs relatively quickly, from a few seconds to a few minutes depending on the size of the data. It was important to precompute the full data coefficient MLEs to save on runtime. We used seeds 0-49 for the replications. All models were implemented using numpy and scipy routines. The k -means coreset uses Cython which was compiled following the instructions in their code.

F k -means coresets parameters

Below we compare the effects of varying k and R . We show on two of the original datasets and metric (estimated test NLL) from the k -means coresets paper to be comparable to the original and not overfit in our main experiments. As seen, $R = 1.0$ is the best, while k does not make much difference.

