Hyperparameter Optimization for Small-Sample Machine Learning via Design of Experiments

Anonymous authors
Paper under double-blind review

Abstract

While current machine learning hyperparameter tuning methods have been thoroughly tested and show consistently high performance in large datasets, few studies have made efforts to rigorously assess their performance in small data regimes. Studies that have examined hyperparameter optimization in small datasets have found reduced generalization performance, poor correlation between validation and test error, and overly optimistic error estimates associated with the chosen hyperparameter. This has been observed across different hyperparameter optimization methods, including grid search and Bayesian optimization. We implement design of experiments principles to mitigate the bias between validation error and generalizable test error when hyperparameter tuning. Specifically, we utilize a surface fitted to a space-filling design on the hyperparameter space to generate optimal hyperparameter sets. Using fourteen publicly available datasets and repeated experiments via Monte Carlo simulation, we show that this method has similar generalizable test error compared to both grid search and Bayesian Optimization, but the bias between the validation error and generalizable test error is drastically reduced by 80-96% compared to both methods. As a secondary outcome of this work, we find that none of the evaluated methods of hyperparameter optimization offer consistent improvement over untuned models in our experiments, raising questions about the general efficacy of hyperparameter optimization in small sample regimes.

1 Introduction

Over the past several decades, machine learning (ML) has evolved from a niche scientific discipline to a tool set utilized across virtually all scientific disciplines. The most prominent recent breakthroughs in ML such as automatic speech recognition systems (Parthasarathi & Strom, 2019; Rao et al., 2017), autonomous vehicles (Geyer et al., 2020; Alqarqaz et al., 2023), and large language models (Liu et al., 2025) all reasonably fall under the umbrella of "big data". However, while the big data applications of ML receive the most attention, ML is increasingly being applied to scientific research in areas where the number of samples available for model training and validation is limited (Kokol et al., 2022). This is particularly prevalent in medical applications where the availability of subjects is frequently limited by the rarity of the disease in question. For example, Hyde et al. (2019) show that in supervised learning with autism spectrum disorder research, there have been several instances where the number of participants has been fewer than 20 people. Additionally, Garcia et al. (2020) have found that many studies in Alzheimer's research use ML techniques on datasets with fewer than 100 participants, with up to four internal subpopulations.

Notwithstanding the small samples gathered, ML methods are sometimes used to deal with the high dimensionality of the data. In speech analysis, random forests and support vector machines (SVM) are trained on datasets with hundreds or thousands of features, with some studies having fewer than 100 participants (Garcia et al., 2020; Berisha et al., 2021). Additionally, models such as tree based methods and SVMs have been used in disease identification on brain imaging datasets with as few as 20 observations (Vabalas et al., 2019). The application of ML to small datasets comes with some obvious challenges. Whereas the classical statistical methods typically applied in these regimes are applied to the whole dataset, best practices for ML analyses require three separate partitions: 1) training (model training) 2) validation (for hyperparameter

tuning / model selection) 3) testing (for performance evaluation). As dividing a small number of samples into three reasonably sized sets is generally impractical, standard practice instead relies on methods such as cross-validation and uses the same partitions for both hyperparameter tuning and model evaluation (Vabalas et al., 2019; Berisha et al., 2022) . Although deviation from fixed partitions is necessary in these regimes, this practice has led to some notable concerns on how accurately they measure model performance. Whereas the idea that a model's performance should improve with additional data is a basic paradigm of machine learning, some reviews have found that classification accuracies reported in some common tasks tend to decrease with sample size, indicating that smaller sample studies are likely reporting overly optimistic performance measures (Berisha et al., 2021; Vabalas et al., 2019). This is likely due to the fact that variance in a measure of classification accuracy is generally inversely related to the size of the test set. Thus, if model selection and hyperparameter optimization (HPO) fits to the maximum observed accuracy, smaller sample sizes are likely to outperform larger ones where the increase in expected accuracy is outweighed by the reduction in assessment variance. This problem could be further exacerbated by publication bias, if journals give preference to high performing models regardless of whether their performance accurately reflects their generalization error (Saidi et al., 2025). Overall, the consistent positive bias observed in small sample machine learning reflects a failure to prioritize accurate model assessments or to dis-incentivize practices prone to overfitting.

Regardless of the cause, the consistent bias observed in small sample machine learning results raises questions about whether current practices are in line with desired outcomes. Whereas concerns about p-hacking have been broadly accepted and guided editorial standards, data leakage in machine learning has not yet reached the same level of influence in machine learning. Most machine learning analyses still prioritize optimizing model performance as a primary goal. However it is worth questioning whether this should necessarily be the case. As Saidi et al. point out, inflated expectations from biased assessments that are not mirrored in model performance could lead to public distrust of the technologies and science broadly (Saidi et al., 2025). These biased assessments are also likely to lead to a misleading relative appraisal of different data representations or predictive models. In Gaussian noise classification (a common model for characterizing overfitting), larger feature sets have been shown to result in increased bias. Thus wider, more complex representations of data likely disproportionately benefit from this phenomenon. Similarly, when tuned to the validation set, models with high sensitivity to their hyperparameters are likely more prone to this overoptimism than insensitive ones (Probst et al., 2019; Vabalas et al., 2019).

Several authors have attempted to solve this discrepancy between reported error and out-of-sample error. For example, Vabalas et al. (2019) utilized several methods to measure a grid search (GS) of hyperparameter combinations that would not overfit to random Gaussian noise for a binary classification problem. They found that nested cross-validation and a train-test split gave unbiased estimates of out-of-sample accuracy, but both k-fold and partially nested cross-validation were overly optimistic. This confirms the findings of Combrisson & Jerbi (2015) and Varma & Simon (2006), who both found that k-fold cross-validation tends to overfit small samples. Additionally, Combrisson & Jerbi (2015) found that this bias holds regardless of the value of k. The conclusions of Vabalas et al. (2019) support the findings of Larracy et al. (2021), who found that nested cross-validation combined with feature filtering resulted in the best reflection of true feature signal among noisy features. With regards to a simple train-test split, Beleites et al. (2005) found that, while it is an unbiased estimator of the generalized error, the train-test split tends to be the highest variance estimator when applied to small sample datasets out of the methods they tested, including k-fold cross-validation. An et al. (2021) also found that the train-test split was prohibitive for small sample sizes if cross-validation was used on the training set to reduce this variance. Thus, the only method found to be unbiased and relatively low variance when estimating generalized error for a hyperparameter GS is nested cross-validation.

However, nested cross-validation can be difficult to implement when faced with an already limited dataset because it requires an additional partition of the data to estimate model error. This additional partitioning can reduce the efficiency of an already limited dataset and may not aid in creating a better model due to the stochasticity found in the inner optimization loop (Bischl et al., 2023). Thus, an additional method for out-of-sample error estimation that allows the utilization of the full dataset would be of particular interest for research that utilizes small sample ML. Two key elements of this estimation are the *calculation method* and *hyperparameter tuning*. The calculation method refers to how the error estimate is calculated (e.g.

averaging across cross-validation, train-test-splits). In contrast, hyperparameter tuning for ML methods refers to finding the combination of hyperparameters that yields the lowest model error via the calculation method. For some of the aforementioned studies, the hyperparameter tuning was accomplished via GS (Varma & Simon, 2006; Vabalas et al., 2019), which trains models using a grid of provided values, then selects the combination that yields the lowest error. Although assessment bias has been primarily attributed to the calculation method (Vabalas et al., 2019), the method of hyperparameter tuning also bears some responsibility. Whereas the error in the assessment of the individual models is nearly unbiased, it is the selection of hyperparameters that minimize this error which leads to the bias. Given a large number of hyperparameters and the presence of some random noise in the error measure (which is larger in small sample sizes), it becomes a near certainty that the calculated accuracy overestimates the generalization performance. In contrast, methods that do not directly minimize the error but instead model a loss surface across different models are likely to exhibit less bias even when the calculation method uses the same data to tune and assess the model. Towards this end, we propose using optimal design of experiment (DoE) methodology to select hyperparameter levels to be tested, fit a surface to the hyperparameter space, then select the hyperparameters in the space that the surface indicates would most reduce model loss.

Utilizing DoE methods for hyperparameter tuning has gained momentum in recent years, seeing applications in cancer detection, food production, and across models such as XGBoost, artificial neural networks, and SVMs (Vasquez-Ramos et al., 2025; Pannakkong et al., 2022; Ayoubi et al., 2024; Bozkurt Keser & Buruk Sahin, 2021). The general method applied by many of these papers follows the work of Lujan-Moreno et al. (2018), which uses a screening experiment, followed by a Box-Behnken design and ridge analysis to determine optimal hyperparameters for a random forest. A separate approach has shown that using a kriging model on space-filling designs is superior to factorial designs (Shi et al., 2023), which augments the work of Johnson et al. (2010), who determine that space-filling designs are useful for deterministic models and high-order polynomial models due to the lack of replication. With regards to small sample analysis, Aftab et al. (2025) utilized response surface methodology with LASSO regression to reduce overfitting on small samples.

Although there have been a number of prior studies investigating the efficacy of DoE for hyperparameter tuning, this paper is the first to specifically examine its ability to combat overoptimistic error assessments in small sample machine learning. Our proposed approach uses a relatively standard space-filling design to obtain model losses across the hyperparameter space. Then, a second-order linear model with cubic terms is used to fit a surface to the hyperparameter space. Once this polynomial surface has been fit, the minimum of the surface in the hyperparameter space can be found easily using gradient-based optimizers. Thus, this approach can be arbitrarily scaled for the number of hyperparameters and the number of design points. We then compare this approach to two common tuning methods: grid search and Bayesian optimization. To effectively measure both generalization error and assessment bias of the different approaches, we select 14 different openML datasets containing a minimum of 5000 samples. By selecting these (relatively) larger datasets, and artificially restricting the amount of data available for training/validation (≤ 200 samples), we can simulate the challenges of small sample ML problems while maintaining a more accurate measure of generalization error that is not typically available. Using this framework, we compare the efficacy of different tuning methods when applied to support vector machines and gradient boosting machines and embed the process in a 100 iteration Monte Carlo simulation to accurately gauge the average performance and bias of the different tuning methods. Our findings show that the proposed DoE approach achieves comparable balanced accuracy to existing hyperparameter tuning strategies and exhibits close to zero bias even in very small samples ($n = 10$).

2 Methods

2.1 Hyperparameter Optimization in Supervised Learning

Consider a dataset $(\mathbf{x}_1, y_1), ..., (\mathbf{x}_\ell, y_\ell)$ where $\mathbf{x}_i$ is a d-dimensional feature vector and y is a target variable, which together are sampled from the joint distribution $P_{\mathbf{X},Y}(\mathbf{x}, y)$. The general goal of a machine learning algorithm is to learn a function $\hat{f}(\mathbf{X}) \rightarrow Y$ that accurately maps the feature vector to the target variable. In order to select the most suitable mapping for a given problem, it is typical to define some loss function

$L(f(\mathbf{x}), y)$ that measures the difference between the model prediction and the true value, at which point we would like to select the model that minimizes the risk produced by this loss function across the joint distribution, defining the risk as

$$R(f) = E_{\mathbf{X},Y}\left[L(f(\mathbf{X}), Y)\right].$$

In practical scenarios, however, the joint distribution is unknown. Thus, instead it is common to estimate the risk empirically by quantifying the sample mean of the loss function across the available data

$$\hat{R}(f) = \frac{1}{n}\sum_{i=1}^{n} L(f(\mathbf{x}_i), y_i).$$

This process of attempting to minimize the risk of a model by minimizing the empirical risk across the sample data is commonly referred to as empirical risk minimization (ERM). In the context of HPO, we may redefine our mapping function as $\hat{f}_\theta(\mathbf{X})$, where θ is a K-dimensional hyperparameter configuration. To select the hyperparameter configuration that minimizes the empirical risk,

$$\theta^* = \arg\min_{\theta\in\Theta} \hat{R}(f_\theta)$$

where $\Theta = (\theta_1, \theta_2, ..., \theta_M)$ denotes the set of M sample hyperparameter configurations we are choosing between. Here Θ may define a grid or random selection of sample hyperparameter within a given range of values. In practice, $\hat{R}(f_\theta)$ is typically estimated using a dataset separate from that used to train the model via either cross-validation or fixed holdout sets to more accurately estimate generalization error and prevent biasing the selection of θ^* towards higher complexity models (Bischl et al., 2023). In the case of cross-validation (CV), we define the empirical risk as

$$\hat{R}_{CV}(f_\theta) = \frac{1}{n}\sum_{j=1}^{J}\sum_{i\in C_j} L\left(y_i, \hat{f}_\theta^{-j}(x_i)\right).$$

A fundamental idea behind ERM is that, for large ℓ, the empirical risk will approximate the true risk of a given model. As a result of this, $\hat{R}_{CV}(f_\theta)$ and other holdout estimates serve as reasonable proxies for $R(f_\theta)$, and the θ that minimizes $\hat{R}_{CV}(f_\theta)$ (denoted as $\hat{\theta}$) is assumed to also minimize $R(f_\theta)$ (Bischl et al., 2023). Here, PAC learning theory even provides guarantees on the maximum difference between generalizable test error and training error given a representative sample (Valiant, 1984).

However, in many places where this cross-validation framework is employed, sample sizes can be relatively small ($\ell < 100$). For smaller samples, the CV error provides an *almost* unbiased estimator of the true generalization error *only* in cases where all stages of the learning process (training, hyperparameter tuning, etc.) are repeated in each cross-validation loop (Varma & Simon, 2006). Varma & Simon (2006) further show that if hyperparameter tuning is done using cross-validation and then the generalizable error is estimated via cross-validation, high bias in the generalizable test error can be observed. Furthermore, Tibshirani & Tibshirani (2009) note that when hyperparameter tuning occurs outside of the loop, $\hat{R}_{CV}(f_{\hat{\theta}})$ is overly optimistic specifically because $\hat{\theta}$ was chosen to minimize it. There have been some prior efforts to address this bias. Efron (2009) proposed a debiasing strategy using empirical Bayes estimates, but do not use cross-validation to tune their model. Tibshirani & Tibshirani (2009) introduced a general debiasing formula for cross-validation accuracies affected by tuning bias. However, empirically their solution was found to trade an overly optimistic bias for an overly pessimistic one. Tsamardinos et al. (2018) implemented a residual bootstrapping strategy to debias CV estimates that performed remarkably well with decreased computational burden compared to nested cross-validation. While nested cross-validation and the bootstrapping approach from Tsamardinos et al. (2018) remain the most reliable solutions for unbiased error estimation in tuned models, surveys of the literature indicate that these strategies are rarely employed in scientific research, and overly optimistic performance estimates remain pervasive in small sample studies (Berisha et al., 2022; Vabalas et al., 2019).

As a final note, it is worth pointing out that in contrast to all of the evidence of optimistic bias in CV estimates, these estimates should actually exhibit a pessimistic bias when the risk is truly excluded from

the model tuning / selection process. Kohavi et al. (1995) investigated the effect of cross-validation using a model on real datasets that was not tuned and showed that k-fold cross-validation error had a positive bias. This bias is the expected result of holdout methods broadly. Since machine learning algorithms are expected to improve with the size of the training set, any holdout method that artificially reduces the amount of training data will induce a positive bias in the risk measure. Thus, in any case where a negative bias in the CV empirical risk can be measured, we can assume it has already overcome the competing positive holdout bias inherent in the CV estimate. As a consequence of this, methods that utilize a greater portion of the data for training (such as leave-one-out CV) and exhibit reduced holdout bias are expected to be more susceptible to overoptimism as the reduced holdout bias will actually yield a greater bias overall.

2.2 Design of Experiments

When applied to hyperparameter optimization, designing an experiment is the process of selecting which hyperparameter levels should be investigated. This selection process is often undertaken via the optimal design process, where the hyperparameter levels are chosen such that some criterion is optimized (Goos & Jones, 2011). We use the optimal DoE framework as follows. Let K be the number of hyperparameters of interest. Additionally, let N be the number of desired hyperparameter combinations to be tested. Define $\mathbf{X} \in \mathcal{X} \subseteq \mathbb{R}^{N \times K}$ to be the design matrix composed of N hyperparameter combination row vectors $\mathbf{x}_i$. It is common practice in DoE to standardize the factor space to $[-1, 1]$ such that each $\mathbf{x}_i \in [-1, 1]^K$ and $\mathcal{X} = [-1, 1]^{N \times K}$; we will follow this practice.

The values inside of the design matrix are found using the solution to an optimization problem. In this case, because training a model on the same data multiple times is often nonsensical due to the deterministic nature of the training regime of many ML models, replicate points are undesirable. Thus, we use a space-filling design to ensure that there are no replicate points. As shown by Johnson et al. (2010), the sphere-packing method for space-filling has the best average prediction variance compared to other space-filling strategies. This is desirable, as we wish to predict which portion of the hyperparameter space would best fit the data, so having a low prediction variance would benefit finding that hyperparameter combination. We formulate the sphere-packing method as

$$\mathbf{X}^* := \arg\max_{\mathbf{X} \in \mathcal{X}} \min_{i<j} d(\mathbf{x}_i, \mathbf{x}_j),$$

where d is a function that finds the pairwise Euclidean distance between points. Thus, the sphere-packing method finds the matrix $\mathbf{X}$ such that the minimum pairwise distance between each hyperparameter combination $\mathbf{x}_i$ is maximized.

Once the design matrix has been generated, the ML model of choice can be fit to the data using the hyperparameter combinations contained in $\mathbf{X}$, then a meta-model can be fit to the hyperparameter surface. By fitting the ML model to the data for each hyperparameter combination in $\mathbf{X}$, a vector of length N is generated via the calculation method, defined here as $\mathbf{y}$. In order to fit a second-order linear model with cubic terms, we utilize the model matrix $\mathbf{F} \in \mathbb{R}^{N \times P}$, where $\mathbf{F}$ is comprised of N row vectors

$$\mathbf{f}(\mathbf{x}_i) = \begin{bmatrix} 1 & x_{i1} & \dots & x_{iK} & x_{i1}x_{i2} & \dots & x_{i(K-1)}x_{iK} & x_{i1}^2 & \dots & x_{iK}^2 & x_{i1}^3 & \dots & x_{iK}^3 \end{bmatrix}.$$

Utilizing the model matrix $\mathbf{F}$, define a linear model such that

$$\mathbf{y} = \mathbf{F}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{1}$$

where $\boldsymbol{\beta}$ is a vector of linear parameters and $\epsilon \sim \mathcal{N}_N(0, \sigma^2 I)$. Then by ordinary least squares, the parameters can be estimated as $\hat{\boldsymbol{\beta}} = (\mathbf{F}^\intercal \mathbf{F})^{-1} \mathbf{F}^\intercal \mathbf{y}$. Thus, we have a linear model of the response surface for the hyperparameter space.

In order to fit the response surface, we use mean squared error (MSE) as a surrogate loss to estimate the balanced accuracy (BA). Due to the low number of samples, BA is noticeably discretized, which results in suboptimal response surfaces. While this discretization is not a problem for GS, we utilize the MSE between the probability associated with the positive label generated by the ML model and the true label (0 or 1) as a surrogate loss for the DoE approach in order to obtain a smoother response surface. Via this method,

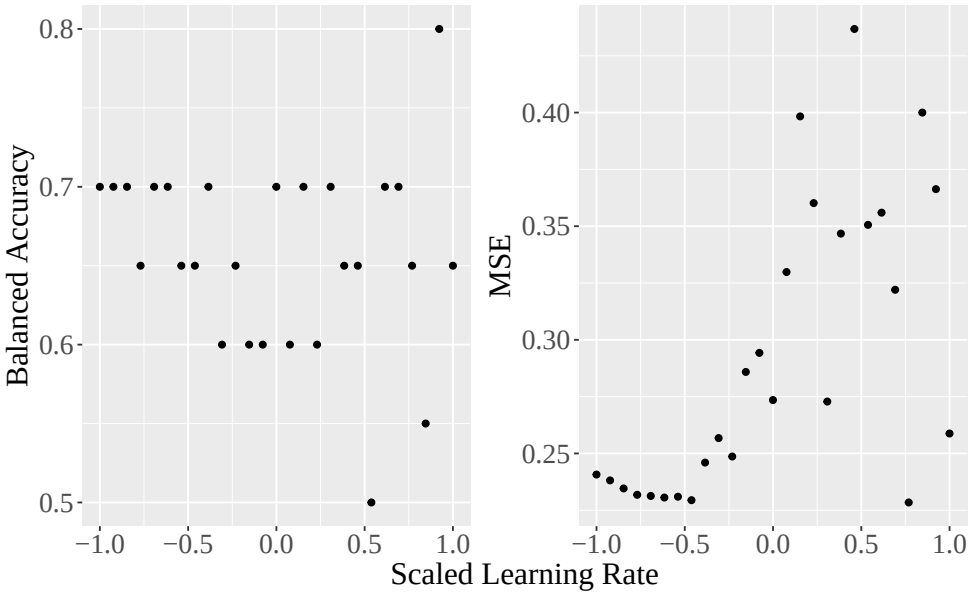


Figure 1: Plot of the loss associated with BA and MSE for learning rate

we obtain a surface that is noticeably less discrete and more suited to fitting a linear model, as reflected by Figure 1.

We note here that the standard approach to hyperparameter tuning via DoE utilizes a quadratic model (Lujan-Moreno et al., 2018; Bozkurt Keser & Buruk Sahin, 2021; Pannakkong et al., 2022). However, in initial simulations using DoE to optimize the learning rate for a gradient boosting classifier on the Japanese Vowel dataset described in Section 2.4, we found the quadratic model to be insufficient, sometimes resulting in a concave function that did not reflect the trends found in the data. From this, we determined that the quadratic model was underfitting the data being obtained, which often results in sub-optimal hyperparameter combinations. Thus, we utilize cubic terms to more accurately identify local trends in the hyperparameter tuning results, as shown in Figure 2. This approach seems to appropriately fit the data, so there should not be bias in the predictions of MSE. In this way, we create a less biased estimator of the surrogate loss.

In order to find the hyperparameter combination that gives the expected minimum surrogate loss, we utilize constrained optimization. Specifically, we use the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (a.k.a L-BFGS) algorithm implementation in the SciPy package in the Python coding language (Liu & Nocedal, 1989; Virtanen et al., 2020) to find the minimum described by the linear model outlined in Equation 1. This optimization is bounded by the scaled hyperparameter space $[-1, 1]^K$. Thus, we obtain a hyperparameter combination that may not exist in the initial experiment that is determined from a smooth function rather than a discrete set of points.

2.3 Simulation Setup

In order to ascertain the bias of the DoE approach, we utilize 100 Monte Carlo (MC) simulations on real datasets and compare DoE to GS and Bayesian Optimization (BO) using the support vector classifier (SVM) and gradient boosting classifier (GBM) implementations in scikit-learn (Pedregosa et al., 2011). For the hyperparameter optimization, we utilize the GS method in scikit-learn and the BO implementation in scikit-optimize (Pedregosa et al., 2011; Head et al., 2018). We begin each simulation with a dataset comprised of $n_{\text{full}} >> 0$ instances. Let $n_{\text{train}} << n_{\text{full}}$ be the number of samples to be included in the training set. We use five levels for n_{train}: 10, 20, 50, 100, and 200 samples. Then for each MC simulation, a balanced subset of the original dataset of size n_{train} is randomly selected from n_{full} rows such that $\frac{n_{\text{train}}}{2}$ samples are from each binary class, which reflects the desired class balance present in many real world experiments.

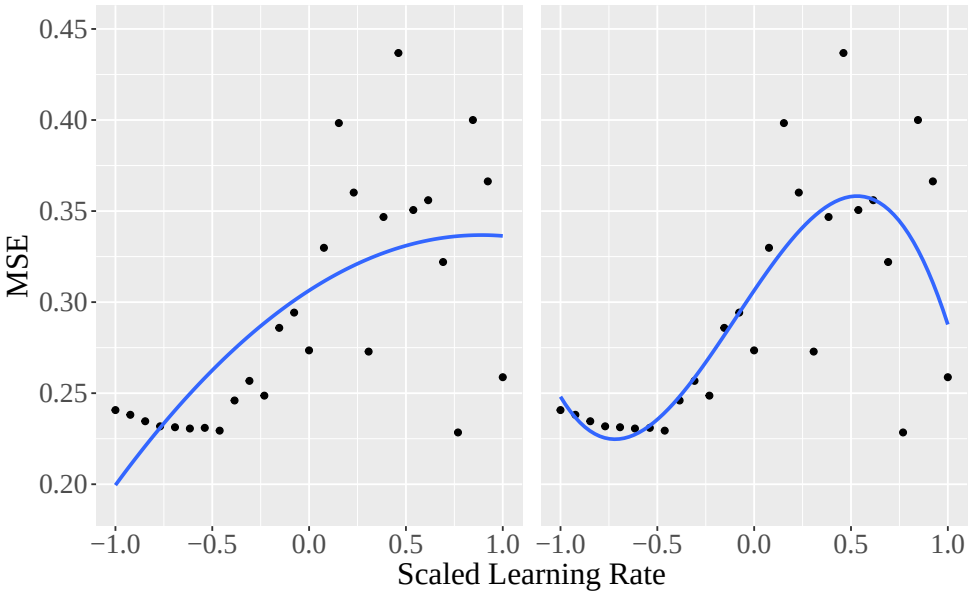


Figure 2: Plot of the fitted models associated with MSE for learning rate

Hyperparameter	Lower Bound	Upper Bound	Log Scale	Rounded
Cost	2^{-5}	2^{15}	True	False
Gamma	2^{-15}	2^{3}	True	False

Table 1: Hyperparameter ranges and attributes for the SVM

Once the sample has been taken, 5-fold cross-validation is used on each hyperparameter combination given by DoE, GS, and BO. Then, the model is trained on the training data using the optimal hyperparameter combinations determined by each method. Once the model has been trained, we measure the generalizable BA of the model using the remaining $n_{\text{test}} = n_{\text{full}} - n_{\text{train}}$ samples, similar to the method used by Beleites et al. (2005), and measure the relative performance and bias of these approaches via their BA. As the test error is essentially treated as a ground truth measure of generalization error in this analysis, datasets are selected to ensure that the test set is very large relative to the training set ($n_{\text{test}} > 4000$).

Before using the GS, BO, and DoE methods, we select a range of hyperparameter values that would be useful for each classification method. For the SVM, we use ranges for the gamma and cost hyperparameters. The exact range values are found in Table 1, with the ranges drawn from Probst et al. (2019). Note that both hyperparameters are on a log scale when determining the values to be used by GS or BO and when defining the scaled space utilized by DoE. With regards to the GBM, we utilize hyperparameter ranges for learning rate, the number of trees, and the maximum tree depth, found in Table 2. The number of trees and the maximum tree depth are numeric and discrete, and the learning rate is continuous and log-scaled, consistent with standard practice (Probst et al., 2019). Thus, after creating the experimental design, we project to the discrete space for tree depth and the number of trees via rounding after unscaling the experimental design from $[-1, 1]$ to the actual hyperparameter ranges. While it is likely that some optimality is lost in this rounding step, this loss is minimal given sufficient grid density. In order to ensure that we span the space well, we use 5^K combinations of hyperparameters for the GS and space-filling designs, as well as the number of iterations used in BO. BO is performed sequentially with a Gaussian Process surrogate, scoring on BA. We also use the default parameters in scikit-learn as a baseline, cross-validating the error as described above.

Hyperparameter	Lower Bound	Upper Bound	Log Scale	Rounded
Number of Trees	100	1000	False	True
Learning Rate	2^{-10}	1	True	False
Maximum Depth	1	5	False	True

Table 2: Hyperparameter ranges and attributes for the GBM

Dataset	ID	Total Instances	Minority Class Size	Number of Features
Pol	722	15000	5041	48
Ailerons	734	13750	5828	40
Small CPU	735	8192	2477	12
Puma	752	8192	4064	32
Computer Activity	761	8192	2477	21
House	821	22784	6744	16
Bank	833	8192	2543	32
Elevators	846	16599	5130	18
Wind	847	6574	3073	14
Japanese Vowels	976	9961	1614	14
Letter	977	20000	813	16
Waveform-5000	979	5000	1692	40
Pendigits	1019	10992	1144	16
Census	44115	5188	2594	20

Table 3: Description of datasets used in the MC simulation study

2.4 Datasets

In this work, we use open source datasets provided by the Open ML project (Vanschoren et al., 2014). Specifically, we use 14 datasets identified in the OpenML Python package by the ID's 722, 734, 735, 752, 761, 821, 833, 846, 847, 976, 977, 979, 1019, and 44115 (Feurer et al., 2021). These datasets were chosen from results of a filter on the OpenML datasets that ensured no missing data, a minority class size of over 500, less than 100,000 instances, more than 10 but fewer than 51 features, binary classification tasks, and fewer than 10 categorical variables. Minimum requirements specified in this filter are to ensure datasets provide a reasonable proxy for problems typically observed in these settings as well as to guarantee a reliable estimate of generalization error. Maximum requirements are imposed to limit the computational burden of any individual dataset. The first 13 datasets were chosen as the first unique dataset results from the filter, while the last (Census) was the only dataset in the filter that had the same label identifier as the first 13. Because each dataset comes from disparate applications, we believe this collection to be a representative sample of tabular datasets such that computation and the overall workflow were suitable for so many MC simulations. The dataset characteristics can be found in Table 3.

2.5 Evaluation Metrics

To evaluate the performance of each approach for hyperparameter tuning (including the untuned baseline model), we use two primary performance measures. The first measure is the expected BA (EBA) across the held out test set, calculated by

$$EBA_{test} = \frac{1}{100}\sum_{i=1}^{100} BA_{\text{test}}^{(i)}$$

where $BA_{\text{test}}^{(i)}$ represents the measured test error for a given method at the i^{th} iteration of the Monte Carlo simulation. As this analysis is restricted to data with sufficient samples for a large held out test set (≥ 4000 samples), this test error provides a more accurate estimate of generalization error than is possible in real

small sample studies. The second measure we use is assessment bias, defined as

$$Bias = \frac{1}{100}\left(\sum_{i=1}^{100} BA_{\text{train}}^{(i)}\right) - EBA_{test}.$$

This measure represents the MC average of the discrepancy between the train and test performance. Together, these measures provide a relatively comprehensive assessment of 1) the true generalization error achieved by a given tuning method and 2) how biased non-nested error assessments of that tuning strategy will be.

3 Results

The results for each method are compiled in terms of both generalization (test) balanced accuracy and assessment bias, and are organized as follows. Results for the SVM analysis are presented first in Section 3.1 followed by the results for the GBM experiments in Section 3.2. In each section, the balanced accuracy and bias are averaged across the 100 MC iterations and 14 datasets and plotted against sample size for each method of hyperparameter tuning. To examine performance across the different datasets, tables presenting the results for the smallest ($n = 10$) and largest ($n = 200$) sample sizes across the individual datasets are also provided. The aggregate rankings of each method across the fourteen datasets (in both EBA and bias) will serve as the primary measure for comparing the different methods explored in this paper.

3.1 Support Vector Machine Results

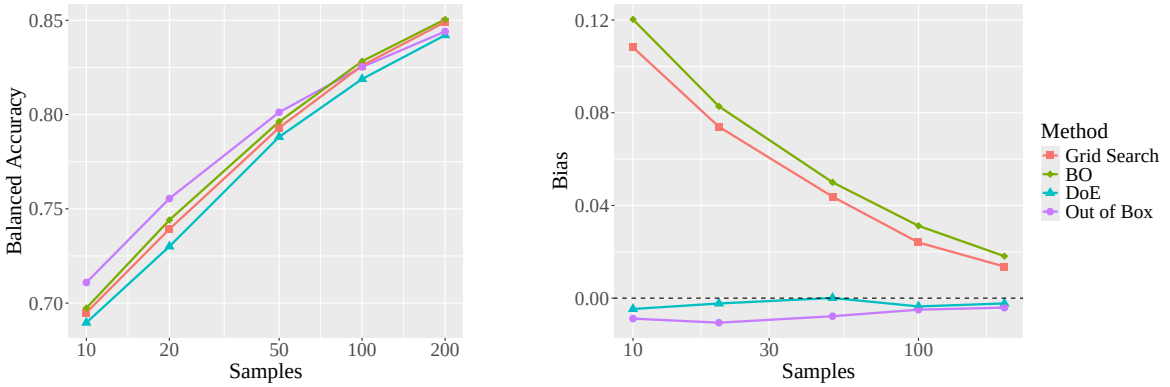
Figure 3a presents the aggregate balanced accuracy across the fourteen datasets and 100 MC iterations for the SVM. As expected, these results show a consistent positive relationship between sample size and accuracy. Comparing the different methods, we find the untuned model achieved the best performance for the smaller samples ($n = 10, 25, 50$) and the Bayesian optimization approach achieved the highest performance at the larger sample sizes. The relative deterioration in performance of the out-of-box approach at larger sample sizes is an expected consequence of increased reliability of tuning methods as the size of the validation set expands. With regards to EBA, the proposed DoE approach achieved the lowest accuracy across all samples sizes, slightly under performing the grid search method. While difference in performances across tuning methods (excluding the default) appear consistent across sample sizes, differences in performance are relatively small and all methods perform within 2.5% of any other method for a given sample size in the aggregate.

Figure 3b displays the bias results as a function of sample size. Here, we see a clear distinction between the bias exhibited by the grid search and Bayesian optimization methods and that exhibited by the untuned model and the DoE based tuning method. Both GS and BO exhibit biases of more than 10% at the smallest sample size ($n = 10$), which reduces with sample size, falling under 2% for the $n = 200$ case. Both DoE and the out-of-box approach exhibit a small negative bias that is relatively consistent across sample size. This bias is likely reflecting the inherent positive bias in CV risk, described in Kohavi et al. (1995), which is caused by the reduced size of the training set. Based on this, we would expect this bias to shrink with n as well, however such an effect is not obvious from these experiments.

More detailed results for how each method performed at $n = 10, 200$ are provided in Tables 4 and 5, with Nemenyi critical distance plots corresponding to those tables provided in Figure 4. From the Nemenyi plot in Figure 4a, the only method that does not significantly underperform the baseline in terms of EBA for $n = 10$ is BO. However, as shown in Figure 4b, no method is significantly better than another for $n = 200$. Figure 4c shows that, while the bias for DoE and the baseline are not significantly different, they are both significantly better than the other two methods for $n = 10$. In Figure 4d, GS shows an improvement such that it is no longer significantly worse than the DoE and baseline methods in terms of bias for $n = 200$.

3.2 Gradient Boosting Results

Figure 5a shows the aggregate balanced accuracy across the fourteen datasets and 100 MC iterations for the GBM. Similar to the results in Section 3.1, there is a consistent positive relationship between EBA



(a) Aggregated EBA

(b) Aggregated Bias

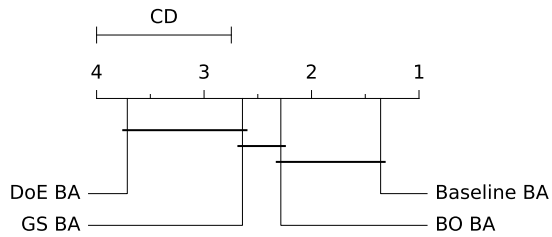
Figure 3: Aggregated results for the SVM across all 14 datasets

	Samples = 10				Samples = 200			
Dataset	Baseline EBA	GS EBA	BO EBA	DoE EBA	Baseline EBA	GS EBA	BO EBA	DoE BA
722	**68.11 (1)**	64.76 (3)	65.93 (2)	64.62 (4)	89.21 (4)	90.60 (2)	90.58 (3)	**90.60 (1)**
734	**66.27 (1)**	65.20 (3)	65.32 (2)	65.14 (4)	83.23 (4)	**84.93 (1)**	84.86 (2)	83.31 (3)
735	78.81 (3)	**79.61 (1)**	78.83 (2)	77.87 (4)	**89.08 (1)**	88.27 (4)	88.46 (2)	88.38 (3)
752	**51.80 (1)**	51.38 (4)	51.54 (2)	51.49 (3)	**58.01 (1)**	57.24 (2)	57.14 (3)	55.30 (4)
761	81.31 (2)	**81.49 (1)**	80.39 (3)	78.95 (4)	**90.31 (1)**	89.57 (2)	89.54 (3)	88.84 (4)
821	63.69 (2)	63.43 (3)	**63.73 (1)**	62.43 (4)	80.35 (3)	**80.50 (1)**	80.41 (2)	78.43 (4)
833	**56.94 (1)**	55.64 (2)	55.51 (3)	55.34 (4)	73.74 (4)	**74.76 (1)**	74.60 (2)	74.26 (3)
846	63.95 (2)	**64.25 (1)**	62.84 (3)	61.29 (4)	75.73 (4)	83.93 (3)	**84.44 (1)**	84.21 (2)
847	**75.81 (1)**	73.78 (3)	74.07 (2)	73.74 (4)	**83.51 (1)**	83.11 (3)	83.17 (2)	82.32 (4)
976	**80.47 (1)**	79.22 (4)	79.94 (3)	80.47 (2)	**96.65 (1)**	95.50 (3)	96.01 (2)	94.89 (4)
977	**73.65 (1)**	70.77 (3)	71.14 (2)	70.23 (4)	95.26 (2)	94.71 (4)	**95.75 (1)**	94.90 (3)
979	**70.57 (1)**	68.16 (3)	68.19 (2)	67.27 (4)	**85.53 (1)**	84.93 (2)	84.77 (3)	82.58 (4)
1019	**92.82 (1)**	85.46 (4)	90.24 (2)	88.77 (3)	98.17 (4)	98.20 (3)	**98.54 (1)**	98.50 (2)
44115	**71.09 (1)**	69.72 (2)	68.64 (3)	67.85 (4)	**82.89 (1)**	82.45 (2)	82.38 (4)	82.42 (3)
Average	**71.09 (1.36)**	**69.49 (2.64)**	**69.74 (2.29)**	**68.96 (3.71)**	**84.41 (2.29)**	**84.91 (2.36)**	**85.05 (2.21)**	**84.21 (3.14)**

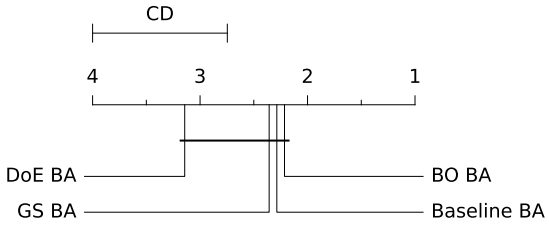
Table 4: Dataset test performance reported as (% accuracy) for 10 and 200 samples for the SVM

	Samples = 10				Samples = 200			
Dataset	Baseline Bias	GS Bias	BO Bias	DoE Bias	Baseline Bias	GS Bias	BO Bias	DoE Bias
722	**-2.11 (1)**	9.84 (3)	12.77 (4)	-6.82 (2)	-0.72 (3)	**0.36 (1)**	1.75 (4)	-0.46 (2)
734	1.63 (2)	15.30 (3)	15.38 (4)	**-0.24 (1)**	-1.02 (3)	0.92 (2)	1.74 (4)	**-0.30 (1)**
735	2.09 (2)	7.99 (3)	10.57 (4)	**1.23 (1)**	-0.08 (2)	1.88 (3)	1.93 (4)	**-0.06 (1)**
752	**-0.70 (1)**	11.42 (3)	13.76 (4)	-1.59 (2)	**0.80 (1)**	3.56 (3)	4.11 (4)	1.11 (2)
761	**-0.01 (1)**	6.41 (3)	9.01 (4)	-0.35 (2)	0.05 (2)	1.84 (3)	2.17 (4)	**-0.01 (1)**
821	**-0.69 (1)**	15.97 (3)	16.37 (4)	1.97 (2)	-0.84 (2)	1.51 (3)	2.14 (4)	**-0.32 (1)**
833	-0.94 (2)	12.86 (3)	15.69 (4)	**-0.24 (1)**	-0.65 (2)	1.82 (3)	2.09 (4)	**-0.35 (1)**
846	-2.65 (2)	11.05 (3)	12.66 (4)	**-0.99 (1)**	**-0.19 (1)**	1.15 (3)	1.72 (4)	0.36 (2)
847	-0.71 (2)	12.22 (4)	11.53 (3)	**-0.54 (1)**	-0.79 (2)	1.61 (3)	1.91 (4)	**-0.28 (1)**
976	-4.37 (2)	7.98 (3)	8.66 (4)	**0.23 (1)**	**-0.30 (1)**	0.76 (3)	1.07 (4)	-0.33 (2)
977	-2.55 (2)	11.33 (3)	12.26 (4)	**-1.73 (1)**	-1.03 (4)	**0.35 (1)**	0.67 (2)	-0.67 (3)
979	**2.93 (1)**	12.64 (3)	14.51 (4)	5.03 (2)	**-0.34 (1)**	1.28 (2)	1.64 (3)	-1.77 (4)
1019	**-0.52 (1)**	6.54 (4)	4.96 (3)	-2.67 (2)	**-0.05 (1)**	0.63 (3)	0.64 (4)	-0.41 (2)
44115	-3.79 (2)	9.98 (3)	10.26 (4)	**0.15 (1)**	-0.57 (2)	1.43 (3)	1.76 (4)	**0.28 (1)**
Average	**-0.88 (1.57)**	**10.82 (3.14)**	**12.03 (3.86)**	**-0.47 (1.43)**	**-0.41 (1.93)**	**1.36 (2.57)**	**1.81 (3.79)**	**-0.23 (1.71)**

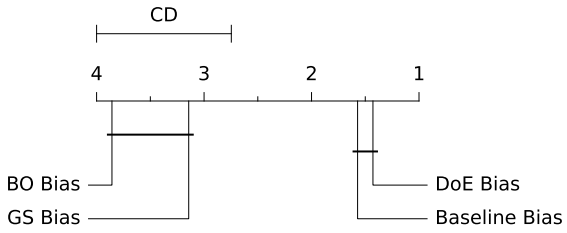
Table 5: Dataset bias for 10 and 200 samples for the SVM. Values have been scaled by a factor of 10^2



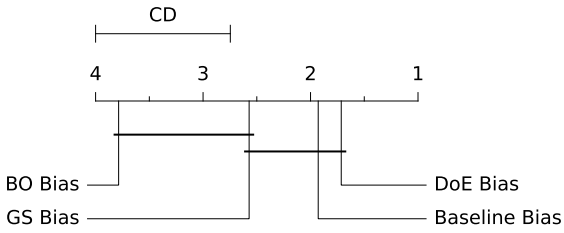
(a) 10 Samples (EBA)



(b) 200 Samples (EBA)

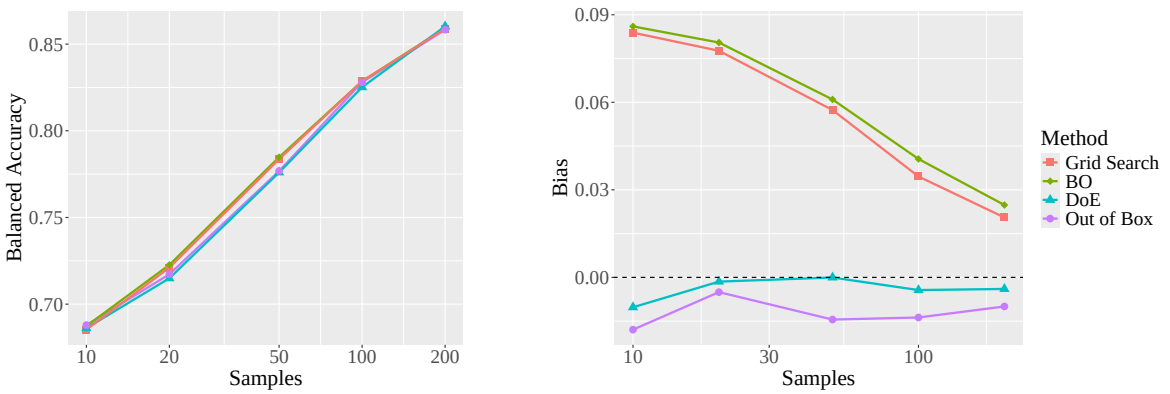


(c) 10 Samples (Bias)



(d) 200 Samples (Bias)

Figure 4: Critical Distance plots of the average rankings of EBA and bias from Tables 4 and 5 for the SVM



(a) Aggregated EBA

(b) Aggregated Bias

Figure 5: Aggregated results for the GBM across all 14 datasets

and sample size. For $n = 20, 50$, it appears that GS and BO outperform the baseline and DoE methods, though never by more than 2.5%. In every other sample size, the methods appear to be almost equivalent in predictive performance. Contrasting the results in Section 3.1, we do not observe the relative deterioration in performance for the baseline model as sample sizes increase.

Figure 5b presents the average bias of each method across each dataset and sample size. Once again, we observe a clear difference between GS and BO compared to DoE and the baseline. GS and BO have biases of close to 8% at $n = 10$, decreasing steadily to about 2% at $n = 200$. The DoE and baseline methods show a small negative bias that is relatively steady across sample sizes. This negative bias again does not clearly show decreased bias for those methods as sample size increases.

More detailed results for how each method performed at n = 10, 200 are provided in Tables 6 and 7, with Nemenyi critical distance plots corresponding to those tables provided in Figures 6. From Figure 6a, it appears that no method is significantly better than any other in terms of EBA for $n = 10$, although from Figure 6b it appears that the DoE method is significantly better than GS for $n = 200$. There were no other significant differences at that sample size. However, as shown in Figure 5*a*, this difference is not more than 2.5%. In Figures 6c and 6d, we observe similar results to Section 3.1. Principally, at $n = 10$, the bias between DoE and the baseline are not significantly different from each other, but are both significantly better than the bias observed in the BO and GS methods. For $n = 200$, DoE is significantly better than the GS and BO methods, but not the baseline. Additionally, the baseline is not significantly different from the GS at this sample size.

4 Discussion

This paper examines the ability of a hyperparameter optimization approach based on response surface methodology to produce less biased performance assessments in small sample ML experiments. We examined the viability of this approach across fourteen publicly available datasets and compared its performance in terms of the expected generalization performance (EBA) and the assessment bias. Results show that the proposed approach mitigates more than 90% of the bias in smaller sample sizes, while maintaining similar generalization performance (slightly higher in the GBM experiments and lower in the SVM experiments). These results provide strong initial support for the viability of this HPO approach in small sample ML experiments. However, there are several phenomena of which we desire to make further mention.

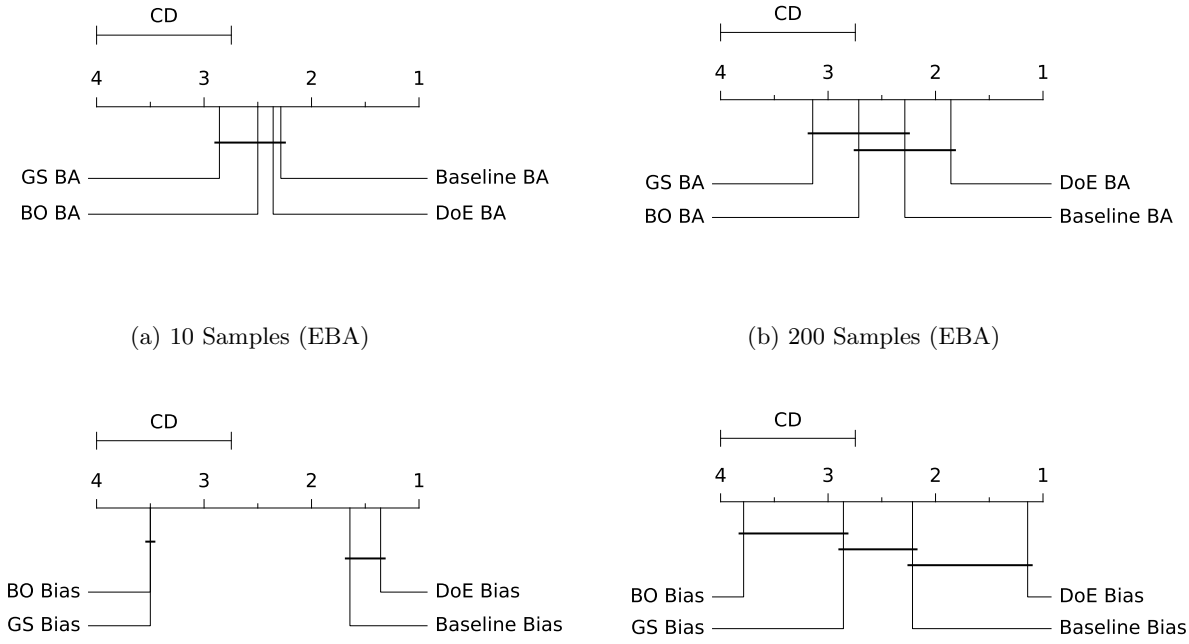
The first question to address is why the DoE methodology so effectively mitigates the bias in the cross validation assessment. This comes as a result from the fact that the optimum found using the DoE methodology identifies the expected optimum, not the sampled optimum. Contrast this approach with GS and BO, which both return the best performing hyperparameter combination they encounter via cross-validation. Whereas noise in the error estimate manifests as residuals in the linear model and has a diffuse effect on the resulting

	Samples = 10				Samples = 200			
Dataset	Baseline EBA	GS EBA	BO EBA	DoE EBA	Baseline EBA	GS EBA	BO EBA	DoE EBA
722	64.06 (2)	**64.18 (1)**	63.46 (3)	62.02 (4)	**92.10 (1)**	91.46 (3)	91.43 (4)	91.80 (2)
734	67.68 (2)	67.60 (3)	**68.30 (1)**	67.59 (4)	83.50 (2)	83.04 (4)	83.14 (3)	**83.71 (1)**
735	74.04 (2)	73.92 (3)	**74.54 (1)**	73.67 (4)	87.94 (2)	87.82 (3)	87.79 (4)	**88.03 (1)**
752	51.15 (2)	51.01 (4)	51.10 (3)	**51.15 (1)**	81.62 (4)	**84.66 (1)**	84.48 (2)	84.18 (3)
761	74.59 (3)	74.11 (4)	74.68 (2)	**74.79 (1)**	89.25 (2)	89.13 (4)	89.18 (3)	**89.37 (1)**
821	**66.78 (1)**	65.91 (4)	66.41 (3)	66.44 (2)	**82.62 (1)**	82.50 (4)	82.52 (3)	82.61 (2)
833	55.19 (4)	55.66 (2)	55.22 (3)	**55.69 (1)**	**74.32 (1)**	73.67 (3)	73.86 (2)	73.46 (4)
846	**62.12 (1)**	60.62 (4)	61.51 (3)	61.56 (2)	**73.60 (1)**	73.47 (2)	73.44 (3)	72.73 (4)
847	75.30 (3)	**75.72 (1)**	75.51 (2)	75.11 (4)	82.83 (2)	82.53 (4)	82.60 (3)	**83.25 (1)**
976	78.32 (3)	78.49 (2)	78.04 (4)	**78.53 (1)**	93.81 (4)	93.96 (3)	94.00 (2)	**94.11 (1)**
977	**71.74 (1)**	70.77 (4)	71.63 (2)	71.43 (3)	94.53 (2)	94.30 (4)	94.34 (3)	**94.53 (1)**
979	65.99 (4)	**66.35 (1)**	66.01 (3)	66.26 (2)	83.44 (2)	83.24 (4)	**83.46 (1)**	83.41 (3)
1019	81.44 (2)	80.55 (4)	**81.70 (1)**	81.13 (3)	96.54 (4)	96.67 (3)	96.89 (2)	**96.95 (1)**
44115	74.67 (2)	74.51 (3)	74.18 (4)	**75.27 (1)**	85.55 (4)	85.93 (2)	85.73 (3)	**86.18 (1)**
Average	**68.79 (2.29)**	**68.53 (2.86)**	**68.73 (2.50)**	**68.62 (2.36)**	**85.83 (2.29)**	**85.88 (3.14)**	**85.92 (2.71)**	**86.02 (1.86)**

Table 6: Dataset test performance reported as (% accuracy) for 10 and 200 samples for the GBM

	Samples = 10				Samples = 200			
Dataset	Baseline Bias	GS Bias	BO Bias	DoE Bias	Baseline Bias	GS Bias	BO Bias	DoE Bias
722	-2.76 (2)	8.22 (3)	11.44 (4)	**0.48 (1)**	-1.09 (2)	1.47 (3)	1.66 (4)	**-0.86 (1)**
734	-2.58 (2)	6.10 (3)	7.40 (4)	**-0.99 (1)**	-0.53 (2)	2.70 (3)	2.84 (4)	**-0.25 (1)**
735	-1.64 (2)	7.98 (4)	6.86 (3)	**0.13 (1)**	-0.25 (2)	2.15 (3)	2.56 (4)	**-0.22 (1)**
752	**-2.45 (1)**	11.49 (3)	13.50 (4)	-3.95 (2)	-2.43 (4)	**0.55 (1)**	1.37 (2)	-1.68 (3)
761	**0.61 (1)**	10.19 (4)	7.72 (3)	1.31 (2)	-0.70 (2)	1.94 (3)	2.32 (4)	**-0.01 (1)**
821	-4.88 (2)	8.19 (4)	6.59 (3)	**-2.04 (1)**	-0.61 (2)	2.72 (3)	3.60 (4)	**-0.24 (1)**
833	-2.29 (2)	11.14 (3)	12.18 (4)	**-0.69 (1)**	-1.19 (2)	3.95 (3)	4.58 (4)	**0.26 (1)**
846	-2.82 (2)	8.28 (3)	10.59 (4)	**-1.46 (1)**	-0.80 (2)	3.44 (3)	4.37 (4)	**0.09 (1)**
847	**-1.50 (1)**	6.18 (4)	5.99 (3)	-1.91 (2)	-1.35 (2)	2.01 (3)	2.46 (4)	**-0.48 (1)**
976	-2.02 (2)	5.91 (3)	6.06 (4)	**-0.93 (1)**	-0.91 (2)	1.12 (3)	1.55 (4)	**-0.33 (1)**
977	**-3.84 (1)**	6.83 (4)	5.17 (3)	-4.03 (2)	-1.32 (3)	1.07 (2)	1.34 (4)	**-1.00 (1)**
979	2.31 (2)	11.95 (3)	13.29 (4)	**0.24 (1)**	-1.54 (2)	2.50 (3)	2.80 (4)	**-0.08 (1)**
1019	-1.14 (2)	6.35 (4)	5.40 (3)	**-0.43 (1)**	-0.64 (2)	0.93 (4)	0.88 (3)	**-0.44 (1)**
44115	**-0.07 (1)**	8.59 (4)	8.22 (3)	-0.07 (2)	-0.60 (2)	2.17 (3)	2.40 (4)	**-0.30 (1)**
Average	**-1.79 (1.64)**	**8.39 (3.50)**	**8.60 (3.50)**	**-1.02 (1.36)**	**-1.00 (2.21)**	**2.05 (2.86)**	**2.48 (3.79)**	**-0.40 (1.14)**

Table 7: Dataset bias for 10 and 200 samples for the GBM. Values have been scaled by a factor of 10^2



(a) 10 Samples (EBA)

(b) 200 Samples (EBA)

(c) 10 Samples (Bias)

(d) 200 Samples (Bias)

Figure 6: Critical Distance plots of average rankings of EBA and bias from Tables 6 and 7 for the GBM

hyperparameter selection, directly selecting a sample optimum (as is done in GS & BO) tunes explicitly to this noise and is highly likely to select hyperparameters exhibiting a positive bias. In larger sample regimes, this noise can generally be assumed to be small relative to the true variation in model performance across different HP tunings, and thus tuning in this way introduces little bias into the resulting assessment. However, in small sample regimes, this practice of selecting the best performing observed hyperparameter set from cross-validation is improper in small sample regimes because it results in high optimistic bias between validation and generalizable error, effectively eradicating the efficacy of k-fold cross-validation as an unbiased estimator of generalizable test error.

While removing the empirical minimum from the selection process explains a reduction in bias, it does not explain why 1) the bias in the DoE method is now negative and 2) why it is lower (on average) than even the untuned classifiers. The negative bias observed in both the DoE and untuned model results from the established fact that k-fold cross-validation results in small negative bias (Kohavi et al., 1995; Austin et al., 2025). This bias is due to the slight reduction in the amount of training data used in each fold and the loss of performance associated with it. Regarding the second question, this negative bias is likely reduced in the DoE method, as this method of tuning is still introducing a small positive bias in the performance assessment which partially offsets the anticipated negative bias. Note that these biases are trivially small relative to those induced by HPO in the GS and BO methods ($< 10\%$).

The second phenomenon we investigate is the relatively good average performance of the baseline hyperparameter set. Where the DoE HPO method yields significantly reduced bias at little cost to generalization error over the GS and BO methods, it offers little benefit in terms of bias reduction or BA over the untuned SVM / GBM models. One of the major challenges with small sample machine learning is that the size of the dataset presents inherent limitations on our ability to accurately measure model performance. So whereas overoptimism can be observed as a general trend across studies through meta-analyses and controlled experiments such as those conducted here, the discrepancy between the reported error and generalization error within any individual study cannot be quantified. With this in mind, the available evidence supporting the efficacy of HPO methods in small sample regimes is actually quite limited and generally based on their success in either 1) large sample regimes or 2) small sample evaluations that are likely to be biased. This presents a strong need to further evaluate the viability of existing HPO strategies in small data regimes and develop new methods of better dealing with the constraints these regimes place on HPO. We can also see in Figures 3 and 5 that the success of the untuned model relative to the different HPO methods declines in the larger sample sizes, as the limitations the sample size restriction imposes on HPO becomes less meaningful. While the success of the untuned model lends some credibility to the default parameterizations included in the package (Pedregosa et al., 2011), it still serves more as an indictment of existing HPO methods for these problems. The default model also appears to behave more inconsistently across datasets than any of the HPO methods. In particular, for dataset 846, the baseline experiences a percentage reduction in predictive performance of almost 10% compared to the next lowest method at $n = 200$, the largest gap observed in any dataset.

The final phenomenon we discuss is the relatively poor performance of BO in these simulations. Although BO has become the gold standard for many current ML applications (Stanton et al., 2022; Wang et al., 2023), the success of BO has been primarily observed in datasets with large sample sizes (Joy et al., 2016; Klein et al., 2017). In smaller sample regimes, prior work has already documented its tendency towards overfitting and producing biased performance estimates (Makarova et al., 2021). There, it is observed that in smaller datasets, low correlation between validation and test error results in inconsistent outcomes for BO-based HPO. This is observable in a dataset containing 2000 observations, indicating that this problem is not strictly limited to very small samples. Overall, these findings suggest that a minimum threshold of data (likely problem specific) must be met to ensure sample loss estimates correlate reliably enough with generalization error for existing HPO methods to serve as a generally preferable alternative over a naive prior hyperparameter specification.

Considering the limitations of this study, it is important to acknowledge that only a small subset of existing HPO methods were considered. Although generally perceived as antiquated in the machine learning literature, grid search appears to be the method of choice in the applied studies for which the small sample bias characteristics have been previously observed (Vabalas et al., 2019; Berisha et al., 2022). As such, GS

serves as a critical baseline for comparison in this analysis. Another limitation of this study is the implicit assumption that the selected mid-sized datasets serve as a reasonable proxy for the smaller datasets observed in the medical classification problems that motivate this research. While there are a number of ways that the datasets explored in this paper differ from those in the referenced meta-analyses (Vabalas et al., 2019; Berisha et al., 2022), the lack of available larger datasets in these regimes is the reason these tasks were subject to meta-analysis in the first place. Further, while there are likely some meaningful differences between the chosen data and data used in actual small sample studies, the consistency of the findings (particularly in the bias) across the diverse set of test cases provides supporting evidence for the idea that these phenomena are not domain-specific. Note also that while we did not explore the role of budget restrictions in our comparison, increasing the budget in these smaller sample regimes would likely only exacerbate overfitting in the GS and BO methods.

As noted in Section 4, the baseline has extremely comparable results to the other three optimization methods across all datasets for sample sizes less than or equal to 50 samples. This indicates that utilizing optimization methods beyond the baseline may not be useful for such small samples in terms of predictive performance. While there appear to be benefits gained from using DoE to offset the bias observed from k-fold cross-validation, DoE does not meaningfully improve the predictive performance for any dataset for very small samples compared to the baseline. Overall, the degree of inconsistency in test error across datasets and models makes it difficult to draw conclusions about the superiority of any method in terms of generalization error for small sample regimes from these results. Thus, future work will include investigating if there are any benefits from utilizing hyperparameter optimization in the small-sample regime.

Finally, the proposed strategy for DoE-based HPO leaves significant room for refinement and additional exploration. Contrasting the results for the GBM and SVM experiments, the DoE approach leaves the widest gap in performance across classifiers of the tested methods. This raises the question of whether misspecification of the linear model could contribute to this inconsistency and whether different hyperparameters and classifiers are more or less conducive to this procedure. Additionally, the choice of surrogate loss is not a perfect proxy for generalizable BA and could thus contribute to this performance gap. Hence, while the DoE method as presented is effective at bias reduction, questions still remain about what changes would need to occur to improve predictive performance. Thus, future work will include investigating methods for improving DoE as an HPO method for small-sample ML.

5 Conclusion

In this study, we used MC simulation applied to fourteen publicly available datasets to show the efficacy of the DoE approach to provide unbiased hyperparameter tuning when limited training data is available. Via these simulations, we show that the DoE method has comparable out-of-sample predictive performance with grid search and Bayesian optimization based tuning methods, while substantially reducing the bias observed in cross-validated accuracy measures. Additionally, our results indicate that, even though the DoE approach reduces absolute bias substantially, it actually has a slight negative bias due to the reduction in available training data from k-fold cross-validation. While these results support the efficacy of the proposed DoE based approach for hyperparameter tuning, comparisons with baseline untuned classifiers show that none of the tested methods of hyperparameter optimizations (including the proposed DoE approach) are able to consistently outperform untuned classifiers in these small data settings. This raises questions about the efficacy of existing hyperparameter optimization methods in these very small sample regimes ($n \leq 200$). However, work is needed to replicate these findings across a larger set of classifiers, datasets, and optimization methods. Overall, our results show that while the proposed DoE approach is highly successful at debiasing CV error outcomes from hyperparameter optimization in small datasets, it shares the limitations in generalization performance found in other approaches. Variation in outcomes across the two tested classifiers (SVM & GBM) for the DoE approach in particular leave room for more specific investigation of which specific classifiers and hyperparameters are best suited to this type of methodology.

References

Muhammad Aftab, Tanvir Ahmad, Shahid Adeel, Sajjad Haider Bhatti, and Muhammad Irfan. Hyperparameter tuning through innovative designing to avoid over-fitting in machine learning modelling: a case study of small data sets. *Journal of Statistical Computation and Simulation*, 95(7):1595–1609, 2025.

Majd Alqarqaz, Maram Bani Younes, and Raneem Qaddoura. An object classification approach for autonomous vehicles using machine learning techniques. *World Electric Vehicle Journal*, 14(2):41, 2023.

Chansik An, Yae Won Park, Sung Soo Ahn, Kyunghwa Han, Hwiyoung Kim, and Seung-Koo Lee. Radiomics machine learning study with a small sample size: Single random training-test set split may lead to unreliable results. *PLoS One*, 16(8):e0256152, 2021.

George I Austin, Itsik Pe'er, and Tal Korem. Distributional bias compromises leave-one-out cross-validation. *Science Advances*, 11(48):eadx6976, 2025.

Mahboube Ayoubi, Babak Teimourpour, and Alireza Hassanzadeh. Exgenet, integrating design of experiments and response surface methodology for cancer gene detection in gene regulatory networks. *Cancer Informatics*, 23:11769351241255645, 2024.

Claudia Beleites, Richard Baumgartner, Christopher Bowman, Ray Somorjai, Gerald Steiner, Reiner Salzer, and Michael G Sowa. Variance reduction in estimating classification error using sparse datasets. *Chemometrics and intelligent laboratory systems*, 79(1-2):91–100, 2005.

Visar Berisha, Chelsea Krantsevich, P Richard Hahn, Shira Hahn, Gautam Dasarathy, Pavan Turaga, and Julie Liss. Digital medicine and the curse of dimensionality. *NPJ digital medicine*, 4(1):153, 2021.

Visar Berisha, Chelsea Krantsevich, Gabriela Stegmann, Shira Hahn, and Julie Liss. Are reported accuracies in the clinical speech machine learning literature overoptimistic? In *Interspeech*, pp. 2453–2457, 2022.

Bernd Bischl, Martin Binder, Michel Lang, Tobias Pielok, Jakob Richter, Stefan Coors, Janek Thomas, Theresa Ullmann, Marc Becker, Anne-Laure Boulesteix, et al. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 13(2):e1484, 2023.

Sinem Bozkurt Keser and Yeliz Buruk Sahin. Response surface methodology to tune artificial neural network hyper-parameters. *Expert Systems*, 38(8):e12792, 2021.

Etienne Combrisson and Karim Jerbi. Exceeding chance level by chance: The caveat of theoretical chance levels in brain signal classification and statistical assessment of decoding accuracy. *Journal of neuroscience methods*, 250:126–136, 2015.

Bradley Efron. Empirical bayes estimates for large-scale prediction problems. *Journal of the American Statistical Association*, 104(487):1015–1028, 2009.

Matthias Feurer, Jan N. van Rijn, Arlind Kadra, Pieter Gijsbers, Neeratyoy Mallik, Sahithya Ravi, Andreas Müller, Joaquin Vanschoren, and Frank Hutter. Openml-python: an extensible python api for openml. *Journal of Machine Learning Research*, 22(100):1–5, 2021. URL `http://jmlr.org/papers/v22/19-920.html`.

Sofia De la Fuente Garcia, Craig W Ritchie, and Saturnino Luz. Artificial intelligence, speech, and language processing approaches to monitoring alzheimer's disease: a systematic review. *Journal of Alzheimer's Disease*, 78(4):1547–1574, 2020.

Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, et al. A2d2: Audi autonomous driving dataset. *arXiv preprint arXiv:2004.06320*, 2020.

Peter Goos and Bradley Jones. *Optimal design of experiments: a case study approach*. John Wiley & Sons, 2011.

Tim Head, MechCoder, Gilles Louppe, Iaroslav Shcherbatyi, fcharras, Zé Vinícius, cmmalone, Christopher Schröder, nel215, Nuno Campos, Todd Young, Stefano Cereda, Thomas Fan, rene rex, Kejia (KJ) Shi, Justus Schwabedal, carlosdanielcsantos, Hvass-Labs, Mikhail Pak, SoManyUsernamesTaken, Fred Callaway, Loïc Estève, Lilian Besson, Mehdi Cherti, Karlson Pfannschmidt, Fabian Linzberger, Christophe Cauet, Anna Gut, Andreas Mueller, and Alexander Fabisch. scikit-optimize/scikit-optimize: v0.5.2, March 2018. URL `https://doi.org/10.5281/zenodo.1207017`.

Kayleigh K Hyde, Marlena N Novack, Nicholas LaHaye, Chelsea Parlett-Pelleriti, Raymond Anden, Dennis R Dixon, and Erik Linstead. Applications of supervised machine learning in autism spectrum disorder research: a review. *Review Journal of Autism and Developmental Disorders*, 6(2):128–146, 2019.

Rachel T Johnson, Douglas C Montgomery, Bradley Jones, and Peter A Parker. Comparing computer experiments for fitting high-order polynomial metamodels. *Journal of Quality Technology*, 42(1):86–102, 2010.

Tinu Theckel Joy, Santu Rana, Sunil Gupta, and Svetha Venkatesh. Hyperparameter tuning for big data using bayesian optimisation. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2574–2579. IEEE, 2016.

Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. Fast bayesian optimization of machine learning hyperparameters on large datasets. In *Artificial intelligence and statistics*, pp. 528–536. PMLR, 2017.

Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pp. 1137–1145. Montreal, Canada, 1995.

Peter Kokol, Marko Kokol, and Sašo Zagoranski. Machine learning on small size samples: A synthetic knowledge synthesis. *Science Progress*, 105(1):00368504211029777, 2022.

Robyn Larracy, Angkoon Phinyomark, and Erik Scheme. Machine learning model validation for early stage studies with small sample sizes. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pp. 2314–2319. IEEE, 2021.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Yang Liu, Jiahuan Cao, Chongyu Liu, Kai Ding, and Lianwen Jin. Datasets for large language models: A comprehensive survey. *Artificial Intelligence Review*, 58(12):403, 2025.

Gustavo A Lujan-Moreno, Phillip R Howard, Omar G Rojas, and Douglas C Montgomery. Design of experiments and response surface methodology to tune machine learning hyperparameters, with a random forest case-study. *Expert Systems with Applications*, 109:195–205, 2018.

Anastasia Makarova, Huibin Shen, Valerio Perrone, Aaron Klein, Jean Baptiste Faddoul, Andreas Krause, Matthias Seeger, and Cedric Archambeau. Overfitting in bayesian optimization: an empirical study and early-stopping solution. In *2nd Workshop on Neural Architecture Search (NAS 2021)@ ICLR 2021*. NAS 2021, 2021.

Warut Pannakkong, Kwanluck Thiwa-Anont, Kasidit Singthong, Parthana Parthanadee, and Jirachai Buddhakulsomsiri. Hyperparameter tuning of machine learning algorithms using response surface methodology: a case study of ann, svm, and dbn. *Mathematical problems in engineering*, 2022(1):8513719, 2022.

Sree Hari Krishnan Parthasarathi and Nikko Strom. Lessons from building acoustic models with a million hours of speech. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6670–6674. IEEE, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

Philipp Probst, Anne-Laure Boulesteix, and Bernd Bischl. Tunability: Importance of hyperparameters of machine learning algorithms. *Journal of Machine Learning Research*, 20(53):1–32, 2019.

Kanishka Rao, Haşim Sak, and Rohit Prabhavalkar. Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer. In *2017 IEEE automatic speech recognition and understanding workshop (ASRU)*, pp. 193–199. IEEE, 2017.

Pouria Saidi, Gautam Dasarathy, and Visar Berisha. Unraveling overoptimism and publication bias in ml-driven science. *Patterns*, 6(4), 2025.

Chenlu Shi, Ashley Kathleen Chiu, and Hongquan Xu. Evaluating designs for hyperparameter tuning in deep neural networks. *The New England Journal of Statistics in Data Science*, 1(3):334–341, 2023.

Samuel Stanton, Wesley Maddox, Nate Gruver, Phillip Maffettone, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson. Accelerating bayesian optimization for biological sequence design with denoising autoencoders. In *International conference on machine learning*, pp. 20459–20478. PMLR, 2022.

Ryan J Tibshirani and Robert Tibshirani. A bias correction for the minimum error rate in cross-validation. 2009.

Ioannis Tsamardinos, Elissavet Greasidou, and Giorgos Borboudakis. Bootstrapping the out-of-sample predictions for efficient and accurate cross-validation. *Machine learning*, 107(12):1895–1922, 2018.

Andrius Vabalas, Emma Gowen, Ellen Poliakoff, and Alexander J Casson. Machine learning algorithm validation with a limited sample size. *PloS one*, 14(11):e0224365, 2019.

Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Joaquin Vanschoren, Jan N Van Rijn, Bernd Bischl, and Luis Torgo. Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.

Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*, 7(1):91, 2006.

Jair Vasquez-Ramos, María Guadalupe Ruiz-Sandoval, Diego Oliva, Oscar Ramos-Soto, Jorge Ramos-Frutos, Marwa Sharawi, and Marco Pérez-Cisneros. Response surface-driven hyperparameter optimization for xgboost: J. vasquez-ramos et al. *The Journal of Supercomputing*, 81(10):1112, 2025.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Xilu Wang, Yaochu Jin, Sebastian Schmitt, and Markus Olhofer. Recent advances in bayesian optimization. *ACM Computing Surveys*, 55(13s):1–36, 2023.