


Honest leave-one-out cross-validation for estimating post-tuning generalization error

Boxiang Wang¹ | Hui Zou² 

¹Department of Statistics and Actuarial Science, University of Iowa, Iowa City, IA, 52242, USA

²School of Statistics, University of Minnesota, Minneapolis, MN, 55455, USA

Correspondence

Hui Zou, School of Statistics, University of Minnesota, Minneapolis, MN 55455, USA.
Email: zouxx019@umn.edu

Funding information

NSF, Grant/Award Numbers: 1915-842, 2015-120

Many machine learning models have tuning parameters to be determined by the training data, and cross-validation (CV) is perhaps the most commonly used method for selecting tuning parameters. This work concerns the problem of estimating the generalization error of a CV-tuned predictive model. We propose to use an honest leave-one-out cross-validation framework to produce a nearly unbiased estimator of the post-tuning generalization error. By using the kernel support vector machine and the kernel logistic regression as examples, we demonstrate that the honest leave-one-out cross-validation has very competitive performance even when competing with the state-of-the-art .632+ estimator.

KEYWORDS

bootstrap, prediction, resampling methods, statistical learning

1 | INTRODUCTION

Many modern machine learning models involve important tuning parameters to be determined by the data. For example, the number of boosting iterations and the size of each tree are some of the tuning parameters in tree-based gradient boosting, and the penalization parameter is a tuning parameter in the kernel support vector machine (Cortes & Vapnik, 1995; Vapnik, 1995), the lasso regression model and ridge regression. Selecting the right tuning parameter is referred to as tuning in the literature, and many statistical methods and theory have been devoted to the topic of tuning. Some popular approaches include information criterion-based methods such as Akaike information criterion (Akaike, 1973), Bayesian information criterion (Stein, 1981a), Stein's unbiased risk estimation (SURE, Stein, 1981b) or some resampling methods like cross-validation (CV, e.g., Allen, 1974; Arlot & Celisse, 2010; Lachenbruch & Mickey, 1968; Stone, 1974; Wahba & Wold, 1975) and bootstrap (Efron & Tibshirani, 1994). Among those proposals, CV is perhaps the most popular technique for tuning in practice. In a standard implementation of CV, one segments the training data into V roughly equal-sized subsets, trains the model on each $(V - 1)$ -fold ensemble and computes the prediction error of the trained model on the remaining fold. Typical choices of V are the sample size n , 10, 5, corresponding to leave-one-out cross-validation (LOOCV), 10-fold and 5-fold CV, respectively. In practice, CV tuning selects the parameter that incurs the least CV error. The basic idea of CV is to obtain a natural estimate of the generalization error of a given model via data splitting. Actually, the LOOCV error is nearly unbiased in terms of estimation bias (Luntz & Brailovsky, 1969). In practice, 10-fold and 5-fold CV are much more popular due to reduced computational efforts, although they tend to have larger bias in terms of estimating the generalization error. As pointed out by Yang (2006), estimating the generalization error and tuning the model are two related but very different tasks. In some settings, estimating the generalization error is more demanding while better estimation of the error does not necessarily lead to a better tuning parameter.

One objection of the LOOCV error comes from the high variance claim: there is a wide-spreading argument that LOOCV has a much larger variance than 10-fold or 5-fold CV such that the LOOCV error is a statistically inferior estimator of the generalization error. The argument is more or less heuristic. Actually, some papers (e.g., Bengio & Grandvalet, 2004; Burman, 1989; Kohavi, 1995; Molinaro et al. 2005) have theoretically and numerically demonstrated that the variance of the LOOCV error tends to be similar to the variance of other V -fold CV errors in many applications; Zhang and Yang (2015) even claimed "LOOCV in fact has the smallest variability" for fixed models in the regression setting. Despite these papers, the misconception still widely exists; for example, see discussions in chapter 7.10 in the famous book, Hastie et al. (2009). In fact, Wang

and Zou (2021) have shown that LOOCV does not have higher variance than 10-fold CV and 5-fold CV in the context of binary classification. Here, we show a graphical illustration. We fit the kernel support vector machine (SVM) on a data set drawn from a mixture Gaussian distribution whose Bayes decision boundary is non-linear. The details of the data generating model are given in the caption of Figure 1, from which we observe that (i) LOOCV has almost no bias in estimating the generalization error, while the bias largely increases as V decreases; (ii) the variance of the CV error exhibits very little difference among $V = 5, 10, n$, but the variance is much larger in two-fold CV.

A legitimate complaint of LOOCV arises from its expensive computation. The vanilla implementation of LOOCV requires n repeats of model fitting, which seems to be too expensive. Thus, many people prefer to use 10-fold or 5-fold CV over LOOCV. This claim can be traced back to the invention of V -fold CV at the outset as a computational remedy of LOOCV (Geisser, 1975). However, the CV error depends on the way one splits the data and hence would vary even if the same CV procedure is deployed with a different data split. Rodríguez et al. (2010) studied repeated V -fold CV to stabilize the CV procedure, but the total computational cost becomes much more expensive. In contrast, LOOCV enjoys the advantage of having a deterministic data split scheme. Moreover, a smart implementation of LOOCV can make the total computation time much less than that by the vanilla implementation. Let us take the ridge regression as an example. Golub et al. (1979) introduced a neat formula to allow for efficient computation of the exact LOOCV error with basically the same cost as fitting a single ridge regression model. This nice result can be generalized to a class of linear smoothers that possess the self-stable property (Fan et al. 2020). As such, computational cost will not become a deciding factor that prevents users from using LOOCV to estimate the generalization error in real applications. In Wang and Zou (2021), the neat formula for ridge regression is extended to the large-margin classifiers and a new algorithm has been proposed for fast and exact LOOCV computation of a kernel SVM or related kernel classifiers.

The above discussion focuses on a single given model and how to estimate its generalization error. In practice, we select a final model from a list of candidate models. Now, suppose that the final model is already selected and one is willing to use this model for predicting future unseen data. Naturally, we would like to know its true accuracy. Statistically speaking, we aim to find a good estimator of the generalization error of the selected model. In particular, consider the final model chosen by CV (10-fold CV or LOOCV), how to estimate the generalization error of this CV-tuned model?

In a series of papers (e.g., Efron, 1983, 1986, 2004; Efron & Tibshirani, 1997), Efron studied the problem of estimating the generalization error of any given model and applied his methods to tuning. Conceptually, his results can be applied to a tuned model to obtain a good estimate of the generalization error of the tuned model. The caveat is to consider the tuning process as a part of the selected model. According to Efron, the state-of-the-art method is the .632+ bootstrap estimator (Efron & Tibshirani, 1997). On the other hand, we have not seen the use of the .632+ estimator in the context of post-tuning estimation. In this paper, we consider the kernel SVM and the kernel logistic regression as examples. We find that the .632+ estimator still performs very well for estimating the generalization error of a CV-tuned kernel SVM and kernel logistic regression. Moreover, we propose an honest LOOCV (HLOOCV) estimator for estimating the generalization error of a CV-tuned kernel SVM and kernel logistic regression. We point out that a naive plug-in LOOCV estimator is biased and the honest LOOCV estimator is nearly unbiased. We further compare the honest LOOCV estimator with the .632+ estimator on various numerical examples and show that the honest LOOCV has very competitive performance even against the .632+ estimator.

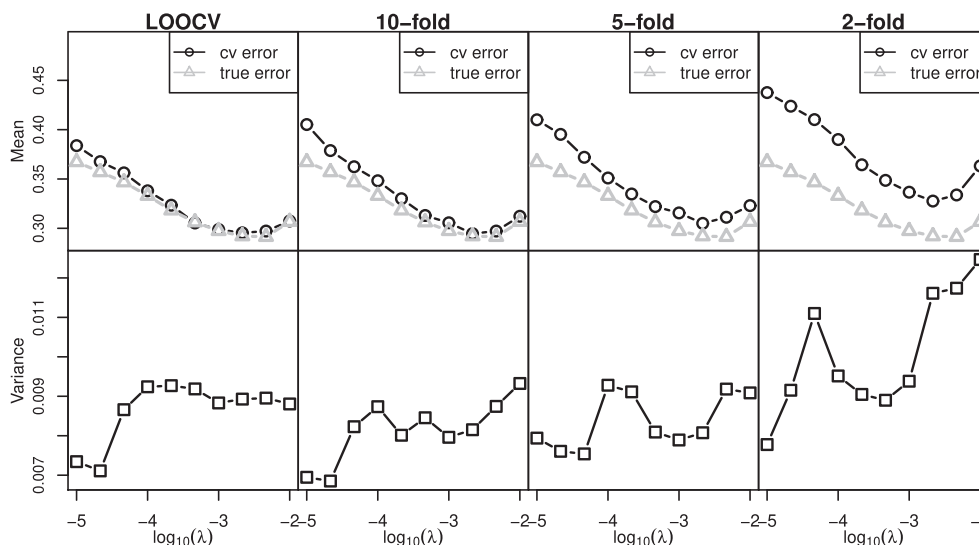


FIGURE 1 The mean and variance of the classification error of the kernel SVM. The training data (the sample size: $n = 100$, the number of input variables: $p = 5$) are generated from a mixture of Gaussian models. We draw 10 centres μ_k from $N((2, 2, 0, 0), \mathbf{I})$, where \mathbf{I} is the p -dimensional identity matrix, and we then randomly pick up one centre and generate one positive-class observation from $N(\mu_k, 4\mathbf{I})$. Observations from the negative class are generated similarly, with $N((0, 2, 2, 0), \mathbf{I})$. The Bayes error is 24.3%. We assess the true error on 10,000 observations that are generated independently. The mean and variance of the classification error are calculated on the basis of 100 independent runs

2 | METHOD

2.1 | Honest leave-one-out cross-validation

We first formally define the generalization error of a tuned model. Suppose training data $\mathcal{F}_n = \{\mathbf{x}_i, y_i\}_{i=1}^n$ are a random sample drawn from an unknown distribution $\mathcal{P}_{\mathcal{F}}$. Denote by \mathcal{A}_λ a family of generic algorithms that are indexed by a parameter λ . By implementing \mathcal{A}_λ on the training data \mathcal{F}_n , we obtain a model \hat{f}_λ which predicts the outcome of data \mathbf{x} to be $\hat{f}_\lambda(\mathbf{x})$. An example of the generic algorithm \mathcal{A}_λ that we use to demonstrate the idea is the kernel large-margin classifier:

$$\hat{f}_\lambda = \operatorname{argmin}_{f \in \mathcal{H}_K} \left[\frac{1}{n} \sum_{i=1}^n L(y_i f(\mathbf{x}_i)) + \lambda \|f\|_{\mathcal{H}_K}^2 \right], \quad (2.1)$$

where $L(u)$ is a margin-based loss function for classification and \mathcal{H}_K is the reproducing kernel Hilbert space (RKHS) induced by a kernel function K . For example, the kernel SVM uses $L(u) = \max(1 - u, 0)$ (the hinge loss), and formulation (2.1) leads to the kernel logistic regression if $L(u) = \log(1 + e^{-u})$ (Hastie et al. 2009). To evaluate the accuracy of the model \hat{f}_λ , a loss function $\phi(y, s)$ is used. For classification, $\phi(y, s)$ is the 0–1 loss by default, but it can also be the binomial deviance or other loss functions. To select the tuning parameter λ from a candidate set Λ , a generic tuning procedure Δ such as LOOCV, 10-fold CV or bootstrap is deployed. In this work, we consider LOOCV as the tuning method.

Denote by $\mathcal{F}_n^{[-i]}$ the training data with the i th observation removed and denote by $\hat{f}_\lambda^{[-i]}$ the predictive model according to the algorithm \mathcal{A}_λ fitted on $\mathcal{F}_n^{[-i]}$. Then the LOOCV error is computed as

$$\widehat{\operatorname{Err}}_\phi(\hat{f}_\lambda) \equiv \sum_{i=1}^n \phi(y_i, \hat{f}_\lambda^{[-i]}(\mathbf{x}_i)). \quad (2.2)$$

The tuning procedure thereby outputs $\hat{\Delta}(\Lambda) = \operatorname{argmin}_{\lambda \in \Lambda} \widehat{\operatorname{Err}}_\phi(\hat{f}_\lambda)$ as an estimate of the “optimal” tuning parameter. After selecting the tuning parameter λ for the algorithm \mathcal{A}_λ , we have finalized a tuned model from the training data. We write the tuned model as $\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}$, where the subscript n is appended to highlight that the model is constructed with the sample size n . We say the tuned model predicts the response of unseen test data \mathbf{x} as $\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}(\mathbf{x})$. Consequently, the generalization error of the tuned model $\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}$ under the ϕ -loss is defined as

$$\operatorname{Err}_\phi(\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}) = \mathbb{E}_{\mathcal{F}_n} \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{P}_{\mathcal{F}}} \phi(y, \widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}(\mathbf{x})),$$

which is referred to as the post-tuning generalization error of $\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}$. Our goal is to construct a good estimate of $\operatorname{Err}_\phi(\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n})$.

Since the procedure for producing $\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}$ includes the LOOCV tuning, the same tuning procedure should be echoed when evaluating the performance of $\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}$ using CV. Therefore, we suggest using the following CV within CV framework to yield an *honest LOOCV estimator* to estimate the generalization error of $\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n}$.

- For each $i = 1, \dots, n$:
 1. for each $j = 1, \dots, i-1, i+1, \dots, n$ and for each $\lambda \in \Lambda$, perform the algorithm \mathcal{A}_λ on $\mathcal{F}_n^{[-i-j]}$ to obtain the predictive model $\hat{f}_\lambda^{[-i-j]}$,
 2. select the tuning parameter according to the procedure Δ , that is, compute

$$\hat{\Delta}_i(\Lambda) = \operatorname{argmin}_{\lambda \in \Lambda} \widehat{\operatorname{Err}}_\phi(\hat{f}_\lambda^{[-i]}) = \operatorname{argmin}_{\lambda \in \Lambda} \sum_{j \neq i} \phi(y_j, \hat{f}_\lambda^{[-i-j]}(\mathbf{x}_j)),$$

3. perform the algorithm $\mathcal{A}_{\hat{\Delta}_i(\Lambda)}$ on $\mathcal{F}_n^{[-i]}$ to obtain the predictive model $\hat{f}_{\hat{\Delta}_i(\Lambda)}^{[-i]}$, and evaluate $\phi(y_i, \hat{f}_{\hat{\Delta}_i(\Lambda)}^{[-i]}(\mathbf{x}_i))$.
- The *honest LOOCV (HLOOCV) estimator* is defined as

$$\widehat{\operatorname{Err}}_\phi^{\text{HLOOCV}} = \frac{1}{n} \sum_{i=1}^n \phi(y_i, \hat{f}_{\hat{\Delta}_i(\Lambda)}^{[-i]}(\mathbf{x}_i)).$$

By its construction, it is easy to see that

$$\mathbb{E}_{\mathcal{F}_n} \widehat{\operatorname{Err}}_\phi^{\text{HLOOCV}} = \mathbb{E}_{\mathcal{F}_{n-1}} \mathbb{E}_{(y, \mathbf{x}) \sim \mathcal{P}_{\mathcal{F}}} \phi(y, \widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n-1}(\mathbf{x})) = \operatorname{Err}_\phi(\widehat{\mathcal{M}}_{\mathcal{A}, \Delta, n-1}).$$

Therefore, the honest LOOCV estimator is an exact unbiased estimator of the post-tuning generalization error of the model based on a training set with the sample size $n - 1$. Consequently, the honest LOOCV estimator is a nearly unbiased estimator of $\text{Err}_{\hat{\phi}}(\hat{\mathcal{M}}_{\lambda, \Delta, n})$, the post-tuning generalization error of the model based on a training set with the sample size n . This nice property holds for each sample size n , and other popular methods for estimating generalization error discussed in the next subsection do not necessarily have this property.

2.2 | Other proposals

In the literature, there are only few papers on the estimation of the generalization error of a post-tuning model. Tibshirani and Rosset (2019) studied the prediction error of SURE-tuned regression models. SURE is mostly used for regression models, and the covariance penalty method (Efron, 1986, 2004) can generalize SURE to classification models under the 0-1 loss. However, SURE or the covariance penalty method has to work with the fixed-X case, which is not the typical setting for some classification methods like the SVM.

It appears to be straightforward to directly use the observed smallest LOOCV error to estimate the generalization error of the tuned model:

$$\widehat{\text{Err}}_{\phi}^{\text{PLOOCV}} = \frac{1}{n} \sum_{i=1}^n \phi(y_i, \hat{f}_{\hat{\lambda}}^{[-i]}(\mathbf{x}_i)),$$

because $\hat{\lambda}$ is the chosen regularization parameter for fitting the tuned classifier. We call it the *plug-in LOOCV estimator* because $\widehat{\text{Err}}_{\phi}$ can be obtained by replacing λ with $\hat{\lambda}$ in Equation (2.2). When the sample size n is large enough so that the variability in selection is ignorable, then the plug-in estimator is expected to perform well. Given a small sample size, the plug-in estimator is too optimistic. To correct the bias, Tibshirani and Tibshirani (2009) proposed a bias-adjusted estimator:

$$\widehat{\text{Err}}_{\phi}^{\text{TT09}} = \widehat{\text{Bias}} + \min_{\lambda \in \Lambda} \left[\frac{1}{n} \sum_{i=1}^n \phi(y_i, \hat{f}_{\lambda}(\mathbf{x}_i)) \right],$$

in which the bias term is estimated as

$$\widehat{\text{Bias}} = \min_{\lambda \in \Lambda} \left[\frac{1}{n} \sum_{i=1}^n \phi(y_i, \hat{f}_{\lambda}(\mathbf{x}_i)) \right] - \frac{1}{n} \sum_{i=1}^n \min_{\lambda \in \Lambda} \left[\phi(y_i, \hat{f}_{\lambda}(\mathbf{x}_i)) \right].$$

We also consider two bootstrap-based methods, namely, .632 estimator and .632+ estimator. Efron (1983) proposed a .632 estimator for assessing the generalization error for a generic algorithm. It can be directly applied to a tuned classifier. For each $b = 1, \dots, B$, non-parametric bootstrap samples $\mathcal{F}_n^{*b} = \{\mathbf{x}_i^{*b}, y_i^{*b}\}_{i=1}^n$ are drawn with replacement from the training data \mathcal{F}_n . The kernel SVM $\hat{f}_{\lambda}^{[*b]}$ is fitted and tuned on \mathcal{F}_n^{*b} , and the *out of bootstrap* prediction error is given as

$$\widehat{\text{Err}}_{\text{out}} = \frac{\sum_{i=1}^n \sum_{b=1}^B l_i^b \phi(y_i, \hat{f}_{\lambda}^{[*b]}(\mathbf{x}_i))}{\sum_{i=1}^n \sum_{b=1}^B l_i^b},$$

where $l_i^b = 1$ if $(\mathbf{x}_i, y_i) \notin \mathcal{F}_n^{*b}$ and $l_i^b = 0$ otherwise. The .632 estimator is then defined as follows:

$$\widehat{\text{Err}}_{\phi}^{.632} = .368\overline{\text{err}} + .632\widehat{\text{Err}}_{\text{out}},$$

where $\overline{\text{err}}$ is the training error.

A more sophisticated .632+ estimator was proposed in Efron and Tibshirani (1997) as an improvement over the .632 estimator:

$$\widehat{\text{Err}}_{\phi}^{.632+} = \widehat{\text{Err}}_{\phi}^{.632} + (\widehat{\text{Err}}_{\text{out}} - \overline{\text{err}}) \frac{.368 \cdot .632 \cdot \hat{R}}{1 - .368\hat{R}},$$

where

$$\hat{R} = \frac{\widehat{\text{Err}}_{\text{out}} - \overline{\text{err}}}{\hat{\gamma} - \overline{\text{err}}},$$

and

$$\hat{y} = \left(\frac{1}{n} \sum_{i=1}^n y_i \right) \left(1 - \frac{1}{n} \sum_{i=1}^n \hat{f}_{\lambda}(\mathbf{x}_i) \right) + \left(\frac{1}{n} \sum_{i=1}^n \hat{f}_{\lambda}(\mathbf{x}_i) \right) \left(1 - \frac{1}{n} \sum_{i=1}^n y_i \right).$$

The .632+ estimator is regarded as the state-of-the-art method for estimating the generalization error of a generic algorithm. Our experiments show that it also works very well for the LOOCV-tuned classifiers.

3 | NUMERICAL STUDIES

In this section, we use the kernel SVM and the kernel logistic regression as examples to demonstrate the performance of the honest LOOCV estimator on both simulated and real data. We show that the honest LOOCV estimator is almost unbiased for the generalization error of the tuned model, as predicted by theory, and its performance is highly competitive with the .632+ estimator.

3.1 | Simulations

In the simulation study, we generated data from a mixture Gaussian distribution following the simulation model used in chapter 12 of Hastie et al. (2009). Let p be the number of input variables and \mathbf{I} be the p -dimensional identity matrix. Let $\boldsymbol{\mu}_+ = (2, \dots, 2, 0, \dots, 0)$ and $\boldsymbol{\mu}_- = (0, \dots, 0, 2, \dots, 2)$, both of which are p -dimensional vectors and have half of the coordinates to be zero. We generated 10 means $\boldsymbol{\mu}_k, k = 1, 2, \dots, 10$, from $N(\boldsymbol{\mu}_+, \mathbf{I})$ for the positive class and 10 means $\boldsymbol{\mu}_k, k = 11, 12, \dots, 20$, from $N(\boldsymbol{\mu}_-, \mathbf{I})$ for the negative class. For each observation, we first picked an integer m from $\{1, 2, \dots, 20\}$ with equal probability, and we then generated a $N(\boldsymbol{\mu}_m, 4\mathbf{I})$. We varied the sample size n as $\{100, 200, 300\}$ and fixed the dimension as $p = 20$. In each example, we fitted the kernel SVM and the kernel logistic regression with the Gaussian kernel on 10 λ values and selected the optimal λ value by LOOCV. In terms of the estimation of the generalization error of the LOOCV-tuned SVMs, we compared the honest LOOCV estimator, the plug-in LOOCV estimator, the bias-adjusted estimator (Tibshirani & Tibshirani, 2009), the .632 estimator and the .632+ estimator. The generalization error was computed by using 1 million independently generated test data. By repeating the experiments 500 times, we computed the bias, standard deviation and the root mean squared error (RMSE) of each estimator.

From Table 1, we make several observations for both the kernel SVM and logistic regression. First, the plug-in LOOCV estimator underestimates the prediction error and thus exhibits a negative bias. Second, the estimator proposed by Tibshirani and Tibshirani (2009) has a significantly large upward bias and is excessively conservative in this simulation example; and its RMSE is significantly larger than the other methods. Third, both the .632 and .632+ estimators yield large negative bias; the .632+ estimator is clearly an improvement over the .632 estimator in terms of RMSE. Fourth, the RMSE of the honest LOOCV estimator is less than the .632+ estimator when $n = 200, 300$ and the honest LOOCV estimator is the least biased among all the methods. It is worth noting that the existing work on estimating the generalization error all heavily focused on the bias reduction; for example, see Efron (1983), Efron and Tibshirani (1997) and Tibshirani and Tibshirani (2009). The honest LOOCV estimator is in general auspicious as it is nearly unbiased and its RMSE is highly competitive with the .632+ estimator, the current state-of-the-art method for estimating the generalization error. Fifth, in terms of estimation variance, the .632 estimator is the best and the plug-in LOOCV estimator is better than HLOOCV. The smaller variance of the plug-in LOOCV estimator compensates its large bias, resulting in an even slightly better RMSE than the honest LOOCV.

3.2 | Benchmark data

We further demonstrate the performance of the honest LOOCV using benchmark data. In order to accurately compute the true generalization error, we used a large-scale benchmark data set, `covtype`, from the UCI machine learning repository (Dua & Graff, 2019). The total sample size is 495,141. We randomly selected $\{100, 200, 300\}$ observations as the training data and treated the remaining data as the test data. The random selection was repeated 500 times. In Table 2, we displayed the estimators and their bias, standard deviation and RMSE. We discover the honest LOOCV estimator tends to unbiased. The plug-in LOOCV estimator is biased downward. The estimator by Tibshirani and Tibshirani (2009) overestimates the generalization error and yields very large RMSE. We find that the .632+ estimator enjoys the lowest RMSE and does perform better than the .632 estimator. When $n = 200$ and 300, the plug-in LOOCV, the honest LOOCV and the .632+ estimators are the three best methods, and the honest LOOCV has a very clear advantage in terms of bias. Again, we note that in terms of estimation variance, the .632 estimator is the best and the plug-in LOOCV estimator is better than HLOOCV. The smaller variance of the plug-in LOOCV estimator compensates its large bias, resulting in an even slightly better RMSE than the honest LOOCV.

TABLE 1 Comparisons on five estimators for the post-tuning generalization error of the kernel SVM and the kernel logistic regression

<i>n</i>	Method	Truth	Criteria	HLOOCV	PLOOCV	TT09	.632	.632+
100	SVM	27.06	est (%)	27.11	24.37	33.96	18.67	24.18
			Bias (%)	0.06	-2.69	6.91	-8.38	-2.88
			$\sqrt{\text{var}}$ (%)	6.13	5.30	8.63	3.11	5.16
			RMSE (%)	6.13	5.95	11.05	8.94	5.91
	Logistic	26.79	est (%)	26.64	24.79	31.04	19.06	24.27
			Bias (%)	-0.15	-1.99	4.25	-7.73	-2.51
			$\sqrt{\text{var}}$ (%)	5.63	5.08	7.31	3.12	4.99
			RMSE (%)	5.63	5.46	8.45	8.33	5.59
200	SVM	24.40	est (%)	24.39	22.69	31.63	17.00	20.75
			Bias (%)	-0.00	-1.71	7.24	-7.40	-3.64
			$\sqrt{\text{var}}$ (%)	3.70	3.11	5.50	1.92	2.82
			RMSE (%)	3.70	3.55	9.09	7.64	4.61
	Logistic	24.10	est (%)	24.13	22.90	28.25	18.62	21.29
			Bias (%)	0.03	-1.20	4.15	-5.48	-2.81
			$\sqrt{\text{var}}$ (%)	3.47	3.08	4.78	2.12	2.77
			RMSE (%)	3.47	3.31	6.33	5.88	3.95
300	SVM	23.47	est (%)	23.36	22.19	30.26	17.18	19.91
			Bias (%)	-0.11	-1.28	6.79	-6.29	-3.56
			$\sqrt{\text{var}}$ (%)	2.99	2.63	4.70	1.71	2.29
			RMSE (%)	3.00	2.92	8.26	6.52	4.24
	Logistic	23.24	est (%)	23.16	22.33	27.09	19.12	20.74
			Bias (%)	-0.07	-0.91	3.85	-4.11	-2.49
			$\sqrt{\text{var}}$ (%)	2.71	2.51	3.76	1.87	2.23
			RMSE (%)	2.71	2.67	5.38	4.52	3.35

Note: The classifiers are trained on training data generated from the mixture Gaussian distributions. Five estimators are honest LOOCV (HLOOCV), plug-in LOOCV (PLOOCV), the bias-adjusted method proposed in Tibshirani and Tibshirani (2009), .632 estimator and .632+ estimator. The term "Truth" stands for the true generalization error computed via Monte Carlo. Reported numbers are the estimates (in the row labelled est) as well as the corresponding bias, standard deviation ($\sqrt{\text{var}}$) and root mean squared error (RMSE), based on 500 replicates.

4 | DISCUSSION

In this work, we have proposed an honest LOOCV estimator for estimating the post-tuning generalization error. We have also compared it with several proposals in the literature, including a naive plug-in LOOCV estimator, a bias-adjusted method proposed by Tibshirani and Tibshirani (2009) and two improved bootstrap methods, the .632 and .632+ estimators. The .632+ estimator is so far the state-of-the-art estimator for the generalization error. Although all these methods aim to reduce the bias, our studies reveal that these methods still have a large bias when estimating the generalization error. By contrast, our honest LOOCV estimator is nearly unbiased, and the variance is roughly on the same scale with these competitors.

One may concern about high computational cost of the honest LOOCV, especially when conducting the generic algorithm itself is computationally expensive. We note that the computation issue is not just for the honest LOOCV. The state-of-the-art .632+ method is also computationally demanding. For example, the standard LOOCV implementation of the kernel SVM has computational complexity $O(n^4)$. Thus, the standard implementation of the honest LOOCV estimator has computational complexity $O(n^5)$, while the .632+ method has computational complexity $O(Bn^4)$. Usually, $B=200$ and hence the cost of the .632+ estimator is at least of the same order of that of the honest LOOCV when n is in the range 100–500. To handle the computational issue, we have employed the algorithm developed in Wang and Zou (2021) to efficiently compute the LOOCV-tuned SVM and obtain the honest LOOCV error. As shown in Wang and Zou (2021), the exact LOOCV computation for the kernel margin classifiers is roughly $O(n^3)$ complexity, which reduces the cost of honest LOOCV estimator and the .632+ estimator down to $O(n^4)$ and $O(Bn^3)$, respectively. The reduction in computation is also true for the kernel logistic regression.

Finally, we point out that our numerical results suggest that it is a difficult task to estimate the generalization error of a tuned predictive model when the sample size is not large. Although the honest LOOCV estimator has nearly zero bias, as predicted by theory, all the methods

TABLE 2 Estimation of the post-tuning generalization error of the kernel SVM and the kernel logistic regression on the benchmark data set covtype

<i>n</i>	Method	Truth	Method	HLOOCV	PLOOCV	TT09	.632	.632+
100	SVM	30.38	est (%)	30.34	26.73	38.24	22.50	28.90
			Bias (%)	-0.04	-3.65	7.86	-7.88	-1.47
			$\sqrt{\text{var}}$ (%)	6.95	5.35	8.78	3.27	5.18
			RMSE (%)	6.95	6.48	11.79	8.53	5.39
	Logistic	29.95	est (%)	30.04	27.56	35.91	23.42	28.76
			Bias (%)	0.09	-2.39	5.96	-6.52	-1.19
			$\sqrt{\text{var}}$ (%)	5.92	5.02	7.40	3.43	4.98
			RMSE (%)	5.92	5.55	9.50	7.37	5.12
200	SVM	27.26	est (%)	26.97	25.16	36.66	22.35	25.62
			Bias (%)	-0.28	-2.09	9.40	-4.91	-1.64
			$\sqrt{\text{var}}$ (%)	4.07	3.57	6.34	2.65	3.45
			RMSE (%)	4.08	4.14	11.34	5.58	3.82
	Logistic	27.02	est (%)	26.94	25.59	34.50	23.34	25.77
			Bias (%)	-0.08	-1.43	7.48	-3.68	-1.24
			$\sqrt{\text{var}}$ (%)	3.78	3.38	5.24	2.67	3.27
			RMSE (%)	3.78	3.67	9.13	4.54	3.49
300	SVM	25.95	est (%)	25.77	24.38	35.69	22.42	24.42
			Bias (%)	-0.17	-1.57	9.75	-3.52	-1.52
			$\sqrt{\text{var}}$ (%)	3.14	2.77	5.04	2.19	2.59
			RMSE (%)	3.14	3.18	10.97	4.15	3.01
	Logistic	25.81	est (%)	25.62	24.67	33.61	23.27	24.70
			Bias (%)	-0.18	-1.14	7.80	-2.53	-1.11
			$\sqrt{\text{var}}$ (%)	2.99	2.64	4.28	2.20	2.49
			RMSE (%)	3.00	2.87	8.90	3.35	2.73

Note: The original data set has 495,141 observations and is randomly split into training and test data. The generalization error is assessed on test data (denoted by Truth) and estimated on training data using honest LOOCV (denoted by HLOOCV), plug-in LOOCV (denoted by PLOOCV), the bias-adjusted estimator proposed in Tibshirani and Tibshirani (2009) (denoted by TT09), .632 estimator and .632+ estimator. The term "Truth" stands for the true generalization error computed on the test data. Reported numbers are the estimates (in the row labelled est) as well as the corresponding bias, standard deviation ($\sqrt{\text{var}}$) and root mean squared error (RMSE), based on 500 replicates.

included in the study exhibit comparable but large variance. It would be very interesting to have an estimator that can further reduce the estimation variance without sacrificing the bias property of honest LOOCV, and the new estimator should at least be computationally manageable in order to be considered as a practical solution.

ACKNOWLEDGEMENTS

We thank the referees for their helpful comments and suggestions. Zou's work is supported in part by NSF grants 1915-842 and 2015-120.

DATA AVAILABILITY STATEMENT

The code for generating the simulated data used in this work is available from the authors. The real data examples used in this work are from the UCI machine learning repository (Dua and Graff 2019). <http://archive.ics.uci.edu/ml>, Irvine, CA

ORCID

Hui Zou  <https://orcid.org/0000-0003-4798-9904>

REFERENCES

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*, pp. 267-281.

- Allen, D. (1974). The relationship between variable selection and data augmentation and a method for prediction. *Technometrics*, 16(1), 125–127.
- Arlot, S., & Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4, 40–79.
- Bengio, Y., & Grandvalet, Y. (2004). No unbiased estimator of the variance of K-fold cross-validation. *Journal of Machine Learning Research*, 5, 1089–1105.
- Burman, P. (1989). A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, 76(3), 503–514.
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.
- Dua, D., & Graff, C. (2019). *UCI machine learning repository*. Irvine CA: University of California, School of Information and Computer Science. <https://archive.ics.uci.edu/ml>
- Efron, B. (1983). Estimating the error rate of a prediction rule: Improvement on cross-validation. *Journal of the American Statistical Association*, 78(382), 316–331.
- Efron, B. (1986). How biased is the apparent error rate of a prediction rule? *Journal of the American Statistical Association*, 81(394), 461–470.
- Efron, B. (2004). The estimation of prediction error: covariance penalties and cross-validation. *Journal of the American Statistical Association*, 99(467), 619–632.
- Efron, B., & Tibshirani, R. (1994). *An introduction to the bootstrap*: CRC Press.
- Efron, B., & Tibshirani, R. (1997). Improvements on cross-validation: The 632+ bootstrap method. *Journal of the American Statistical Association*, 92(438), 548–560.
- Fan, J., Li, R., Zhang, C., & Zou, H. (2020). *Statistical foundations of data science*: CRC Press.
- Geisser, S. (1975). The predictive sample reuse method with applications. *Journal of the American Statistical Association*, 70(350), 320–328.
- Golub, G., Heath, M., & Wahba, G. (1979). Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2), 215–223.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Prediction, inference and data mining*. (2nd edition). New York: Springer-Verlag.
- Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *International Joint Conference on Artificial Intelligence*, 14(2), 1137–1145.
- Lachenbruch, P., & Mickey, M. (1968). Estimation of error rates in discriminant analysis. *Technometrics*, 10(1), 1–11.
- Luntz, A., & Brailovsky, V. (1969). On estimation of characters obtained in statistical procedure of recognition (in Russian). *Technicheskaya Kibernetika*, 3(6), 6–12.
- Molinero, A., Simon, R., & Pfeiffer, R. (2005). Prediction error estimation: A comparison of resampling methods. *Bioinformatics*, 21(15), 3301–3307.
- Rodríguez, J., Pérez, A., & Lozano, J. (2010). Sensitivity analysis of k-fold cross validation in prediction error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3), 569–575.
- Stein, C. (1981a). Estimating the dimension of a model. *Annals of Statistics*, 6(2), 461–464.
- Stein, C. (1981b). Estimation of the mean of a multivariate normal distribution. *Annals of Statistics*, 9(6), 1135–1151.
- Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B*, 36(2), 111–147.
- Tibshirani, R. J., & Rosset, S. (2019). Excess optimism: How biased is the apparent error of an estimator tuned by SURE? *Journal of the American Statistical Association*, 114(526), 697–712.
- Tibshirani, R. J., & Tibshirani, R. (2009). A bias correction for the minimum error rate in cross-validation. *Annals of Applied Statistics*, 3(2), 822–829.
- Vapnik, V. (1995). *The nature of statistical learning theory*: Springer-Verlag.
- Wahba, G., & Wold, S. (1975). A completely automatic French curve: Fitting spline functions by cross validation. *Communications in Statistics-Theory and Methods*, 4(1), 1–17.
- Wang, B., & Zou, H. (2021). Fast and exact leave-one-out analysis of large-margin classifiers. manuscript.
- Yang, Y. (2006). Comparing learning methods for classification. *Statistica Sinica*, 16, 635–657.
- Zhang, Y., & Yang, Y. (2015). Cross-validation for selecting a model selection procedure. *Journal of Econometrics*, 187(1), 95–112.

How to cite this article: Wang, B., & Zou, H. (2021). Honest leave-one-out cross-validation for estimating post-tuning generalization error. *Stat*, 10(1), e413. <https://doi.org/10.1002/sta4.413>