Smoothed Bootstrap and Statistical Data Cloning for Classifier Evaluation

Gregory Shakhnarovich GREGORY@AI.MIT.EDU
Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139, USA

Ran El-Yaniv RANI@CS.TECHNION.AC.IL
Department of Computer Science, Technion, Haifa 32000, Israel

Yoram Baram BARAM@CS.TECHNION.AC.IL
Department of Computer Science,Technion, Haifa 32000, Israel

Abstract

This paper is concerned with the estimation of a classifier's accuracy. We present a number of novel bootstrap estimators, based on kernel smoothing, that consistently show superior performance on both synthetic and real data, with respect to other established methods. We call the process of (re)sampling the data via kernel-based smoothed bootstrap *data cloning*. The new cloning methods outperform cross-validation and the .632+ bootstrap, which, according to Efron and Tibshirani, is the estimator of choice. Finally, we extend our estimators to complex real-life data sets, in which a data point might include real, bounded, integer and nominal attributes, thus allowing for better classifier evaluation over limited real data repositories such as the UCI repository.

1. Introduction

A common approach to the evaluation and comparison of inductive learning algorithms is to test them on data sets from various "real-life" settings. We shall refer to such data as "real", in contrast to data generated by artificial methods. While many theoretical conclusions can be drawn from an algorithm's performance on synthetic data, good performance on real data sets is believed to be evidence for an algorithm's practical plausibility. Public repositories of real data have appeared in recent years; one of the most known and widely used of these is the University of California at Irvine (UCI) repository (Blake & Merz, 1998).

Unfortunately, real data is expensive, tedious to collect, and often available only in limited amounts. The UCI repository, which is one of the largest, contains about 100 data sets, just a few compared to the huge number of studies using these data sets. Such overusage of data can lead to a compromise in the significance of the obtained results (Salzberg, 1997).

This problem would be greatly alleviated if one could generate arbitrarily many "new" data samples, distributed according to the same probability law as the real data set at hand. This would provide a valuable tool for the machine learning research community, since algorithm performance results on real data would be obtained with greater statistical significance. Since the true distribution of the data is unknown, this task is impossible. However, one can "clone" the available data and generate a family of data sets that are different from but similar to the original one (in a statistical sense to be defined) and evaluate the performance of an algorithm on these data sets. The bootstrap, introduced in (Efron, 1979), provides a means for such data cloning.

In this paper we propose a number of novel bootstrap estimators for classifier error. The new estimators are based on a smoothed version of bootstrap where we use a density estimation technique to resample from the available data with some additional noise. Smoothed bootstrap estimators are already known; as far as we know, however, they have never been employed for classifier error estimation. The new estimators improve on existing ones by variance reduction. We present an extensive set of simulation results on synthetic data that show the consistent advantage of our estimators over the best known estimators to date. In particular, our estimators outperform the .632+ of (Efron & Tibshirani, 1997) and the standard cross-

validation technique. We repeat the exact experiments performed by Efron and Tibshirani and test our methods with respect to various classifiers, including Support Vector Machines.

Second, we propose algorithms for computing smoothed bootstrap samples on complex real data sets, such as those in the UCI repository, that may include real, integer and nominal attributes. We refer to this method as *statistical data cloning*. Finally, we test our cloning algorithms on a few data sets from UCI, and show again that our data cloning techniques exhibit performance superior to that of the traditional estimators. Throughout the paper we use the term "cloning" for all smoothed bootstrap sampling techniques.

1.1 The learning paradigm

We consider the following supervised learning paradigm. A data set $X^{(n)}$ consists of n labeled pairs $\langle \mathbf{x}_1, y_1 \rangle, \ldots, \langle \mathbf{x}_n, y_n \rangle$. The points $\mathbf{x}_i$ are generated i.i.d. in a d-dimensional data space $\mathcal{D}$, according to an unknown probability distribution F, and $y = 0, 1, \ldots$ are *class labels*. For brevity, we shall usually use the notation $\mathbf{x}$ to refer to $\langle \mathbf{x}, y \rangle$. Given a data sample $X^{(n)}$, the *learning algorithm* $\mathcal{A}$ produces a concept $C_{\mathcal{A}}(X^{(n)}) = \mathcal{A}(X^{(n)})$ from a certain concept class. This concept, called a *classifier*, is a function that assigns a class label to each $\mathbf{x}$.

Throughout the paper, we consider the zero-one loss function and, following standard notation (e.g. (McLachlan, 1992)), we denote the error function (the penalty for the classification decision made by $\mathcal{A}(X^{(n)})$ on the *test point* $\mathbf{x}$) by $Q\left(\mathbf{x}, X^{(n)}, \mathcal{A}(X^{(n)})\right)$. We assume that this penalty is 0 for correct classification and 1 otherwise. In every context in this paper, we shall consider a certain fixed learning algorithm $\mathcal{A}$. It is convenient then to use the shorthand

$$Q\left(\mathbf{x}, X^{(n)}\right) \;=\; Q\left(\mathbf{x}, X^{(n)}, \mathcal{A}(X^{(n)})\right).$$

For further simplicity, we shall use a similar notation for the average error of the classifier trained on $X^{(n)}$, over a *test set* $W^{(m)} = (\mathbf{w}_1, \ldots, \mathbf{w}_m)$:

$$Q\left(W^{(m)}, X^{(n)}\right) \;=\; \frac{1}{m}\sum_{i=1}^{m} Q(\mathbf{w}_i, X^{(n)})$$

1.2 Generalization error of a classifier

We are interested in measuring the accuracy of a classifier by its ability to generalize, that is, to assign the correct class label to a previously unseen data point. The *true error* Err of a classifier trained on $X^{(n)}$ is

$$Err = Err(X^{(n)}, F) = \mathrm{E}_{F(\mathbf{x}_0)}[Q(\mathbf{x}_0, X^{(n)})]. \quad (1)$$

Here, the assumption is that a random test point $\langle \mathbf{x}_0, y_0 \rangle$ is distributed according to the same F as the training set. Err is sometimes called a *conditional true error*, since it depends on the random variable $X^{(n)}$. For a fixed training set, Err can be written as

$$Err(X^{(n)}, F) = \int_{\mathcal{D}} Q(\mathbf{x}_0, X^{(n)}) F(\mathbf{x}_0) d\mathbf{x}_0 \quad (2)$$

Note that this is different from the *expected true error* for a training set of size n:

$$\mu_n(F) = \mathrm{E}_{F(X^{(n)})}[Err(X^{(n)}, F)]. \quad (3)$$

In practice, given a small data set, it is not clear how to estimate the expectation over all training sets of size n . We therefore focus in this paper on the task of estimating Err.

1.3 Bias and variance in error estimation

Being based on $X^{(n)}$, which is a random variable, any estimator $\widehat{Err}$ of the true error Err is a random variable itself. Neither its *bias* $\mathrm{E}_{F(X^{(n)})}[\widehat{Err} - Err]$ nor its variance each by itself can serve as a good measure of the estimator's quality. In a single experiment, the error of an unbiased estimator with a large variance can to be greater than that of a mildly biased estimator with a small variance.

We follow the practice of using the squared error $(\widehat{Err} - Err)^2$ as a measure of estimator quality (Efron & Tibshirani, 1997), and motivate it as follows. Consider the task of obtaining a single estimate of a classifier's accuracy on a given data set. Assuming that the objective in choosing an estimator is to minimize the probability of a large absolute (or squared) error in this single estimate, then by Markov's inequality,

$$\Pr\left((Err - \widehat{Err})^2 \geq \epsilon\right) \leq \mathrm{E}\left[(Err - \widehat{Err})^2\right]/\epsilon. \quad (4)$$

The sample mean of $(Err - \widehat{Err})^2$ (MSE) is an estimate of $\mathrm{E}\left[\left(Err - \widehat{Err}\right)^2\right]$ and can be used to estimate the bound in (4). We compute the root mean squared error $(RMSE)$ to bring the value to linear scale.

2. On some known methods for estimation of Err

2.1 Empirical error

A classifier can be tested on the same data it was trained on. The *empirical* (or *resubstitution*) *error*

$$\overline{\mathrm{err}} = Q(X^{(n)}, X^{(n)}), \quad (5)$$

is typically over-optimistic and has a negative bias that is especially large for learning algorithms with high overfitting. In an extreme case, like that of the nearest neighbor rule, $\overline{\text{err}}$ can be zero even when Err is 1/2.

2.2 Cross-validation

To avoid underestimating the error due to resubstitution, in the k-fold cross-validation (CV), the data set is partitioned into k mutually disjoint subsets called *folds*. For each fold $\mathbf{S}^j, j = 1, \ldots, k$, the classifier is trained on all of the data except $\mathbf{S}^j$ and tested on $\mathbf{S}^j$. The resulting estimator is computed as the average of the error rates over the k folds:

$$Err_{CV\times k} = \frac{1}{k}\sum_{j=1}^{k} Q(\mathbf{S}^j, X^{(n)}\backslash\mathbf{S}^j). \qquad (6)$$

The extreme case of the k-fold CV-based estimator is the *leave-one-out* estimator, $Err_{CV\times n}$. The CV is known to produce an almost unbiased estimate of Err, since for every fold the classifier is trained on almost the complete $X^{(n)}$; however, it often suffers from high variance due to the learning algorithm's instability under small perturbations in the data (Kohavi, 1995). A possible approach to reducing the variance of CV is to average $Err_{CV\times k}$ over several k-fold divisions of the data, giving $Err_{MCV\times k}$.

2.3 Bootstrap and its use for error estimation

The bootstrap method is based on the following idea. A (nonparametric) maximum likelihood estimator of a statistic $\theta(X^{(n)})$ is given by the expectation of θ with respect to the empirical distribution $\hat{F}_n$, which gives a probability mass of $1/n$ to each sample $\mathbf{x}_i$ in $X^{(n)}$ (Efron, 1992). Usually no analytical expression for this expectation exists, as it is in case of $\theta(X^{(n)}) = Err$. However, a Monte-Carlo algorithm allows for numerical evaluation of this expectation by drawing B samples of size n from $\hat{F}_n$. These samples ${X^{(n)}}^*_1, \ldots, {X^{(n)}}^*_B$ are called *bootstrap samples*, and the value $\theta({X^{(n)}}^*_b)$ is called a *bootstrap replication* of θ. A bootstrap estimate of θ is then computed by averaging over the bootstrap replications. The smoothing done by bootstrap typically reduces the discontinuities in the statistic, and thus lowers its variability.

2.3.1 Ordinary bootstrap estimator

The ordinary, or naive, bootstrap estimate of Err, constructed with B replications, is given by

$$Err_{BS} = \frac{1}{B}\sum_{b=1}^{B} Q(X^{(n)}, {X^{(n)}}^*_b), \qquad (7)$$

that is, the classifier is trained on a BS sample and tested on the original data set, and the average over the B samples is computed. Note that the test point $\mathbf{x}_i$ is included in the training set ${X^{(n)}}^*_b$ with probability $1-(1-1/n)^n$, which is approximately .632 for large n. Due to this expected partial resubstitution, one should expect Err_{BS} to be biased downwards.

2.3.2 Leave-one-out bootstrap

Leave-one-out bootstrap (Efron & Tibshirani, 1997) is a "smoothed" version of $Err_{CV\times n}$. To compute it, one draws the bootstrap samples from the empirical distribution of the original sample with the i-th point removed, $X^{(n)}_{(i)}$. This distribution $\hat{F}_{(i)}$ assigns probability $1/(n-1)$ to all $\mathbf{x}_j$, $j \neq i$. Then

$$Err^{(1)}_{BS} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{B}\sum_{b=1}^{B} Q(\mathbf{x}_i, {X^{(n)}_{(i)}}^*_b). \qquad (8)$$

This smoothing reduces the variance. However, in computation of a single term in (8), the training set has an expected $.632n$ distinct data points, as opposed to $Err_{CV\times n}$, which is based on classifiers trained on $n-1$ points. Since, for many classifiers, Err tends to decrease as the size of the training set grows, this fact implies an upward bias.

2.3.3 Hybrid bootstrap and .632+

The error estimate of the general form of a *hybrid bootstrap error estimator* is given by

$$\widehat{Err}^{\lambda} = \lambda Err^{(1)}_{BS} + (1-\lambda)\overline{\text{err}},$$

and a mixing parameter λ is sought that minimizes the bias. Many authors (e.g. (McLachlan, 1992)) report "substantial empirical evidence" favoring $\lambda = 0.632$. Efron also gives a heuristic motivation for this number: a bootstrap sample of size n is expected to be supported by approximately $.632n$ original data points. This choice of λ gives rise to the .632 bootstrap estimator given by $Err_{.632} = .632\,Err^{(1)}_{BS} + .368\overline{\text{err}}$. The intuition behind $Err_{.632}$ suggests compensation for the downward bias of $\overline{\text{err}}$ with the upward bias of $Err^{(1)}_{BS}$. However, for a classifier with high overfitting (such as 1-NN) $\overline{\text{err}} \equiv 0$, and $Err_{.632}$ is downward biased itself.

$Err_{.632+}$, proposed by Efron and Tibshirani in (Efron & Tibshirani, 1997), is a sophisticated estimator that attempts to estimate the amount of overfitting and adjust λ accordingly. The authors define the "no-information" error rate, which is the error rate of the classifier when the data conveys no information for the classifier ($\mathbf{x}$'s and y's independent), and estimate it by

averaging the error rate over all possible permutations of the data and labels. From the no-information error rate estimate $\hat{\gamma}$, the overfitting rate $\widehat{R}$ is derived, and finally the .632+ estimator is obtained as

$$Err_{.632+} = Err_{.632} + (Err_{BS}^{(1)} - \overline{\text{err}}) \frac{.368 \cdot .632 \cdot \widehat{R}}{1 - .368\widehat{R}}.$$

3. Related work

A great deal of work exists on both regression (with the prediction error usually measured in terms of squared loss) and classification (with various loss functions). According to reports in the ML community, cross-validation, and in particular leave-one-out, is the most widely used method for this task. Bootstrap methods are less popular. They are known to have higher biases than CV but lower variances, and consequently show better results in terms of squared error when a large number of trials is performed ((Efron, 1983),(Shao & Tu, 1995)). (Kohavi, 1995) shows, for 10 UCI data sets, that CV-based estimators work better due to the large bias of the bootstrap. However, (Efron & Tibshirani, 1997) obtain the best performance with .632+ for a number of data sets, including some of those used by Kohavi. Therefore, .632+ seems to be the state-of-the-art BS-based estimator.

There have been previous efforts at classification using density estimation for complex, real-life data involving non-numeric features (Hermans et al., 1982), but with no clear means of sampling from the estimated density.

4. Smoothed bootstrap

Constructing a bootstrap replication of the data by sampling with replacement is equivalent to randomly drawing samples from the empirical probability distribution $\hat{F}_n$. Instead, one may estimate the *PDF* of the data, and use the estimated density to draw samples. This is the underlying idea of smoothed bootstrap (Silverman & Young, 1987).

4.1 Kernel-based nonparametric density estimation

We use kernel-based density estimation (Parzen windows), as described, for example, in (Scott, 1992). The *PDF* of a point $\mathbf{x}$, based on the sample $X^{(n)}$, is

$$\hat{f}(\mathbf{x}) = \frac{1}{n|\mathbf{H}|} \sum_{i=1}^{n} K\left(\mathbf{H}^{-1}(\mathbf{x} - \mathbf{x}_i)\right), \qquad (9)$$

where the kernel K is a probability density function, and $\mathbf{H}$ is a matrix which essentially determines the covariance structure of K. It has been suggested in the literature (Fukunaga, 1990) that if the data set is *whitened*[1], one can use the product kernel, which is a product of d unidimensional kernels:

$$\hat{f}(\mathbf{x}) = \frac{1}{n\prod_{j=1}^{d} h_j} \sum_{i=1}^{n} \prod_{j=1}^{d} K\left(\frac{x_j - x_{ij}}{h_j}\right) \quad . \quad (10)$$

To apply a kernel method, two parameters must be set: the kernel function and the bandwidth. The choice of the bandwidth is critical to the success of the method, and a large number of methods for automatic bandwidth selection exist. Intuitively, the optimal bandwidth must obey some smoothness conditions, which can be expressed as bounds on $\int f''$. Our system uses the direct plug-in method, based on kernel-based estimation of the derivatives of f (Wand & Jones, 1995).

The choice of kernel is considered to be much less important for the quality of the density estimate. We use the Epanechnikov kernel $K(x) = 3(x-1)^2/4$, which has been shown to be the optimal kernel for density estimation, in terms of minimizing the mean squared error (Scott, 1992), and also allows simple resampling.

4.2 Sampling from the estimated density

Kernel-based *PDF* estimation is a computationally expensive task. However, to sample from $\hat{f}$, one does not need to compute (9) explicitly. For arbitrary $\mathbf{x}$, the value of $\hat{f}(\mathbf{x})$ is the sum of the contributions of all of the sample points, each weighted by $1/n$. Thus, the result of adding random noise $\mathbf{w}$ drawn from K to a uniformly chosen $\mathbf{x}_i$ will have density $\hat{f}$. Using the Epanechnikov kernel, each component of $\mathbf{w}$ can be generated in expected time $O(1)$, e.g. using the rejection method (Devroye & Győfri, 1985). Therefore, resampling n points in smoothed BS has the same order of magnitude as that in ordinary BS.

5. New estimators

It is evident from both theoretical and empirical results that variance reduction in an estimator of *Err* can be more advantageous than bias correction. In order to reduce the variance, we consider two modifications of known estimators. Both are guided by the need to smooth the discontinuities in Q in order to reduce the variability due to perturbations in the data.

[1]We say that the data is whitened if it is transformed to have a unit covariance matrix.

5.1 Using cloning in BS-based estimators

The first idea is to use cloning where normally an ordinary BS is used. Possible improvement in the expected squared error here is achieved by using a more sophisticated sampling method. Smoothed bootstrap is known (Shao & Tu, 1995) to outperform the ordinary BS for statistics that are sensitive to local properties of the underlying distribution. Our intuition is that this holds for Q in the case of many classifiers. In particular, since the .632+ estimator is believed to be the best choice of BS estimator, it can be modified to use cloning. We denote BS-type estimators that use cloning instead of ordinary BS by appending an asterisk, e.g., $Err_{.632+}*$. From the previous section, we know that this change does not increase the computational intensity of the procedure.

5.2 Bootstrapped cross-validation

An alternative approach is to smooth the estimation results by bootstrapping the data in each stage of the computation. Here, increased computational effort should allow us to get a more stable estimate. To smooth the discontinuities in the results of CV caused by perturbations in the training and test sets, CV is performed on each BS sample (or clone) of $X^{(n)}$:

$$Err_{CV*\times k} \;=\; \frac{1}{B}\sum_{b=1}^{B} Err_{CV\times k}\left(X^{(n)*}_{b}\right). \qquad (11)$$

The computational cost is similar to that of computing $Err_{MCV\times k}$ with B trials.

6. Simulation study with synthetic data

In order to test the plausibility of the proposed estimators, we followed the experimental framework of (Efron & Tibshirani, 1997), described in Table 1. In all cases, there are two classes. In experiments 2 and 4, the distribution of the data is independent of the class label, so that any classifier has an expected accuracy of 1/2. The true error Err was approximated by testing each trained classifier on 20,000 data points drawn from the same distribution (a validation set). Both the training data and the validation data are balanced: the same amount of data is assigned to each class. In all of the experiments, the classifiers are 1- and 3-NN, Fisher's Linear Discriminant Functions (LDF), and Support Vector Machines (SVMs) with radial basis function kernels. Whenever bootstrapping was used, 100 BS samples were generated. In each trial, the same data sets and the same bootstrap samples and clones were used for all of the estimators (to reduce the variability in the differences due to random factors).

Table 1. Setting of the simulation with synthetic data described in Section 6. All data sets have two classes. In 1 and 3 the classes have distinct means. In 2 and 4 the distribution is independent of the class label. The size of each data set is given by n.

#	Distribution	n	Trials
1	$N\left((\pm 1,0,0,0,0)^T, \mathbf{I}_5\right)$	14	100
2	$N\left(\mathbf{0}_5, \mathbf{I}_5\right)$	14	100
3	$N\left((\pm .5,0)^T, \mathbf{I}_2\right)$	20	100
4	$N\left(\mathbf{0}_2, \mathbf{I}_2\right)$	20	100
5	$N\left(\mathbf{0}_{10}, \mathbf{I}_{10}\right)$ vs. $\prod_{j=1}^{10} N\left(\sqrt{j}/2, 1/j\right)$	100	50

In Table 2 we compare the best of the new methods to the best of the old ones. To determine the statistical significance of these results, we perform hypothesis testing for each classifier/data set. The null hypothesis is that the expected squared errors of the two methods are equal. We want to reject this hypothesis in favor of the alternative that the expected error of the new method is smaller. Since all of the estimators are applied to the same data set in each trial, the obtained values are dependent, and the paired single-tailed test for comparison of the means is appropriate (Papoulis, 1991). The sample of interest here is $d = (\widehat{Err}_{old} - Err)^2 - (\widehat{Err}_{new} - Err)^2$. The significance α of the test corresponds to the z-value $z_\alpha = \mu_d/(\sigma_d/\sqrt{N})$, where N is the number of trials, μ_d is the sample mean, and σ_d is the sample variance of the differences. These values appear in the last three columns of Table 2.

In 17 out of 20 cases, one of the new estimators was significantly ($\alpha \leq .01$) better in terms of RMSE than any of the old ones. While our estimators are sometimes more biased, their variance is noticeably smaller, as hoped. We note the particularly good performance of $Err_{.632+}*$, which in 12 cases was better than $Err_{.632+}$, and of $Err_{CV*\times 5}*$, which outperformed all versions of $Err_{CV\times k}$ and $Err_{CV*\times k}$ in 16 cases.

7. Cloning real-life data sets

Generally, the data space can be written as

$$\mathcal{D} \;\subseteq\; \mathbb{R}^{m_c} \times \mathbb{Z}^{m_o} \times \mathbb{A}_1 \times \ldots \times \mathbb{A}_n \quad , \qquad (12)$$

where m_c, m_o and m_n are the numbers of continuous, discrete and nominal attributes, respectively. $\mathbb{A}_i$ denotes a nominal domain (possibly different for each i). Kernel density estimation as in (9) may not be applicable in the general case. Below we propose ways to overcome some of the difficulties.

Table 2. Results on synthetic data. Better RMSE shown in bold.

Data set	Classifier	Err	$\widetilde{Err}_{\mathbf{new}}$	Mean	STD	RMSE	$\widetilde{Err}_{\mathbf{old}}$	Mean	STD	RMSE	$\sqrt{\mu_{\mathbf{d}}}$	$\sigma_{\mathbf{d}}$	α
1	LDF	.262	$Err_{CV*\times n*}$	.2668	.0626	.0767	Err_{BS}	.2537	.0662	.0776	.0121	.006	.37
	1-NN	.2785	$Err_{.632*}$	.217	.0707	**.0863**	$Err_{.632+}$	.256	.1139	.0999	.05	.0105	.01
	3-NN	.2462	$Err_{CV*\times 5*}$	.24	.0651	**.0598**	Err_{BS}	.199	.0745	.0815	.055	.0061	10^{-13}
	SVM	.2367	$Err_{CV*\times 5*}$	.247	.0632	**.0671**	$Err_{.632}$	.146	.0626	.1108	.055	.0061	10^{-13}
2	LDF	.5	$Err_{CV*\times n*}$	.511	.0729	.0731	$Err^{(1)}_{BS}$	.507	.0705	.0703	-.02	.0024	.15
	1-NN	.5	$Err_{CV*\times n*}$	.535	.0729	**.0803**	$Err^{(1)}_{BS}$	.54	.1051	.1118	.078	.0137	10^{-10}
	3-NN	.5	$Err_{.632+*}$	.4585	.0526	**.0668**	$Err_{.632+}$	.4588	.066	.0772	.039	.0066	.01
	SVM	.5	$Err_{.632+*}$	.412	.0472	.1005	$Err_{.632+}$	.4173	.0562	.1003	-.0057	.0084	.48
3	LDF	.35	$Err_{CV*\times 5*}$	.35	.0794	**.0813**	Err_{BS}	.313	.091	.0996	.0575	.0078	10^{-10}
	1-NN	.414	$Err_{CV*\times 5*}$	.363	.0646	**.0737**	$Err_{.632+}$	.354	.0834	.1002	.068	.0106	10^{-10}
	3-NN	.392	$Err_{CV*\times 5*}$	.361	.0679	**.0689**	$Err_{.632+}$	.391	.0848	.0816	.044	.0067	10^{-5}
	SVM	.356	$Err_{CV*\times 5*}$	.368	.072	**.0776**	Err_{BS}	.32	.0886	.0959	.056	.01	10^{-6}
4	LDF	.5	$Err_{.632+*}$	.471	.0771	**.0828**	$Err_{.632+}$	.468	.086	.0924	.041	.0026	10^{-19}
	1-NN	.5	$Err^{(1)}_{BS*}$	.527	.0572	**.0633**	$Err^{(1)}_{BS}$	.533	.1005	.1061	.085	.0126	10^{-15}
	3-NN	.5	$Err_{.632+*}$	.458	.0457	**.0625**	$Err_{.632+}$	.457	.0622	.0763	.0438	.0065	10^{-5}
	SVM	.5	$Err_{.632+*}$	.485	.0511	**.0537**	$Err_{.632+}$	.484	.0614	.0641	.035	.0044	.0002
5	LDF	.018	$Err_{CV*\times n*}$	.009	.0084	.0148	$Err_{.632}$	.0085	.0076	.0147	.0017	$5 \cdot 10^{-5}$	.33
	1-NN	.022	$Err_{CV*\times n*}$	.016	.0094	**.0123**	$Err^{(1)}_{BS}$	.014	.01	.0141	.007	$3 \cdot 10^{-5}$	0
	3-NN	.027	$Err_{CV*\times 10*}$	.0171	.0073	**.013**	$Err^{(1)}_{BS}$	.0171	.0106	.0156	.009	.0001	10^{-6}
	SVM	.0036	$Err_{CV*\times n*}$	.002	.0018	**.0029**	$Err_{.632}$	.001	.0018	.0033	.0015	$\approx 10^{-6}$	10^{-7}

7.1 Cloning data with boundaries

If the actual density is discontinuous at some point, then the regular kernel estimate is obviously very inaccurate. Such a discontinuity occurs at an endpoint of an interval outside which the density vanishes. Near discontinuities, (9) becomes inaccurate since it overestimates the density outside the boundaries. Most boundary correction techniques mentioned in the literature involve kernel functions with negative values (Scott, 1992). Such a kernel is no longer a *PDF*, making sampling problematic. In addition, different *PDF*s behave in different ways at boundaries. No general solution to the task of finding the "right" type of boundary kernel to fit the specific behavior of the estimated *PDF* is known to date.

To generate boundary-obeying values, we use the following method. Let D be the set of admissible values ("inside" the boundaries). We force the kernel to be zero outside D and divide by $s = \int_{x \in D} K(x)dx$ in order for the kernel to integrate to unity.

Sampling from this boundary corrected kernel is straightforward. We repeat the algorithm for sampling from $\hat{f}$ until the result falls within D. The probability of failure of a single trial is $1 - s$. As long as we do not apply the algorithm to an isolated point of the distribution, we have a positive lower bound on s, and an upper bound on the expected time before a valid point is generated.

7.2 Cloning data with discrete attributes

Discrete feature values come from a domain that can be mapped to $\mathbb{Z}$, with a corresponding metric and order. Applying kernel functions of a continuous variable to a discrete attribute is unnatural and, in our experience, leads to poor results when applied to resampling. Instead we follow (Aitchison & Aitken, 1976) and use a family of discrete kernels:

$$K^d(x, x_i) = K^d(h, x, x_i) \;=\; \frac{h^{\|x - x_i\|^2}}{\sum_{k=1}^{T} h^{\|x - x_i\|^2}}. \quad (13)$$

$K^d(x, x_i)$ is centered on x_i and assigns a weight to x that is proportional to its distance from x_i; the rate at which it drops depends on h. Since we have no theoretically solid method of choosing h, the following heuristic was used to avoid oversmoothing. For each x, we would like to keep 95% of the probability mass assigned to x by the empirical distribution $\hat{F}_n$ close to x. We take the standard deviation of the values of x as a measure of spread, and set $h = .05^{1/\sigma_x^2}$. The discrete attributes are sampled similarly to the continuous ones, with the substitution of K^d for K.

7.3 Cloning data with nominal attributes

Our choice of classifiers in the experiments reported in this paper dictated the use of data sets with no nominal attributes. For completeness, we present the outline of our method for resampling nominal attributes; a detailed description will appear in a fuller version.

The support space of a *nominal* (sometimes called categorical) attribute is a non-metric finite unordered space $\mathbb{A}$. Smoothing of the marginal distribution of nominal attributes could introduce impossible data points. We feel that this hazard overwhelms the possible benefit from enriching the cloned data, and choose not to perform smoothing directly on a nominal domain. First, we sample the values of the numeric

and preceding nominal attributes from the estimated marginal distributions. Next, we estimate the joint distribution of the attribute in question and the preceding attributes. Using Bayes rule, we can then estimate of the conditional distribution of the attribute given the previously determined prefix of the point.

Table 5. RMSE performance of multiple CVs versus proposed estimators. '-' indicates that $Err_{MCV\times k}$ is worse, '+' that it is better, and '?' that it is not different with significance below .1

	$Err_{.632+}*$			$Err_{CV*\times 5}*$			$Err_{CV*\times n}*$		
	-	+	?	-	+	?	-	+	?
$Err_{MCV\times n}$	8	0	0	6	1	1	4	2	2
$Err_{MCV\times 10}$	7	1	0	5	2	1	4	2	2
$Err_{MCV\times 5}$	7	0	1	6	2	0	4	2	2

8. Simulation study with data from UCI machine learning repository

We used the Wisconsin Breast Cancer, Vehicle (both used in (Efron & Tibshirani, 1997)) and Pima Indians Diabetes data sets. For each data set, we chose classifiers reported to work well on that domain. Table 3 describes the simulation. In each trial, a small data subset of size n was chosen as the input for the experiments. A classifier was then trained on this $X^{(n)}$ and tested on the remaining larger subset to estimate the conditional true error. This value was then compared to the predictions of different estimators, to which only $X^{(n)}$ was made available.

As with the synthetic data, in the majority of the cases one of the new estimators performed significantly better in terms of RMSE than any of the old estimators (Table 6). In most cases, the winning estimator had the lowest variance. In Table 4 we show that the cloned version of $Err_{.632+}$ outperformed its traditional counterpart in 6 out of 8 cases with significance below .05, and in a seventh case with significance .15. We also show (Table 5) that running the standard CV multiple times provides only a partial solution, as suggested in (Salzberg, 1997), in exchange for a significant increase in computational cost. $Err_{MCV\times k}$ for various k are outperformed by $Err_{.632+}*$ and by $Err_{CV*\times 5}*$.

9. Conclusions and future work

The contribution of this paper is two-fold. First, we propose new smoothed bootstrap schemes to clone data for better error estimation for supervised learning algorithms, and report on extensive simulations with both synthetic and real data. We present an argument, based on Markov's inequality, in favor of the root mean squared error as a measure of estimator quality, and show that in terms of RMSE, cloning improves the quality of the bootstrap-based estimators. Our results show a high correlation of this improvement with reduction of variance in the estimator. In particular, in 7 out of 8 experiments on the UCI data sets (Table 4), and in 12 of 20 on synthetic data sets, .632+ is outperformed by its cloning-based counterpart. In general, while no estimator wins in all cases, in 24 of 28 cases, including 7 of 8 experiments on UCI data , one of the new estimators achieved lower RMSE than any of the old ones. Based on these observations, we recommend the novel cloning-based .632+* estimator. Cloned 5-fold CV was the best estimator for synthetic data, but showed inferior performance on UCI sets. We intend to investigate what properties of the data affect the relative performance of these two estimators.

Second, we propose a suite of algorithms that can statistically clone real data from "complex" domains in which data points have several types of dependent attributes (real, integer, bounded and nominal).

We should emphasize that the use of statistical cloning techniques comes with increased computation. However, with the proposed choice of kernel, the increase in computation needed for cloning is within a constant factor relative to ordinary bootstrap. Bootstrapped cross-validation involves $O(n^2)$ additional applications of the learning algorithm, which is expensive. However, in the spirit of work done recently to allow effective leave-one-out estimation "customized" for specific classifiers, such as SVMs (Joachims, 2000), the design of an efficient cloning technique for particular classifiers is an interesting problem.

One of the advantages of bootstrap over cross-validation is the ability to provide confidence intervals for the estimated value. We are working on expanding our system to include this capability.

Acknowledgements

We would like to thank Karen Livescu for proofreading and providing helpful comments, and Anat Sakov for valuable discussions. We thank the anonymous reviewers for their valuable comments. The software used in the experiments includes G. Cawley's SVM package and Netlab.

References

Aitchison, J., & Aitken, C. G. G. (1976). Multivariate binary discrimination by the kernel method. *Biometrika*, *63*, 413–420.

Blake, C. L., & Merz, C. J. (1998). UCI

Table 3. Settings of the simulation study with UCI data sets, Section 8.

Data set	Size	n	Number of features	Type of features	number of classes	Number of trials
Breast cancer	683	36	9	all integer, bounded	2	150
Vehicle	846	100	18	all continuous, bounded	4	150
Pima	768	60	8	6 integer, 2 continuous, bounded	2	100

Table 4. RMSE of .632+ with and without cloning. The better estimator for each test is shown in bold.

	Breast cancer				Pima indians		Vehicle	
	1-NN	3-NN	SVM linear	SVM rbf	17-NN	SVM rbf	1-NN	SVM rbf
$Err_{.632+}$	.0275	.0307	.0276	.0230	**.0301**	.0624	.076	.0567
$Err_{.632+*}$	**.0269**	**.0297**	**.0271**	**.0225**	.0324	**.0584**	**.0417**	**.0438**
significance	.15	.0093	.0064	.0262	.0463	.0355	10^{-11}	10^{-6}

Table 6. Results on UCI data sets, experiments in Section 8.

Data set	Classifier	Err	$\widehat{Err}_{\mathbf{new}}$	Mean	STD	RMSE	$\widehat{Err}_{\mathbf{old}}$	Mean	STD	RMSE	$\sqrt{\mu_{\mathbf{d}}}$	$\sigma_{\mathbf{d}}$	α
Breast cancer	1-NN	.052	$Err_{.632+*}$	.036	.0259	.0269	$Err_{.632}$	.035	.0255	.0271	.0025	.0002	.3
	3-NN	.045	$Err_{CV*\times 5*}$	.037	.0225	.0248	Err_{BS}	.036	.0234	.0259	.007	.0002	.002
	SVM linear	.04	$Err_{CV*\times 5*}$	.028	.0203	.0254	Err_{BS}	.028	.0206	.0259	.005	.0002	.058
	SVM rbf	.04	$Err_{.632*}$	.027	.0199	.022	$Err_{.632}$	.028	.0209	.0222	.003	.0001	.09
Pima Indians	17-NN	.33	$Err_{CV*\times 5}$	.3135	.0235	.0284	$Err_{MCV\times 10}$	.333	.0236	.025	-.013	.001	.019
	SVM rbf	.294	$Err_{.632+*}$	.302	.0567	.0584	$Err_{.632+}$	.286	.0596	.0624	.022	.003	.0355
Vehicle	1-NN	.438	$Err_{CV*\times 5*}$	.447	.0278	.0338	$Err_{BS}^{(1)}$	.464	.0431	.0537	.042	.004	10^{-5}
	SVM rbf	.351	$Err_{CV*\times 5*}$	.334	.0264	.0399	$Err_{.632+}$	.321	.0384	.0567	.04	.003	10^{-8}

repository of machine learning databases. [http://www.ics.uci.edu/∼mlearn/MLRepository.html].

Devroye, L., & Győfri, L. (1985). *Nonparametric density estimation: The l_1 view*. New York: Wiley.

Efron, B. (1979). Bootstrap methods: another look at the jackknife. *Annals of Statistics*, *7*, 1–26.

Efron, B. (1983). Estimating the error rate of a prediction rule: improvements on cross-validation. *Journal of the American Statistical Association*, *78*, 316–331.

Efron, B. (1992). Jacknife-after-bootstrap standard errors and influence functions *(with discussion)*. *Journal of Royal Statistical Society*, *54*, 83–127.

Efron, B., & Tibshirani, R. (1997). Improvements on cross-validtion: The .632+ bootstrap method. *Journal of the American Statistical Association*, *92*, 548–560.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition, second edition*. Boston, MA: Academic Press.

Hermans, J., Habbema, J., Kasanmoentalib, T., & Raatgeven, J. (1982). Manual for the alloc80 discriminant analysis program. Leiden, Netherlands.

Joachims, T. (2000). Estimating the generalization performance of an SVM efficiently. *Proc. 17th International Conf. on Machine Learning* (pp. 431–438). Morgan Kaufmann, San Francisco, CA.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* (pp. 1137–1145). San Mateo: Morgan Kaufmann.

McLachlan, G. J. (1992). *Discriminant analysis and statistical pattern recognition*, chapter 10, 337–377. New York: Wiley.

Papoulis, A. (1991). *Probability, random variables, and stochastic processes*. New York: McGrow Hill. 3rd edition.

Salzberg, S. (1997). On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, *1*, 317–327.

Scott, D. W. (1992). *Multivariate density estimation. theory, practice and visualization*. New York: Wiley.

Shao, J., & Tu, D. (1995). *The jacknife and bootstrap*. New York: Springer-Verlag.

Silverman, B. W., & Young, G. A. (1987). The bootstrap: to smooth or not to smooth? *Biometrika*, *74*, 469–479.

Wand, M. P., & Jones, M. C. (1995). *Kernel smoothing*. London: Chapman & Hall.