Proceedings of the Seventeenth International Conference on Machine Learning. Stanford, 2000.

Predicting the Generalization Performance of Cross Validatory Model Selection Criteria

Tobias Scheffer SCHEFFER@IWS.CS.UNI-MAGDEBURG.DE
University of Magdeburg, FIN/IWS, Universitätsplatz 2, 39106 Magdeburg, Germany

Abstract

We conduct an average-case analysis of the generalization error rate of holdout testing and n-fold cross validation "wrappers" for model selection. Unlike previous approaches, we do not rely on *worst-case* bounds that hold for all possible learning problems. Instead, we study the behavior of a learning algorithm with a cross-validation wrapper *for a given problem*, taking properties of the problem (that can be estimated using the sample) into account. We have to pay for this (and the efficiency of our solution) by having to make some approximations. Experiments show that our analysis can nevertheless predict the behavior of cross validation wrappers fairly accurately.

1. Introduction

Cross validation is an extremely general tool for estimating the error probability of a hypothesis. As such, it enables us to assess the error rates of hypotheses learned from various distinct model classes (*e.g.,* neural networks with exactly i hidden units, or decision trees with i nodes) and to select the model class (*i.e.,* the number of hidden units or the number of decision tree nodes) that gives us the highest generalization performance. Several variants of cross validation exist; see, *e.g.,* Toussaint (1974), for an overview. We study holdout-testing (also referred to as one-shot training and test), and n-fold cross validation (Mosier, 1951) for finite model classes. "Leave-one-out" (Lachenbruch, 1967) is a special case of n-fold cross validation (n equals the sample size).

Unfortunately, cross validation is not very well understood. Until recently, it has not even been shown that n-fold cross validation often does in fact better than holdout testing (Blum et al., 1999). Confidence bounds of n-fold cross validation estimates rely on crude independence assumptions; known general analyses of cross validation (Kearns & Ron, 1997; Anthony & Holden, 1998) are *worst-case* analyses in the spirit of PAC- and VC-theory. That is, they are distribution-free and hold even if the learner has the worst possible learning bias for the given problem, but they describe the *actual* behavior of a learning algorithm *for a given problem* only very inaccurately.

We want to conduct a computationally efficient *average-case* analysis that predicts the behavior of a classification algorithm with a cross validation wrapper more accurately *for a given problem* (and takes estimatable properties of the problem into account), even if we have to pay for this by having to make reasonable approximations. Our principle objective is to be able to predict the error rate as accurately as possible, which we see as one possible indicator of how well we understand cross validation, and which would make the analysis an interesting heuristic for determining parameters of the wrapper (*e.g.,* the split ratio).

After a quick clarification of some notational details in this section, we will, in Section 2, briefly recapture the average-case analysis of exhaustive learning algorithms of Scheffer and Joachims (1999) on which our analysis of cross validation is based. We will present this analysis in Section 3. In Section 4, we will study empirically how accurately our analysis predicts the behavior of holdout testing and n-fold cross validation wrappers for model selection using both randomly drawn boolean functions and some problems from the UCI repository. In Section 5, we briefly discuss some related average-case approaches.

Preliminaries. A *classification problem* is given by an (unknown) density $p(x,y)$. A *hypothesis* $h: X \rightarrow Y$ incurs a *true or generalization error rate* $\epsilon(h) = \int \sum_y \ell(h(x),y)p(x,y)dx$ (where ℓ is the zero-one loss function). Let the *sample* S be a sequence of m independent examples, governed by $(p(x,y))^m$. The *empirical or training set error rate* is $e(h) = \frac{1}{m}\sum_{(x,y)\in S}\ell(h(x),y)$. Let $\langle H_1,\ldots,H_k\rangle$ be a collection of *finite model classes.* For instance, H_i could

$$p(\epsilon(h_i^L) = \epsilon | H_i, m) \quad = \quad \frac{\sum_e P(h_\epsilon \in H_i^* | e(h_\epsilon) = e, m) P(e(h_\epsilon) = e | \epsilon, m) \pi(\epsilon | H_i)}{\int_{\epsilon'} \sum_e P(h_{\epsilon'} \in H_i^* | e(h_{\epsilon'}) = e, m) P(e(h_{\epsilon'}) = e | \epsilon', m) \pi(\epsilon' | H_i) d\epsilon'} \tag{1}$$

$$\text{with} \quad P(h_\epsilon \in H_i^* | e(h_\epsilon) = e, m) = \prod_{\epsilon'} P(e(h) \geq e | \epsilon', m)^{|H_i| \pi(\epsilon' | H_i)}. \tag{2}$$

contain all decision trees with i leaf nodes. A *learner* accepts a sample S and a model class H_i (a set of potentially available hypotheses) and returns a hypothesis $h_i^L \in H_i$. For our analysis, we assume that, within the given model class, the learner minimizes the empirical error rate. When more than one empirical error minimizing hypothesis exists, we assume that the learner breaks ties by drawing at random. We define the *model selection problem* as the search for the number i that minimizes the expected resulting generalization error $E(\epsilon(h_i^L)|m, H_i, p(x, y))$.

One solution to the model selection problem is to "wrap" a cross validation layer around the learner. We study holdout-testing and n-fold cross validation wrappers. In the holdout-testing setting, the sample is split randomly into training and holdout set according to the *split ratio* (which specifies the fraction of the sample used for training). In the n-fold cross validation setting, the sample is split into n parts, $n-1$ of which form the training set. After the learning algorithm has returned an empirical error minimizing hypothesis h_i^L for each model class H_i, the hypotheses are assessed on the holdout data (this is repeated n times in the n-fold setting). The model class with the least (averaged) holdout error is then selected as H_* (ties are broken in favor of the model class with the smaller index). The whole sample is now used to minimize the empirical error within H_* which yields the final hypothesis h_*^L that is returned by the wrapper.

2. Error Rate for a Fixed Model Class

In this Section, we summarize the expected error analysis (Scheffer & Joachims, 1999; Scheffer, 1999) on which our analysis of cross validation is based.

Let us assume that the model class H_i is fixed and that our learner minimizes the *empirical* error within H_i (breaking ties by drawing at random, when the hypothesis with least empirical error rate is not uniquely determined) which will result in a hypothesis h_i^L. Let us assume that the sample size m is fixed and given in advance, whereas the sample S itself is a random variable, governed by the distribution $(p(x, y))^m$. The quantity which we are interested in is $p(\epsilon(h_i^L)|H_i, m, p(x, y))$, the density (over all samples) that governs the generalization error of the empirical error minimizing hypothesis h_i^L (since $p(x, y)$ is arbitrary but fixed throughout the analysis, we simply write $p(\epsilon(h_i^L)|H_i, m)$). When we integrate over this distribution, we obtain $E(\epsilon(h_i^L)|H_i, m) = \int \epsilon \cdot p(\epsilon(h_i^L) = \epsilon | H_i, m) d\epsilon$, the expected (over all samples) *generalization* error of the hypothesis h_i^L that minimizes the *empirical* error rate.

In order to determine $p(\epsilon(h_i^L)|H_i, m)$ we need to determine the probability that a hypothesis h with generalization error $\epsilon(h)$ achieves the lowest empirical error in H_i (for all values of $\epsilon(h)$). For this we factorize the empirical error $e(h)$. We can easily determine the chance that a hypothesis h achieves an empirical error of $e(h)$, given $\epsilon(h)$: this probability is governed by the binomial distribution which we write as $B[m, \epsilon(h)](e(h))$. Under the assumption that – given the true error rates – the empirical error rates of distinct hypotheses are independent $(P(e(h_1)|e(h_2), \epsilon(h_1)) = P(e(h_1)|\epsilon(h_1))$, we can determine the chance that $e(h)$ is the *least* empirical error by calculating the chance that no other hypothesis achieves a lower empirical error.

This assumption can clearly be violated by some problems. In principle, we could weaken this assumption by considering the joint distribution of empirical error rates of k hypotheses (the assumption is removed for $k = |H_i|$). Later in this paper, we will discuss a density π that we have to estimate from the data. Without this independence assumption, we would have to estimate a higher dimensional density which would require very large samples and the analysis would be much less efficient (for $k = H_i$ the computational expense would be outrageous).

We also need to know *how many* hypotheses with an error rate of ϵ there are in H_i. Intuitively, when almost all hypotheses incur a generalization error of .01, then the chance that one of them incurs the least empirical error and is returned as h_i^L is much bigger than it was when there were only few such hypotheses. We write the density of error rates in H_i as $\pi(\epsilon|H_i)$. This density quantifies the proportion of hypotheses in H_i with a true error rate of ϵ. If we would draw a hypothesis at random, $\pi(\epsilon|H_i)$ would be the chance that it would incur a generalization error rate of ϵ.

These considerations result in Equations 1 and 2 (a slightly simplified version of Equation 2 of Scheffer and Joachims (1999)) which quantifies the exact (under the mentioned independence assumption) density (over all samples, for a fixed problem $p(x,y)$) that governs $\epsilon(h_i^L)$. Equations 1 and 2 depend on three quantities: the number of hypotheses in $|H_i|$ (which is usually known), the sample size m (which is known), and the density of error rates in H_i, $\pi(\epsilon|H_i)$. Note that, for Equation 1 to give us the expected error $\epsilon(h_i^L)$, it is not necessary to actually run the learner and determine $e(h_i^L)$. The solution assumes that the learning algorithm will always find the empirical error minimizing hypothesis (which cannot be guaranteed for greedy learners). A modified version of this solution (Scheffer, 2000a) quantifies the generalization error of greedy learners but requires the empirical error of the returned hypothesis as additional input.

Sample size m and $|H_i|$ suffice to specify a Chernoff-style bound such that the generalization error rate lies, with high probability, between 0 and this bound. It is impossible to determine where in this interval the error rate lies (Kearns et al., 1997). Additionally given the density $\pi(\epsilon|H_i)$, however, we can determine the *actual* density that governs the generalization error, and thereby also the expected generalization error.

Estimating $\pi(\epsilon|H_i)$. In order to implement our analysis, we need to estimate $\pi(\epsilon|H_i)$ from the data. We can estimate the empirical counterpart $\pi(e|H_i)$ (the distribution of empirical error rates of hypotheses in H_i with respect to the sample S) which we can record when H_i is known and a sample S is available. We can obtain $\pi(e|H_i)$ by repeatedly drawing hypotheses from H_i under uniform distribution, or by conducting a Markov random walk in the hypothesis space with the uniform distribution as stationary distribution (Gilks et al., 1995). While the general problem of estimating densities is very hard, the situation is not quite as bad in our special case. Like $\pi(\epsilon|H_i)$, $\pi(e|H_i)$ is one-dimensional, but is is furthermore discrete since there are only $m+1$ possible empirical error rates when m is the sample size. In order to estimate each individual probability of $\pi(e|H_i)$ up to a fixed small error with high confidence, we only need need to draw $O(\log m)$ hypotheses. There are rare cases in which input of a $\pi(\epsilon|H_i)$ with a small error into Equation 1 can result in an inaccurate output of $\epsilon(h_i^L)$ from Equation 1. In this theoretical worst case, estimating π sufficiently accurately can be as difficult as running a learning algorithm; see Scheffer (1999) for a more detailed discussion. Recently, Langford and McAllester (2000) have shown that this estimate converges to the true density when m grows, even if m grows parallel to $|H_i|$ and $\frac{\log|H_i|}{m}$ does not vanish. Replacing the density $\pi(\epsilon|H_i)$ by the distribution $\pi(e|H_i)$, also turns all the integrals into sums that can easily be evaluated.

3. Expected Error Rate of Cross Validation

3.1 Training and Test

Suppose that our algorithm splits the sample S into a training set S' (of size m') and a holdout set S'' (of size m''). For each model class H_i in the available collection of model classes $\langle H_1, \ldots, H_k \rangle$, the wrapper invokes the learner and receives a hypothesis h_i^L from the set of hypotheses with least empirical error rate. Note that the sample S' is a random variable because it is drawn at random under the distribution $(p(x,y))^{m'}$. Which hypothesis h_i^L the learner will return depends on the randomly drawn sample, and therefore the error rate of these hypotheses is a random variable, too.

Let us first determine the distribution that governs the holdout error of h_i^L, $p(e_{S''}(h_i^L)|H_i, m', m'')$ (over all samples S' and S''). For this, we have to factorize the true error of h_i^L, $\epsilon(h_i^L)$ (we do this in Equation 3). We can factorize $\epsilon(h_i^L)$ by integrating over all possible values ϵ and weighting ϵ according to the density that governs $\epsilon(h_i^L)$ (the true error of the empirical error minimizing hypothesis h_i^L). Note that we have already determined this density, $p(\epsilon(h_i^L)|H_i, m')$ in Equation 1. Given the true error, the holdout error is governed by the binomial distribution (Equation 4). Intuitively, Equation 4 says that the holdout error of h_i^L is governed by the binomial distribution with the true error $\epsilon(h_i^L)$ as mean value, where the true error $\epsilon(h_i^L)$ is a random variable, too, governed by the density $p(\epsilon(h_i^L)|H_i, m')$ specified in Equation 1.

$$
\begin{aligned}
&P(e_{S''}(h_i^L) = e|H_i, m', m'') && (3)\\
&= \int P(e_{S''}(h_i^L) = e|\epsilon(h_i^L)) p(\epsilon(h_i^L) = \epsilon|H_i, m') d\epsilon \\
&= \int B[m'', \epsilon](e) p(\epsilon(h_i^L) = \epsilon|H_i, m') d\epsilon. && (4)
\end{aligned}
$$

Now let us determine the chance that the wrapper selects model class H_i as H_*. This happens when no model class H_j, achieves a lower holdout error. In Equation 5, we first factorize the holdout error e of h_i^L (we have already described the distribution which governs this holdout error in Equation 4) and then say that the wrapper selects H_i only if the learner finds a hypothesis h_j^L with strictly greater holdout error than e in all previous model classes H_j and greater or equal holdout error in all succeeding model classes (*i.e.,* we assume that the wrapper favors small indices). The density governing the holdout error of the hypotheses h_j^L is described by Equation 4, too. In Equation 5 we assume the holdout error rates of hypotheses h_i and $h_{i'}$ from distinct model classes i and i' to be independent. This assumption seems not unreasonable since such hypotheses come from distinct model classes.

$$
\begin{aligned}
&P(H_i = H_*|m', m'') \qquad (5)\\
&= \sum_e P(e_{S''}(h_i^L) = e|H_i, m', m'')\\
&\quad \times \prod_{j<i} P(e_{S''}(h_j^L) > e|H_j, m', m'')\\
&\quad \times \prod_{j>i} P(e_{S''}(h_j^L) \geq e|H_j, m', m'')
\end{aligned}
$$

Let us finally describe the distribution that governs the generalization error of h_*^L, the hypothesis returned by the wrapper. Once the wrapper has settled for a model class H_*, it will invoke the learner again for that model class and provide it with the whole sample $S = S' \cup S''$. In Equation 6, we factorize the index i of the model class selected by the wrapper. We weight the density that governs the generalization error of h_i^L (based on the whole sample $S' \cup S''$) (described by Equation 1) by the chance that the wrapper selects H_i as H_* (described in Equation 5). This results in the density that governs the true error of h_*^L.

$$
\begin{aligned}
&p(\epsilon(h_*^L) = \epsilon|H_*, m', m'')\\
&= \sum_{i=1}^{k} p(\epsilon(h_i^L) = \epsilon|H_i, m' + m'') \qquad (6)\\
&\qquad P(H_i = H_*|m', m'')
\end{aligned}
$$

We can determine the expectation by integrating over this distribution.

3.2 n-Fold Cross Validation

Equation 5 describes the chance that model class H_i is selected because it incurs a low holdout error rate. When we use n-fold cross validation instead of holdout testing, we need to tweak this equation. In this setting, n estimates based on distinct holdout sets of size $\frac{m}{n}$ are averaged and yield the cross validation error rate. To describe this process mathematically, we have to assume that these n estimates are statistically independent. Although the n holdout sets are in fact independent (under the standard assumption that the examples are drawn independently and identically distributed), the n holdout error rates are not, in case $n > 2$. The reason is that the n training sets are "similar"; any two training sets share a fraction of $\frac{n-2}{n-1}$ of their examples. This is a standard assumption (*e.g.*, the calculation of p values which is required to compare cross validation error rates of different learning techniques relies on it), although it is obvious that it can be violated; see Dietterich (1998) for a discussion.

Let us study the probability that model class H_i is selected as H_*. This happens when it incurs the least cross validation error. In Equation 7 we factorize the cross validation error $e_{CV}(H_i)$ of H_i and say that H_i is selected when all previous model classes lead to strictly higher cross validation errors and no succeeding model class incurs a lower cross validation error (again we break ties in favor of small indices).

$$
\begin{aligned}
&P(H_i = H_*|m, n) \qquad (7)\\
&= \sum_e P(e_{CV}(H_i) = e|H_i, m, n)\\
&\quad \times \prod_{j<i} P(e_{CV}(H_j) > e|H_i, m, n)\\
&\quad \times \prod_{j>i} P(e_{CV}(H_j) \geq e|H_i, m, n)
\end{aligned}
$$

Now we need to discuss the distribution that governs the n-fold cross validation error $e_{CV}(H_i)$ (analogously to the holdout error $e_{S''}(h_i^L)$ that we described in Equation 3). In Equation 8, we factorize the holdout error e' of the first cross validation fold. Equation 8 says that the n-fold cross validation error of H_i is e just in case that the holdout error on the first fold is e' and the $n-1$-fold cross validation error on the remaining folds is $e - e'$.

$$
\begin{aligned}
&P(e_{CV}(H_i) = e|H_i, m, n) \qquad (8)\\
&= \sum_{e'} P\left(e_{S''}(h_i^L) = e' \middle| H_i, \frac{m(n-1)}{n}, \frac{m}{n}\right)\\
&\qquad P(e_{CV}(H_i) = e - e'|H_i, m, n-1)
\end{aligned}
$$

Equation 8 is recursive in n and can be evaluated in time linear in n. In order to anchor Equation 8, we need to add that the 0-fold error is always 0: $P(e_{CV}(H_i) = e|H_i, m, 0)$ is 1 for $e = 0$, 0 otherwise. Note that we have already determined the term $P\left(e_{S''}(h_i^L) = e' \middle| H_i, \frac{m(n-1)}{n}, \frac{m}{n}\right)$ (the distribution of the holdout error on the first fold) in Equation 3. Equation 6 quantifies the error of the resulting hypothesis h_*^L returned by the cross validation wrapper when we replace the reference to Equation 5 by Equation 7.

We have now an efficient approximate average-case analysis of the generalization error of a classification algorithm with a cross validation wrapper that characterizes the behavior of that wrapper *for a given learning problem* $p(x, y)$ (which is observed via the sample). Note again that it is not necessary to actually run the learner in order to obtain the predictions.

4. Empirical Studies

Our analysis is based on some simplifying approximations, and the density $\pi(\epsilon|H_i)$ is estimated from the sample; therefore the predictions cannot be perfectly accurate. We now want to study (for many artificial and some UCI repository problems) to what degree predicted and observed error rates match. A close fit would indicate that our approximations were not unreasonable, and that our analysis can in fact predict the generalization error of cross validation accurately.

4.1 Boolean Functions

In the first part of this treatment, we conduct controlled experiments on randomly drawn boolean functions. Using many randomly drawn boolean functions lowers the risk of making accurate predictions only because assumptions of the analysis happen to be met by these specific problems. We can study the expected accuracy of our analysis over a large class of problems. Also, we can control the parameters of the learning problem, such as the number of relevant attributes.

The objective of the first set of experiments is to check whether the analysis predicts the change of the resulting error rate under varying split ratio correctly. The wrapper has model classes H_1 through H_6 available where H_i contains all functions over i boolean attributes. We drew 200 boolean functions with 4, 5, and 6 attributes and samples of size 50 and 100 at random and invoked the wrapper with split ratios of between 0.2 and 0.9. Since we knew the target function, we could determine the exact generalization error rates. Figure 1 compares the resulting error rates to the predictions (the outcome of Equation 6). Although there are significant deviations between predicted and measured error rates, the predicted error curves are reasonable approximations of the measured curves. The optimal training/test split (which a practitioner would be interested in) is predicted fairly accurately. This indicates that our simplifying assumptions are not completely unreasonable. It also indicates that the analysis is a possible heuristic for choosing the split ratio for a given problem, although the results do not show that it is better than known heuristics.

The objective of the next set of experiments is to check how accurately the analysis predicts the change of the error rate of a holdout and a 5-fold cross validation wrapper when we vary the number of relevant attributes of the target function. Also, we want to predict and measure the error rates of both a wrapper and an empirical error minimizing learner (that has all 6 attributes available) and see if our theory predicts correctly which one does better. We expect the wrapper to perform better than the learner when some attributes are irrelevant. We drew 200 target functions with between 3 and 6 relevant attributes at random (again H_i contains all functions over i attributes). Figure 2a shows the actual error rates of the hypotheses that were found by the learner and by the wrapper, respectively. When five or less attributes are relevant, the wrapper does better than the learner. The error rate of the wrapper is lowest (but not zero) when only three attributes are relevant since these are the "easiest" target functions. Figure 2b shows the predicted error rates which are fairly close to the actual values. For 5-fold cross validation (Figures 2c and 2d), we can again see a stronger deviation between actual and predicted error rates (we expected this since the analysis of n-fold cross validation relies on an additional independence assumption) but prediction and measurement agree in that, when all 6 attributes are relevant, the wrapper does just slightly worse than empirical error minimization.

4.2 Decision Trees

The objective of the following experiments is to check to which degree our analysis is able to predict the error rates of a realistic decision tree learner that uses cross validation to select an appropriate number of leaf nodes. Our decision tree learner is similar to the CART algorithm (Breiman et al., 1984): We wrap a greedy learner that minimizes the empirical error rate using exactly k leaf nodes into a layer of cross validation or holdout testing that chooses a k to minimize the cross validation error. We use the information gain criterion to select a split. We refer the reader to Scheffer (2000a) for algorithmic details. In our implementation, we considered 4000×2^i hypotheses to estimate $\pi(\epsilon|H_i)$ where i is the number of leaf nodes. We used the algorithm of Scheffer (1999) that runs in $O(i \times m)$.

For these problems, in contrast to the boolean problems reported above, we cannot determine the true error rates of the resulting hypotheses exactly. In order to obtain almost unbiased error estimates of the resulting decision tree algorithm, it is therefore necessary to wrap a second layer of 10-fold cross validation around the cross validation wrapper that selects the number of leaves; see Scheffer and Herbrich (1997) and Kohavi and John (1997) for more detailed discussions on almost unbiased error estimation. All error rates which we will refer to in the following are measured in an outer cross validation wrapper and are only subject to a very small pessimistic bias. Note that disagreements between prediction and measurement can mean that either the prediction *or* the cross validation estimate is inaccurate. In roughly 14% of all cases, the true error differs from the cross validation estimate by at least one standard deviation. As a rule of thumb, when prediction and measurement disagree by two standard deviations it is very likely that the prediction is wrong.

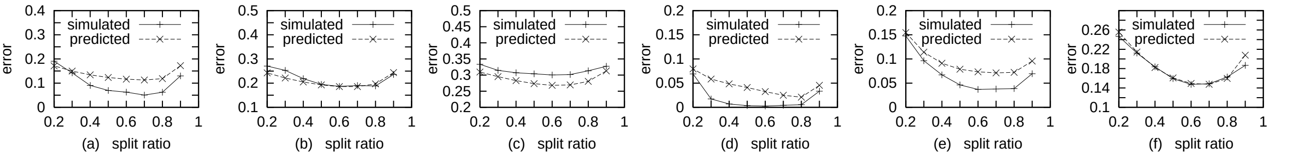

Figure 1: Predicted and measured generalization error of the wrapper for 200 boolean functions under varying training/test split ratios. (a)-(c): $m = 50$, target model has 4, 5, 6 attributes, respectively. (d)-(f) $m = 100$, 4, 5, 6 attributes respectively.

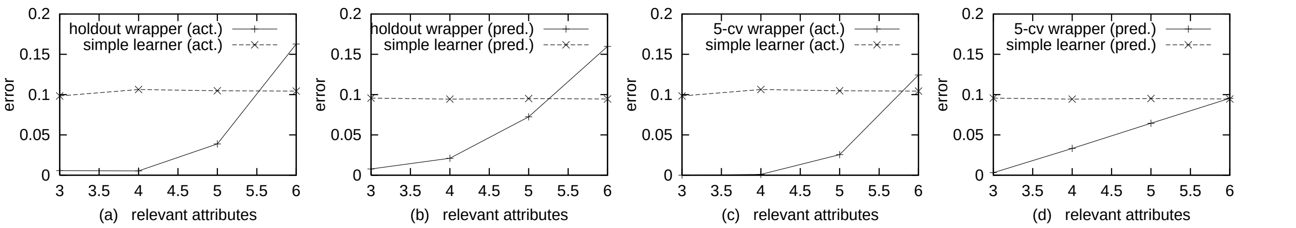

Figure 2: Predicted and measured tradeoff between simple empirical error minimizing learner and wrapper, depending on the number of attributes which are relevant to the target function. (a) Actual error rates (holdout wrapper) (b) predicted (c) actual error rates (5-fold cross validation wrapper) (d) predicted.

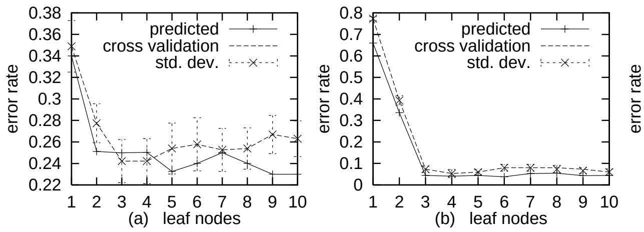

Figure 3: Predicted (Equation 1) and measured error rates of a decision tree with i leaf nodes for (a) diabetes and (b) iris.

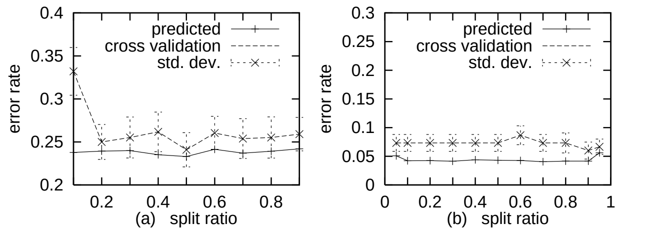

Figure 4: Predicted and measured error rate of a wrapper under varying split ratio for (a) diabetes and (b) iris.

We drew a set of eight learning problems (with no or few missing attributes) at random from the UCI repository. Due to lack of space, we cannot present all experimental results here. In order to obtain an unbiased picture of the results we refer the reader to Scheffer (2000b) where all curves are presented.

Equation 6 can only quantify the generalization error of the wrapper if Equation 1 predicts the error rate of an empirical error minimizing learner correctly. Let us see if the prediction of Equation 1 and the cross validation measurement agree when we vary the number of leaf nodes of a decision tree for some of the problems. We have conducted experiments on eight domains of which we present the first two in Figure 3. Predictions and measurements agree relatively closely; we can see significant deviations only at some points for the cleveland and *E.coli* data sets (Scheffer, 2000b).

Let us now study if predicted and measured error rate of the wrapper agree when we vary the split ratio. Figure 4 compares the measured to the predicted error rate of the wrapper for split ratios between 0.1 and 0.9 (for the first two of four studied problems). Surprisingly, prediction and simulation agree that the generalization performance hardly depends on the split ratio. Since the curves of Figure 1 look differently, this seems to be a property of the studied problems. Predictions and simulated error rates do not differ significantly except for diabetes with a split ratio of 0.1.

The objective of the last experiment is to check whether predicted and measured error rates of the wrapper agree when we vary the number n of cross validation folds of the wrapper. Figure 5 compares measured to predicted error rates from $n = 1$ (holdout testing) to $n = 10$ for the first two (out of eight) experiments that we conducted. Like in the previous experiments, the measurements are based on an outer layer of 10-fold cross validation. The results (Figure 5) are still relatively accurate; for the cleveland and *E. Coli* problems we can see significant deviations. Surprisingly, the number of cross validation folds has a rather low impact on the resulting error rate (except for the wine data set). But as the previous experiments

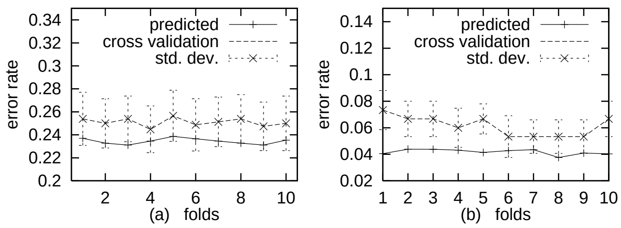

Figure 5: Predicted and measured error rate of an n-fold cross validation wrapper, under varying n for (a) diabetes and (b) iris.

show, this is not always the case, it is a property of the selected learning problems (the posed model selection problems are easy, even holdout testing solves them not much worse than 10-fold cross validation).

In total, the reported experiments indicate that our assumptions were not unreasonable and that our analysis can predict the behavior of wrappers with reasonable accuracy. But many predictions are not perfectly accurate; this suggests that further improvement can be gained by weakening the underlying assumptions.

5. Discussion and Related Results

Traditional PAC- and VC-style analyses of learning algorithms give worst-case bounds on the generalization error that hold for a certain class of target functions, but are overly pessimistic for almost each single function. By contrast, average-case analyses quantify the expected (over all samples) error of a learning algorithm for *a* given target function. We presented an approximate, computationally efficient average-case analysis of empirical error minimizing learners with n-fold cross validation wrappers. We checked empirically that, despite the simplifying assumptions and the need to estimate a distribution from the data, the analysis approximates empirically observed behavior well. The analysis can serve as a heuristic for determining a split ratio, or to find a reasonable trade-off between computational expenses and accuracy when determining the number of cross validation folds.

Approximate average-case analyses have been discussed for various learners. Iba and Langley (1992) have studied the behavior of decision stump learners. Okamoto and Nobuhiro (1997) presented an analysis for k-nearest neighbor classifiers; Fukumizu (1999) for linear neural networks. Reischuk and Zeugmann (1998) analyzed an algorithm that learns one-variable pattern languages. Cross validation has been studied for support vector machines by Joachims (1999).

An analysis of naive Bayesian classifiers has been presented by Langley et al. (1992); under some simplifying approximations (Langley & Sage, 1999) the analysis is computationally efficient. Learning in naive Bayes means estimating class-specific probabilities for each attribute; one crucial question is how misestimation of these probabilities increases the resulting error. Their solution depends on the target function (the correct set of relevant attributes) which has to be known before the analysis can be applied. By contrast, our analysis treats hypotheses as black boxes that only possess error rates; our solution depends on a density that, in principle, can be estimated from the sample.

Scheffer and Joachims (1998) and Domingos (1999) discussed almost identical analyses for exhaustive learners under some independence assumptions; Scheffer and Joachims (1999) removed one of these assumptions. These results for empirical error minimizing learners provide the basis of our analysis of cross validation. The fact that we can get from this point relatively easily to a result that gives us at least some information on the process of cross validation emphasizes that their analysis is an interesting tool.

Unfortunately, our analysis has several limitations that we would like to overcome in the future. Firstly, the analysis is restricted to cross validation wrappers for *exhaustive* learners. We have extended our analysis of empirical error minimizing learners to the case of greedy learners (Scheffer, 2000a) but this approach cannot directly be applied to our analysis of cross validation. Secondly, we assume that the model classes be finite. This assumption is reasonable for decision trees but it would be interesting to see if similar analyses can be conducted for, *e.g.,* neural networks.

Thirdly, the independence assumptions present in the analysis offer potential for further improvement. Experiments by Scheffer (1999) have shown that even when $\pi(\epsilon|H_i)$ is known the predictions of Equation 1 are not perfectly accurate; the independence assumptions are the only remaining source of this inaccuracy. It would be particularly interesting to remove the assumption of independent holdout errors of distinct cross validation folds. For error minimizing learners, Langford and McAllester (2000) have in fact removed all independence assumptions, but they arrive at a worst-case analysis again (that benefits from the additional information of $\pi(\epsilon|H_i)$). It would be interesting to see if there is an efficient average-case analysis of cross validation with weaker assumptions.

Acknowledgement

I would like to thank the reviewers for their detailed and helpful comments.

References

Anthony, M., & Holden, S. (1998). Cross-validation for binary classification by real-valued functions. *Proceedings of the Eleventh International Conference on Computational Learning Theory*.

Blum, A., Kalai, A., & Langford, J. (1999). Beating the hold-out: Bounds for k-fold and progressive cross-validation. *Proceedings of the International Conference on Computational Learning Theory*.

Breiman, L., Friedman, J., Ohlsen, R., & Stone, C. (1984). *Classification and regression trees*. Pacific Grove.

Dietterich, T. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, *10*, 1895–1924.

Domingos, P. (1999). Process-oriented estimation of generalization error. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligenct*.

Fukumizu, K. (1999). Generalization error of linear neural networks in unidentifiable cases. *Proceedings of the Tenth International Conference on Algorithmic Learning Theory*.

Gilks, W., Richardson, S., & Spiegelhalter, D. (Eds.). (1995). *Markov chain monte carlo in practice*. Chapman & Hall.

Iba, W., & Langley, P. (1992). Induction of one-level decision trees. *Proceedings of the Ninth International Conference on Machine Learning* (pp. 233–240).

Joachims, T. (1999). *Estimating the generalization performance of a SVM efficiently*. Technical Report, Uni Dortmund, Dept. of Computer Science, LS-8.

Kearns, M., Mansour, Y., Ng, A., & Ron, D. (1997). An experimental and theoretical comparison of model selection methods. *Machine Learning*, *27*, 7–50.

Kearns, M., & Ron, D. (1997). Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Proceedings of the Tenth International Conference on Computational Learning Theory*.

Kohavi, R., & John, G. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, *97*, 245–271.

Lachenbruch, P. (1967). An almost unbiased method of obtaining confidence intervals for the probability of misclassification in discriminant analysis. *Biometrics*, *23*, 639–645.

Langford, J., & McAllester, D. (2000). Computable shell decomposition bounds. Unpublished manuscript, CMU, Pittsburgh.

Langley, P., Iba, W., & Thompson, K. (1992). An analysis of bayesian classifiers. *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp. 223–228).

Langley, P., & Sage, S. (1999). Tractable average case analysis of naive bayes classifiers. *Proceedings of the Sixteenth International Conference on Machine Learning* (pp. 220–228).

Mosier, C. (1951). Problems and designs of cross validation. *Educational and Psychological Measurement*, *11*, 5–11.

Okamoto, S., & Nobuhiro, Y. (1997). An average-case analysis of the k-nearest neighbor classifier for noisy domains. *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence* (pp. 238–243).

Reischuk, R., & Zeugmann, T. (1998). Learning 1-variable pattern languages in linear average time. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory* (pp. 198–208).

Scheffer, T. (1999). *Error estimation and model selection*. Sankt Augustin: Infix Publisher.

Scheffer, T. (2000a). Nonparametric regularization of decision trees. *Proceedings of the European Conference on Machine Learning*.

Scheffer, T. (2000b). Predicting the generalization performance of cross validatory model selection criteria. Technical Report, Uni Magdeburg, http://kd.cs.uni-magdeburg.de/~scheffer/papers/icml2k-tr.ps.

Scheffer, T., & Herbrich, R. (1997). Unbiased assessment of learning algorithms. *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence* (pp. 798–803).

Scheffer, T., & Joachims, T. (1998). Estimating the expected error of empirical minimizers for model selection (abstract). *Proceedings of the Fifteenth National Conference on Artificial Intelligence*.

Scheffer, T., & Joachims, T. (1999). Expected error analysis for model selection. *Proceedings of the Sixteenth International Conference on Machine Learning*.

Toussaint, G. (1974). Bibliography on estimation of misclassification. *IEEE Transactions on Information Theory IT20*, *4*, 472–479.