Algorithmic Stability and Sanity-Check Bounds for Leave-One-Out Cross-Validation

Michael Kearns
AT&T Labs Research
Murray Hill, New Jersey
mkearns@research.att.com

Dana Ron
MIT
Cambridge, MA
danar@theory.lcs.mit.edu

January 1997

Abstract: In this paper we prove *sanity-check bounds* for the error of the leave-one-out cross-validation estimate of the generalization error: that is, bounds showing that the worst-case error of this estimate is not much worse than that of the training error estimate. The name sanity-check refers to the fact that although we often expect the leave-one-out estimate to perform considerably better than the training error estimate, we are here only seeking assurance that its performance will not be considerably worse. Perhaps surprisingly, such assurance has been given only for rather limited cases in the prior literature on cross-validation.

Any nontrivial bound on the error of leave-one-out must rely on some notion of algorithmic stability. Previous bounds relied on the rather strong notion of *hypothesis stability*, whose application was primarily limited to nearest-neighbor and other local algorithms. Here we introduce the new and weaker notion of *error stability*, and apply it to obtain sanity-check bounds for leave-one-out for other classes of learning algorithms, including training error minimization procedures and Bayesian algorithms. We also provide lower bounds demonstrating the necessity of error stability for proving bounds on the error of the leave-one-out estimate, and the fact that for training error minimization algorithms, in the worst case such bounds must still depend on the Vapnik-Chervonenkis dimension of the hypothesis class.

1 Introduction and Motivation

A fundamental problem in statistics, machine learning, and related areas is that of obtaining an accurate estimate for the generalization ability of a learning algorithm trained on a finite data set. Many estimates have been proposed and examined in the literature, some of the most prominent being the *training error* (also known as the *resubstitution* estimate), the various *cross-validation* estimates (which include the *leave-one-out* or *deleted* estimate, as well as *k-fold* cross-validation), and the *holdout* estimate. For each of these estimates, the hope is that for a fairly wide class of learning algorithms A, the estimate will usually produce a value $\hat{\epsilon}$ that is close to the true (generalization) error ϵ of the hypothesis function chosen by A.

There are surprisingly few previous results providing bounds on the accuracy of the various estimates [15, 2, 3, 17, 9, 8, 12, 10] (see the recent book of Devroye, Györfi and Lugosi [1] for an excellent introduction and survey of the topic). Perhaps the most *general* results are those given for the (classification) training error estimate by Vapnik [17], who proved that for any target function and input distribution, and for any learning algorithm that chooses its hypotheses from a class of VC dimension d, the training error estimate is at most $\tilde{O}(\sqrt{d/m})$ [1] away from the true error, where m is the size of the training sample. On the other hand, among the *strongest* bounds (in the sense of the quality of the estimate) are those given for the leave-one-out estimate by the work of Rogers and Wagner [15], and Devroye and Wagner [2, 3]. The (classification error) leave-one-out estimate is computed by running the learning algorithm m times, each time removing one of the m training examples, and testing the resulting hypothesis on the training example that was deleted; the fraction of failed tests is the leave-one-out estimate. Rogers and Wagner [15] and Devroye and Wagner [2, 3] proved that for several specific algorithms, but again for any target function and input distribution, the leave-one-out estimate can be as close as $O(1/\sqrt{m})$ to the true error. The algorithms considered are primarily variants of nearest-neighbor and other local procedures, and as such do not draw their hypotheses from a fixed class of bounded VC dimension, which is the situation we are primarily interested in here.

A tempting and optimistic intuition about the leave-one-out estimate is that it should *typically* yield an estimate that falls within $O(1/\sqrt{m})$ of the true error. This intuition derives from viewing each deleted test as an independent trial of the true error. The problem, of course, is that these tests are not independent. The Devroye, Rogers and Wagner results demonstrate that for certain algorithms, the intuition is essentially correct despite the dependencies. In such cases, the leave-one-out estimate may be vastly preferable to the training error, yielding an estimate of the true error whose accuracy is independent of any notion of dimension or hypothesis complexity (although the true error itself may depend strongly on such quantities).

However, despite such optimism, the prior literature leaves open a disturbing possibility for the leave-one-out proponent: the possibility that its accuracy may often be, for wide classes of natural algorithms, *arbitrarily poor*. We would like to have what we shall informally refer to as a *sanity-check bound*: a proof, for large classes of algorithms, that the error of the leave-one-out estimate is not much worse than the $\tilde{O}(\sqrt{d/m})$ worst-case behavior of the training error estimate. The name sanity-check refers to the fact that although we believe that under many circumstances, the leave-one-out estimate will perform much *better* than the training error (and thus justify its

[1]The $\tilde{O}(\cdot)$ notation hides logarithmic factors in the same way that $O(\cdot)$ notation hides constants.

computational expense) the goal of the sanity-check bound is to simply prove that it is not much *worse* than the training error. Such a result is of interest simply because the leave-one-out estimate is in wide experimental use (largely because practitioners do expect it to frequently outperform the training error), so it behooves us to understand its performance and limitations.

A moment's reflection should make it intuitively clear that, in contrast to the training error, even a sanity-check bound for leave-one-out cannot come without restrictions on the algorithm under consideration: some form of algorithmic *stability* is required [3, 9, 13]. If the removal of even a single example from the training sample may cause the learning algorithm to "jump" to a different hypothesis with, say, much larger error than the full-sample hypothesis, it seems hard to expect the leave-one-out estimate to be accurate. The precise nature of the required form of stability is less obvious.

Devroye and Wagner [3] first identified a rather strong notion of algorithmic stability that we shall refer to as *hypothesis* stability, and showed that bounds on hypothesis stability directly lead to bounds on the error of the leave-one-out estimate. This notion of stability demands that the removal of a single example from the training sample results in hypotheses that are "close" to each other, in the sense of having small symmetric difference with respect to the input distribution. For algorithms drawing hypotheses from a class of fixed VC dimension, the first sanity-check bounds for the leave-one-out estimate were provided by Holden [9] for two specific algorithms in the *realizable* case (that is, when the target function is actually contained in the class of hypothesis functions).

However, in the more realistic *unrealizable* (or *agnostic* [11]) case, the notion of hypothesis stability may simply be too strong to be obeyed by many natural learning algorithms. For example, if there are many local minima of the true error, an algorithm that managed to always minimize the training error might be induced to move to a rather distant hypothesis by the addition of a new training example (we shall elaborate on this example shortly). Many gradient descent procedures use randomized starting points, which may even cause runs on the same sample to end in different local minima. Algorithms behaving according to Bayesian principles will choose two hypotheses of equal training error with equal probability, regardless of their dissimilarity. What we might hope to be relatively stable in such cases would not be the algorithm's hypothesis itself, but the *error* of the algorithm's hypothesis.

The primary goal of this paper is to give sanity-check bounds for the leave-one-out estimate that are based on the error stability of the algorithm. In Section 2, we begin by stating some needed preliminaries. In Section 3, we review the Devroye and Wanger notion of hypothesis stability, and generalize the results of Holden [9] by showing that in the realizable case this notion can be used to obtain sanity-check bounds for *any* consistent learning algorithm; but we also discuss the limitations of hypothesis stability in the unrealizable case. In Section 4, we define our new notion of error stability, and prove our main results: bounds on the error of the leave-one-estimate that depend on the VC dimension of the hypothesis class and the error stability of the algorithm. The bounds apply to a wide class of algorithms meeting a mild condition that includes training error minimization and Bayesian procedures. In Section 5, we give a number of lower bound results showing, among other things, the necessity of error stability for proving bounds on leave-one-out, but also the absence of sufficiency. In Section 6 we conclude with some interesting open problems.

2 Preliminaries

Let f be a fixed *target function* from domain X to range Y, and let P be a fixed distribution over X. Both f and P may be arbitrary [2]. We use S_m to denote the random variable $S_m = \langle x_1, y_1 \rangle, \ldots, \langle x_m, y_m \rangle$, where m is the *sample size*, each x_i is drawn randomly and independently according to P, and $y_i = f(x_i)$. A *learning algorithm* A is given S_m as input, and outputs a *hypothesis* $h = A(S_m)$, where $h : X \to Y$ belongs to a fixed *hypothesis class* H. If A is randomized, it takes an additional input $\vec{r} \in \{0,1\}^k$ of random bits of the required length k to make its random choices. In this paper we study mainly the case in which $Y = \{0,1\}$, and briefly the case in which $Y = \Re$. For now we restrict our attention to boolean functions.

For any boolean function h, we define the *generalization error* of h (with respect to f and P) by $\epsilon(h) = \epsilon_{f,P}(h) \stackrel{\text{def}}{=} \Pr_{x \in P}[h(x) \neq f(x)]$. For any two boolean functions h and h', The *distance* between h and h' (with respect to P) is $\text{dist}(h, h') = \text{dist}_P(h, h') \stackrel{\text{def}}{=} \Pr_{x \in P}[h(x) \neq f(x)]$. Since the target function f may or may not belong to H, we define $\epsilon_{opt} \stackrel{\text{def}}{=} \min_{h \in H}\{\epsilon(h)\}$, and h_{opt} to be some function such that $\epsilon(h_{opt}) = \epsilon_{opt}$. Thus, the function h_{opt} is the best approximation to f (with respect to P) in the class H, and ϵ_{opt} measures the quality of this approximation. We define the *training error* of a boolean function h with respect to S_m by $\hat{\epsilon}(h) = \hat{\epsilon}_{S_m}(h) \stackrel{\text{def}}{=} |\{\langle x_i, y_i \rangle \in S_m : h(x_i) \neq y_i\}|/m$, and the (generalized) *version space* $VS(S_m) \stackrel{\text{def}}{=} \{h \in H : \hat{\epsilon}(h) = \min_{h' \in H}\{\hat{\epsilon}(h')\}\}$ consists of all functions in H that minimize the training error.

Throughout this paper we assume that the algorithm A is *symmetric*. This means that A is insensitive to the ordering of the examples in the input sample S_m, so for every ordering of S_m it outputs the same hypothesis. (In case A is randomized, it should induce the same distribution on hypotheses.) This is a very mild assumption, as any algorithm can be transformed into a symmetric algorithm by adding a randomizing preprocessing step. Thus, we may refer to S_m as an unordered set of labeled examples rather than as a list of examples. For any index $i \in [m]$, we denote by S_m^i the sample S_m with the i^{th} labeled example, $\langle x_i, y_i \rangle$, removed. That is, $S_m^i \stackrel{\text{def}}{=} S_m \setminus \{\langle x_i, y_i \rangle\}$. The *leave-one-out cross validation* estimate, $\hat{\epsilon}_{\text{cv}}^A(S_m)$, of the error of the hypothesis $h = A(S_m)$ is defined to be $\hat{\epsilon}_{\text{cv}}^A(S_m) \stackrel{\text{def}}{=} |\{i \in [m] : h^i(x_i) \neq y_i\}|/m$, where $h^i = A(S_m^i)$. We are thus interested in providing bounds on the error $|\hat{\epsilon}_{\text{cv}}^A(S_m) - \epsilon(A(S_m))|$ of the leave-one-out estimate.

The following uniform convergence bound, due to Vapnik [17] will be central to this paper.

THEOREM 2.1 *Let H be a hypothesis class with VC dimension $d < m$. Then, for every $m > 4$ and for any given $\delta > 0$, with probability at least $1 - \delta$, for every $h \in H$,*

$$|\hat{\epsilon}(h) - \epsilon(h)| \;<\; 2\sqrt{\frac{d\,(\ln(2m/d) + 1) + \ln(9/\delta)}{m}}. \tag{1}$$

We shall denote the quantity $2\sqrt{\frac{d(\ln(2m/d)+1)+\ln(9/\delta)}{m}}$ by $VC(d, m, \delta)$. Thus, for any learning algorithm A using a hypothesis space of VC dimension d, for any $\delta > 0$, with probability at least $1 - \delta$ over S_m, $|\hat{\epsilon}(A(S_m)) - \epsilon(A(S_m))| < VC(d, m, \delta)$.

[2] Our results generalize to the case in which we allow the target process to be any joint distribution over the sample space $X \times Y$, but it will be convenient to think of there being a distinct target function.

3 Sanity-Check Bounds via Hypothesis Stability

As we have mentioned already, it is intuitively clear that the performance of the leave-one-out estimate must rely on some kind of algorithmic stability (this intuition will be formalized in the lower bounds of Section 5). Perhaps the strongest notion of stability that an interesting learning algorithm might be expected to obey is that of *hypothesis stability*: namely, that small changes in the sample can only cause the algorithm to move to "nearby" hypotheses. The notion of hypothesis stability is due to Devroye and Wagner [3], and is formalized in a way that suits our purposes in the following definition [3].

DEFINITION 3.1 *We say that an algorithm A has* hypothesis stability (β_1, β_2) *if*

$$\Pr_{S_{m-1},\langle x,y\rangle}[\mathrm{dist}(A(S_m), A(S_{m-1})) \geq \beta_2] \quad \leq \quad \beta_1 \tag{2}$$

where $S_m = S_{m-1} \cup \{\langle x, y\rangle\}$.

We shall shortly argue that hypothesis stability is in fact too demanding a notion in many realistic situations. But first, we state the elegant theorem of Devroye and Wagner [3] that relates the error of the leave-one-out estimate for an algorithm to the hypothesis stability.

THEOREM 3.1 *Let A be any symmetric algorithm that has hypothesis stability (β_1, β_2). Then for any $\delta > 0$, with probability at least $1 - \delta$ over S_m,*

$$|\hat{\epsilon}^A_{\mathrm{CV}}(S_m) - \epsilon(A(S_m))| \leq \sqrt{\frac{1/(2m) + 3(\beta_1 + \beta_2)}{\delta}}. \tag{3}$$

Thus, if we are fortunate enough to have an algorithm with strong hypothesis stability (that is, small β_1 and β_2), the leave-one-out estimate for this algorithm will be correspondingly accurate. What kind of hypothesis stability should we expect for natural algorithms? Devroye, Rogers and Wagner [15, 3] gave rather strong hypothesis stability results for certain nonparametric local learning algorithms (such as nearest-neighbor rules), and thus were able to show that the error of the leave-one-out estimate for such algorithms decreases like $1/m^\alpha$ (for values of α ranging from $1/4$ to $1/2$).

Note that for nearest-neighbor algorithms, there is no fixed "hypothesis class" of limited VC dimension — the algorithm may choose arbitrarily complex hypotheses. This unlimited complexity often makes it difficult to quantify the performance of the learning algorithm except in terms of the *asymptotic* generalization error (see Devroye, Györfi and Lugosi [1] for a detailed survey of results for nearest-neighbor algorithms). For this and other reasons, practitioners often prefer to commit to a hypothesis class H of fixed VC dimension d, and use heuristics to find a good function in H. In this case, we gain the possibility of finite-sample generalization error bounds (where we compare the error to that of the optimal model from H). However, in such a situation, the goal of hypothesis stability may in fact be at odds with the goal of good performance in the sense of learning. To see

[3] Devroye and Wagner [3] formalized hypothesis stability in terms of the *expected* difference between the hypotheses; here we translate to the "high probability" form for consistency.

this, imagine that the input distribution and target function define a generalization error "surface" over the function space H, and that this surface has minima at $h_{opt} \in H$, where $\epsilon(h_{opt}) = \epsilon_{opt} > 0$, and also at $h' \in H$, where $\epsilon(h') = \epsilon(h_{opt}) + \alpha$ for some small $\alpha > 0$. Thus, h_{opt} is the "global" minimum, and h' is a "local" minimum. Note that $\mathrm{dist}(h_{opt}, h')$ could be as large as $2\epsilon_{opt}$, which we are assuming may be a rather large (constant) quantity. Now if the algorithm A minimizes the training error over H, then we expect that as $m \to \infty$, algorithm A will settle on hypotheses closer and closer to h_{opt}. But for $m << 1/\alpha$, A may well choose hypotheses close to h'. Thus, as more examples are seen, at some point A may need to move from h' to the rather distant h_{opt}.

We do not know how to rule out such behavior for training error minimization algorithms, and so cannot apply Theorem 3.1. Perhaps more importantly, for certain natural classes of algorithms (such as the Bayesian algorithms discussed later), and for popular heuristics such as C4.5 and backpropagation, it is far from obvious that any nontrivial statement about hypothesis stability can be made. For this reason, we would like to have bounds on the error of the leave-one-out estimate that rely on the weakest possible notion of stability. Note that in the informal example given above, the quantity that we might hope *would* exhibit some stability is not the hypothesis itself, but the *error* of the hypothesis: even though h_{opt} and h' may be far apart, if A chooses h' then α must not be "too large". The main question addressed in this paper is when this weaker notion of error stability is sufficient to prove nontrivial bounds on the leave-one-out error, and we turn to this in Section 4.

First, however, note that the instability of the hypothesis above relied on the assumption that $\epsilon_{opt} > 0$ — that is, that we are in the unrealizable setting. In the realizable $\epsilon_{opt} = 0$ case, there is still hope for applying hypothesis stability. Indeed, Holden [9] was the first to apply uniform convergence results to obtain sanity-check bounds for leave-one-out via hypothesis stability, for two particular (consistent) algorithms in the realizable setting [4]. Here we generalize Holden's results by giving a sanity-check bound on the leave-one-out error for *any* consistent algorithm. The simple proof idea again highlights why hypothesis stability seems difficult to apply in the unrealizable case: in the realizable case, minimizing the training error forces the hypothesis to be close to some *fixed* function (namely, the target). In the unrealizable case, there may be many different functions, all with optimal or near-optimal error.

THEOREM 3.2 *Let H be a class of VC dimension d, and let the target function f be contained in H (realizable case). Let A be a symmetric algorithm that always finds an $h \in H$ consistent with the input sample. Then for every $\delta > 0$, with probability at least $1 - \delta$,*

$$|\hat{\epsilon}_{\mathrm{CV}}(S_m) - \epsilon(A(S_m))| = O\left(\sqrt{\frac{(d/m)\log(m/d)}{\delta}}\right). \tag{4}$$

PROOF: By uniform convergence, with probability at least $1 - \delta'$,

$$\epsilon(A(S_m)) = \mathrm{dist}(f, A(S_m)) = O\left(\frac{d\log(m/d) + \log(1/\delta')}{m}\right) \tag{5}$$

[4] Holden [8] has recently obtained sanity-check bounds, again for the realizable setting, for other cross-validation estimates.

and

$$\epsilon(A(S_{m-1})) = \mathrm{dist}(f, A(S_{m-1})) = O\left(\frac{d\log(m/d) + \log(1/\delta')}{m-1}\right). \tag{6}$$

(Here we are using the stronger $\tilde{O}(d/m)$ uniform convergence bounds that are special to the realizable case.) Thus by the triangle inequality, with probability at least $1-\delta'$, $\mathrm{dist}(A(S_m), A(S_{m-1})) = O\left(\frac{d\log(m/d)+\log(1/\delta')}{m}\right)$. The theorem follows from Theorem 3.1, where δ' is set to d/m.
□(Theorem 3.2)

We should note immediately that the bound of Theorem 3.2 has a dependence on $\sqrt{1/\delta}$, as opposed to the $\log(1/\delta)$ dependence for the training error given by Theorem 2.1. Unfortunately, it is well-known [1] (and demonstrated in Section 5) that, at least in the unrealizable setting, a $1/\delta$ dependence is in general unavoidable for the leave-one-out estimate. Thus, it appears that in order to gain whatever benefits leave-one-out offers, we must accept a worst-case dependence on δ inferior to that of the training error. Also, we note in passing that Theorem 3.2 can also be generalized (perhaps with a worse power of d/m) to the case where the target function lies in H but is corrupted by random classification noise: again, minimizing training error forces the hypothesis to be close to the target.

It is possible to give examples in the realizable case for which the leave-one-out estimate has error $O(1/\sqrt{m})$ while the training error has error $\Omega(\sqrt{d/m})$; such examples merely reinforce the intuition discussed in the introduction that leave-one-out may often be superior to the training error. Furthermore, there are unrealizable examples for which the error of leave-one-out is again independent of d, but for which *no* nontrivial leave-one-out bound can be obtained by appealing to hypothesis stability. It seems that a more general notion of stability is called for.

4 Sanity-Check Bounds via Error Stability

In this section, we introduce the notion of error stability and use it to prove our main results. We give bounds on the error of the leave-one-out estimate that are analogous to those given in Theorem 3.1, in that the quality of the bounds is directly related to the error stability of the algorithm. However, unlike Theorem 3.1, in all of our bounds there will be a residual $\tilde{O}(\sqrt{d/m})$ term that appears regardless of the stability; this is the price we pay for using a weaker — but more widely applicable — type of stability. In Section 5, we will show that the dependence on the error stability is always necessary, and also that a dependence on d/m cannot be removed in the case of algorithms which minimize the training error without further assumptions on the algorithm.

For expository purposes, we limit our attention to deterministic algorithms for now. The generalization to randomized algorithms will be discussed shortly. Our key definition mirrors the form of Definition 3.1.

DEFINITION 4.1 *We say that a deterministic algorithm A has* error stability (β_1, β_2) *if*

$$\Pr_{S_{m-1},\langle x,y\rangle}[|\epsilon(A(S_m)) - \epsilon(A(S_{m-1}))| \geq \beta_2] \quad \leq \quad \beta_1 \tag{7}$$

where $S_m = S_{m-1} \cup \{\langle x,y\rangle\}$, and both β_1 and β_2 may be functions of m.

Our goal is thus to prove bounds on the error of the leave-one-out estimate that depend on β_1 and β_2. This will require an additional (and hopefully mild) assumption on the algorithm that is quantified by the following definition. We will shortly prove that some natural classes of algorithms do indeed meet this assumption, thus allowing us to prove sanity-check bounds for these classes.

DEFINITION 4.2 *For any deterministic algorithm A, we say that the leave-one-out estimate* (γ_1, γ_2)-overestimates the training error *for A if*

$$\Pr_{S_{m-1},\langle x,y\rangle}[\hat{\epsilon}_{\mathrm{CV}}^{A}(S_m) \leq \hat{\epsilon}(A(S_m)) - \gamma_2] \quad \leq \quad \gamma_1 \tag{8}$$

where $S_m = S_{m-1} \cup \{\langle x, y\rangle\}$*, and both* γ_1 *and* γ_2 *may be functions of* m*.*

While we cannot claim that training error overstimation is in general necessary for obtaining bounds on the error of the leave-one-out estimate, we note that it is clearly necessary whenever the training error *underestimates* the true error, as is the case for algorithms that minimize the training error. In any case, in Section 5 we show that *some* additional assumptions (beyond error stability) are *required* to obtain nontrivial bounds for the error of leave-one-out.

Before stating the main theorem of this section, we give the following simple but important lemma that is well-known [1].

LEMMA 4.1 *For any symmetric learning algorithm A,*

$$\mathrm{E}_{S_m}[\hat{\epsilon}_{\mathrm{CV}}^{A}(S_m)] \quad = \quad \mathrm{E}_{S_{m-1}}[\epsilon(A(S_{m-1}))]. \tag{9}$$

PROOF: For any fixed sample S_m, let $h^i = A(S_m^i)$, and let $e_i \in \{0, 1\}$ be 1 if and only if $h^i(x_i) \neq y_i$. Then

$$\mathrm{E}_{S_m}[\hat{\epsilon}_{\mathrm{CV}}^{A}(S_m)] \;=\; \mathrm{E}_{S_m}\left[\frac{1}{m}\sum_i e_i\right] \;=\; \frac{1}{m}\sum_i \mathrm{E}_{S_m}[e_i] \;=\; \mathrm{E}_{S_m}[e_1] \;=\; \mathrm{E}_{S_{m-1}}[\epsilon(A(S_{m-1}))]. \tag{10}$$

The first equality follows from the definition of leave-one-out, the second from the additivity of expectation, the third from the symmetry of A, and the fourth from the definition of e_1. □(Lemma 4.1)

The first of our main results follows.

THEOREM 4.1 *Let A be any deterministic algorithm using a hypothesis space H of VC dimension d such that A has error stability* (β_1, β_2)*, and leave-one-out* (γ_1, γ_2)*-overestimates the training error for A. Then for any* $\delta > 0$*, with probability at least* $1 - \delta$ *over* S_m*,*

$$|\hat{\epsilon}_{\mathrm{CV}}^{A}(S_m) - \epsilon(A(S_m))| \;\leq\; \left(3\sqrt{\frac{(d+1)\,(\ln(9m/d)+1)}{m}} + 3\beta_1 + \beta_2 + \gamma_1 + \gamma_2\right)/\delta. \tag{11}$$

Let us briefly discuss the form of the bound given in Theorem 4.1. First of all, as we mentioned earlier, there is a residual $\tilde{O}(\sqrt{d/m})$ term that remains no matter how error-stable the algorithm this. This means that we cannot hope to get something better than a sanity-check bound from this result. Our main applications of Theorem 4.1 will be to show specific, natural cases in which γ_1 and γ_2 can be eliminated from the bound, leaving us with a bound that depends *only* on the error stability and the residual $\tilde{O}(\sqrt{d/m})$ term. We now turn to the proof of the theorem.

PROOF: From Theorem 2.1 and the fact that leave-one-out (γ_1, γ_2)-overestimates the training error, we have that with probability at least $1 - \delta' - \gamma_1$, (where δ' will be determined by the analysis)

$$\hat{\epsilon}^A_{\mathrm{cv}}(S_m) \;\geq\; \hat{\epsilon}(A(S_m)) - \gamma_2 \;\geq\; \epsilon(A(S_m)) - VC(d, m, \delta') - \gamma_2. \tag{12}$$

Thus, the fact that leave-one-out does not underestimate the training error by more than γ_2 (with probability at least $1 - \gamma_1$) immediately lets us bound the amount by which leave-one-out could *underestimate* the true error $\epsilon(A(S_m))$ (where here we set δ' to be $\delta/2$ and note that whenever $\gamma_1 \geq \delta/2$, the bound holds trivially). It remains to bound the amount by which leave-one-out could *overestimate* the true error.

Let us define the random variable $\chi(S_m)$ by $\hat{\epsilon}^A_{\mathrm{cv}}(S_m) = \epsilon(A(S_m)) + \chi(S_m)$, and let us define $\tau \stackrel{\mathrm{def}}{=} VC(d, m, \delta') + \gamma_2$, and $\rho \stackrel{\mathrm{def}}{=} \delta' + \gamma_1$. Then Equation (12) says that with probability at most ρ, $\chi(S_m) < -\tau$. Furthermore, it follows from the error stability of A that with probability at least $1 - \beta_1$,

$$\chi(S_m) \;\leq\; \hat{\epsilon}^A_{\mathrm{cv}}(S_m) - \epsilon(A(S_{m-1})) + \beta_2 \tag{13}$$

(where $S_{m-1} \cup \{\langle x, y\rangle\} = S_m$). By Lemma 4.1 we know that $\mathrm{E}_{S_{m-1},\langle x,y\rangle}[\hat{\epsilon}^A_{\mathrm{cv}}(S_m) - \epsilon(A(S_{m-1}))] = 0$, and hence on those samples for which Equation (13) holds (whose total probability weight is at least $1 - \beta_1$), the expected value of $\hat{\epsilon}^A_{\mathrm{cv}}(S_m) - \epsilon(A(S_{m-1}))$ is at most $\beta_1/(1 - \beta_1)$. Assuming $\beta_1 \leq 1/2$ (since otherwise the bound holds trivially), and using the fact that $|\chi(S_m)| \leq 1$, we have that

$$\mathrm{E}_{S_m}[\chi(S_m)] \;\leq\; 3\beta_1 + \beta_2. \tag{14}$$

Let α be such that with probability exactly δ, $\chi(S_m) > \alpha$. Then

$$3\beta_1 + \beta_2 \;\geq\; \mathrm{E}_{S_m}[\chi(S_m)] \;\geq\; \delta\alpha + \rho(-1) + (1 - \delta - \rho)(-\tau) \;\geq\; \delta\alpha - \rho - \tau \tag{15}$$

where we have again used the fact that $|\chi(S_m)| \leq 1$ always. Thus

$$\alpha \;\leq\; \frac{3\beta_1 + \beta_2 + \rho + \tau}{\delta}. \tag{16}$$

From the above we have that with probability at least $1 - \delta$,

$$\begin{aligned} \hat{\epsilon}^A_{\mathrm{cv}}(S_m) &\leq \epsilon(A(S_m)) + (3\beta_1 + \beta_2 + \rho + \tau)/\delta & (17)\\ &= \epsilon(A(S_m)) + (VC(d, m, \delta') + 3\beta_1 + \beta_2 + \gamma_1 + \gamma_2 + \delta')/\delta. & (18) \end{aligned}$$

If we set $\delta' = d/m$, we get that with probability at least $1 - \delta$,

$$\hat{\epsilon}^A_{\mathrm{cv}}(S_m) \;\leq\; \epsilon(A(S_m)) + \left(3\sqrt{\frac{(d+1)\,(\ln(9m/d) + 1)}{m}} + 3\beta_1 + \beta_2 + \gamma_1 + \gamma_2\right)/\delta \tag{19}$$

which together with Equation (12) proves the theorem. □(Theorem 4.1)

4.1 Application to Training Error Minimization

In this section, we give one of our main applications of Theorem 4.1, by showing that for training error minimization algorithms, a $\tilde{O}(\sqrt{d/m})$ bound on the error of leave-one-out can be obtained from error stability arguments. We proceed by giving two lemmas, the first bounding the error stability of such algorithms, and the second proving that leave-one-out overestimates their training error.

LEMMA 4.2 *Let A be any algorithm performing training error minimization over a hypothesis class H of VC dimension d. Then for any $\beta_1 > 0$, A has error stability* $(\beta_1, 2VC(d, m-1, \beta_1/2)))$.

PROOF: From uniform convergence (Theorem 2.1), we know that with probability at least $1-\beta_1$, both $\epsilon(A(S_{m-1})) \leq \epsilon_{opt} + 2VC(d, m-1, \beta_1/2)$ and $\epsilon(A(S_m)) \leq \epsilon_{opt} + 2VC(d, m, \beta_1/2)$ hold, while it is always true that both $\epsilon(A(S_m)) \geq \epsilon_{opt}$, and $\epsilon(A(S_{m-1})) \geq \epsilon_{opt}$. Thus with probability at least $1-\beta_1$, $|\epsilon(A(S_{m-1})) - \epsilon(A(S_m))| \leq 2VC(d, m-1, \beta_1/2))$. □(Lemma 4.2)

LEMMA 4.3 *Let A be any algorithm performing training error minimization over a hypothesis class H. Then leave-one-out* $(0,0)$*-overestimates the training error for A.*

PROOF: Let $h = A(S_m)$ and $h^i = A(S^i_m)$. Let $err(S_m)$ be the subset of unlabeled examples in S_m on which h errs. We claim that for every $\langle x_i, y_i \rangle \in err(S_m)$, h^i errs on $\langle x_i, y_i \rangle$ as well, implying that $\hat{\epsilon}^A_{\mathrm{CV}}(S_m) \geq \hat{\epsilon}(A(S_m))$. Assume, contrary to the claim, that for some i, $h(x_i) \neq y_i$ while $h^i(x_i) = y_i$. For any function g and sample S, let $e_g(S)$ denote the number of errors made by g on S (thus $e_g(S) = \hat{\epsilon}(g) \cdot |S|$). Since A performs training error minimization, for any function $h' \in H$ we have $e_{h'}(S_m) \geq e_h(S_m)$. Similarly, for any $h' \in H$, we have $e_{h'}(S^i_m) \geq e_{h^i}(S^i_m)$. In particular this must be true for h, and thus $e_h(S^i_m) \geq e_{h^i}(S^i_m)$. Since h errs on $\langle x_i, y_i \rangle$, $e_h(S^i_m) = e_h(S_m) - 1$, and hence $e_{h^i}(S^i_m) \leq e_h(S_m) - 1$. But since h^i does not err on $\langle x_i, y_i \rangle$, $e_{h^i}(S_m) = e_{h^i}(S^i_m) \leq e_h(S_m) - 1 < e_h(S_m)$, contradicting the assumption that h minimizes the training error on S_m. □(Lemma 4.3)

THEOREM 4.2 *Let A be any algorithm performing training error minimization over a hypothesis class H of VC dimension d. Then for every $\delta > 0$, with probability at least $1-\delta$,*

$$|\hat{\epsilon}^A_{\mathrm{CV}}(S_m) - \epsilon(A(S_m))| \;\leq\; \left(8\sqrt{\frac{(d+1)\left(\ln(9m/d)+2\right)}{m}}\right)/\delta. \tag{20}$$

PROOF: Follows immediately from Lemma 4.2 (where β_1 is set to $2d/m$), Lemma 4.3, and Theorem 4.1. □(Theorem 4.2)

Thus, for training error minimization algorithms, *the worst-case behavior of the leave-one-out estimate is not worse than that of the training error (modulo the inferior dependence on $1/\delta$ and constant factors).* We would like to infer that a similar statement is true if the algorithm *almost* minimizes the training error. Unfortunately, Lemma 4.3 is extremely sensitive, forcing us to simply *assume* that leave-one-out overestimates the training error in the following theorem. We will later discuss how reasonable such an assumption might be for natural algorithms; in any case, we will show in Section 5 that some assumptions beyond just error stability are required to obtain interesting bounds for leave-one-out.

THEOREM 4.3 *Let A be a deterministic algorithm that comes within Δ of minimizing the training error over H (that is, on any sample S_m, $\hat{\epsilon}(A(S_m)) \leq \min_{h \in H}\{\hat{\epsilon}(h)\} + \Delta$), and suppose that leave-one-out $(0,0)$-overestimates the training error for A. Then with probability at least $1-\delta$,*

$$|\hat{\epsilon}_{\text{CV}}(S_m) - \epsilon(A(S_m))| \leq \left(8\sqrt{\frac{(d+1)\left(\ln(2m/d)+2\right)}{m}} + \Delta\right)/\delta. \quad (21)$$

PROOF: The theorem follows from the fact any algorithm that comes within Δ of minimizing the training error has error stability $(\beta_1, \Delta + 2VC(d, m-1, \beta_1/2))$ (the proof is similar to that of Lemma 4.2), and from Theorem 4.1. □(Theorem 4.3)

4.2 Application to Bayesian Algorithms

We have just seen that training error minimization in fact implies error stability sufficient to obtain a sanity-check bound on the error of leave-one-out. More generally, we might hope to obtain bounds that depend on whatever error stability an algorithm *does* possess (but not better bounds; see Section 5). In this section, we show that this hope can be realized for a natural class of randomized algorithms that behave in a Bayesian manner.

To begin with, we generalize Definitions 4.1 and 4.2 to include randomization simply by letting the probability in both definitions be taken over both the sample S_m and any randomization required by the algorithm. We use the notation $A(S, \vec{r}\,)$ to denote the hypothesis output by A on input sample S and random string $\vec{r}$, and $\hat{\epsilon}^A_{\text{CV}}(S_m, \vec{r}_1, \ldots, \vec{r}_m\,)$ to denote the leave-one-out estimate when the random string $\vec{r}_i$ is used on the call to A on S^i_m.

DEFINITION 4.3 *We say that a randomized algorithm A has* error stability (β_1, β_2) *if*

$$\Pr_{S_{m-1},\langle x,y\rangle,\vec{r},\vec{r}\,'}\left[|\epsilon(A(S_m,\vec{r}\,)) - \epsilon(A(S_{m-1},\vec{r}\,'\,))| \geq \beta_2\right] \leq \beta_1 \quad (22)$$

where $S_m = S_{m-1} \cup \{\langle x,y\rangle\}$.

DEFINITION 4.4 *For any randomized algorithm A, we say that* leave-one-out (γ_1, γ_2)-overestimates the training error *for A if*

$$\Pr_{S_{m-1},\langle x,y\rangle,\vec{r},\vec{r}_1,\ldots,\vec{r}_m}\left[\hat{\epsilon}^A_{\text{CV}}(S_m,\vec{r}_1,\ldots,\vec{r}_m\,) \leq \hat{\epsilon}(A(S_m,\vec{r}\,)) - \gamma_2\right] \leq \gamma_1 \quad (23)$$

where $S_m = S_{m-1} \cup \{\langle x,y\rangle\}$.

The proof of the following theorem is essentially the same as the proof of Theorem 4.1 where the only difference is that all probabilities are taken over the sample S_m *and* the randomization of the algorithm.

THEOREM 4.4 *Let A be any randomized algorithm using a hypothesis space H of VC dimension d such that leave-one-out (γ_1, γ_2)-overestimates the training error for A, and A has error stability (β_1, β_2). Then for any $\delta > 0$, with probability at least $1-\delta$,*

$$|\hat{\epsilon}^A_{\text{CV}}(S_m,\vec{r}_1,\ldots,\vec{r}_m\,) - \epsilon(A(S_m,\vec{r}\,))| = \frac{3\sqrt{\frac{(d+1)\left(\ln\frac{2m}{d}+1\right)}{m}} + 3\beta_1 + \beta_2 + \gamma_1 + \gamma_2}{\delta}. \quad (24)$$

Here the probability is taken over the choice of S_m, and over the coin flips $\vec{r}_1, \ldots, \vec{r}_m$ and $\vec{r}$ of A on the S_m^i and S_m.

We now apply Theorem 4.1 to the class of *Bayesian* algorithms — that is, algorithms that choose their hypotheses according to a posterior distribution, obtained from a prior that is modified by the sample data and a temperature parameter. Such algorithms are frequently studied in the simulated annealing and statistical physics literature on learning [16, 5].

DEFINITION 4.5 *We say that a randomized algorithm A using hypothesis space H is a* Bayesian *algorithm if there exists a* prior *$\mathcal{P}$ over H and a* temperature *$T \geq 0$ such that for any sample S_m and any $h \in H$,*

$$\Pr_{\vec{r}}\left[A(S_m, \vec{r}\,) = h\right] \;=\; \frac{1}{Z}\mathcal{P}(h)\exp\left(-\frac{1}{T}\sum_i I(h(x_i) \neq y_i)\right). \tag{25}$$

Here $Z = \sum_{h \in H} \mathcal{P}(h)\exp\left(-\frac{1}{T}\sum_i I(h(x_i) \neq y_i)\right)$ is the appropriate normalization.

Note that we still do not assume anything about the target function (for instance, it is not necessarily drawn according to $\mathcal{P}$ or any other distribution) — it is only the *algorithm* that behaves in a Bayesian manner. Also, note that the special case in which $T = 0$ and the support of $\mathcal{P}$ is H results in training error minimization.

We begin by giving a general lemma that identifies the only property about Bayesian algorithms that we will need; thus, all of our subsequent results will hold for any algorithm meeting the conclusion of this lemma.

LEMMA 4.4 *Let A be a Bayesian algorithm. For any sample S and any example $\langle x, y\rangle \in S$, let p be the probability over $\vec{r}$ that $A(S, \vec{r}\,)$ errs on $\langle x, y\rangle$, and let p' be the probability over $\vec{r}\,'$ that $A(S - \{\langle x, y\rangle\}, \vec{r}\,')$ errs on $\langle x, y\rangle$. Then $p' \geq p$.*

PROOF: Let $\mathcal{P}$ be the distribution induced over H when A is called on S, and let $\mathcal{P}'$ be the distribution over H induced when A is called on $S - \{\langle x, y\rangle\}$. Then for any $h \in H$, $\mathcal{P}(h) = (1/Z)\mathcal{P}'(h)$ if h does not err on $\langle x, y\rangle$, and $\mathcal{P}(h) = (1/Z)\exp(-\frac{1}{T})\mathcal{P}'(h)$ if h does err on $\langle x, y\rangle$. Thus the only change from $\mathcal{P}'$ to $\mathcal{P}$ is to decrease the probability of drawing an h which errs on $\langle x, y\rangle$. □(Lemma 4.4)

The key result leading to a sanity-check bound for Bayesian algorithms follows. It bounds the extent to which leave-one-out overestimates the training error in terms of the error stability of the algorithm.

THEOREM 4.5 *Let A be a Bayesian algorithm (or any other algorithm satisfying the conclusion of Lemma 4.4) that has error stability (β_1, β_2). Then for any $\gamma > 0$, leave-one-out*

$$(2\gamma + 3\sqrt{\beta_1}, 2\sqrt{\beta_1} + 4\beta_2 + 4VC(d, m, \gamma) + \sqrt{\log(1/\gamma)/m}) \tag{26}$$

-overestimates the training error for A.

In order to prove Theorem 4.5, we will first need the following lemma, which says that with respect to the randomization of a Bayesian algorithm, the leave-one-out estimate is likely to overestimate the *expected* training error.

LEMMA 4.5 *Let A be a Bayesian algorithm (or any randomized algorithm satisfying the conclusion of Lemma 4.4). Then for any fixed sample $S_m = S_{m-1} \cup \{\langle x, y\rangle\}$, with probability at least $1-\delta$ over $\vec{r}_1, \ldots, \vec{r}_m$ and $\vec{r}$,*

$$\hat{\epsilon}^A_{\mathrm{CV}}(S_m, \vec{r}_1, \ldots, \vec{r}_m) \geq \mathrm{E}_{\vec{r}}\,[\hat{\epsilon}(A(S_m, \vec{r}\,))] - \sqrt{\log(1/\delta)/m}. \tag{27}$$

PROOF: For each $\langle x_i, y_i\rangle \in S_m$, let p_i be the probability over $\vec{r}$ that $A(S_m, \vec{r}\,)$ errs on $\langle x_i, y_i\rangle$ and let p'_i be the probability over $\vec{r}_i$ that $A(S^i_m, \vec{r}_i)$ errs on $\langle x_i, y_i\rangle$. By Lemma 4.4 we know that $p'_i \geq p_i$. Then

$$\begin{aligned}
\mathrm{E}_{\vec{r}}\,[\hat{\epsilon}(A(S_m, \vec{r}\,))] &= \sum_{h\in H} \Pr_{\vec{r}}\,[A(S_m, \vec{r}\,) = h] \cdot \hat{\epsilon}(h) && (28)\\
&= \sum_{h\in H} \Pr_{\vec{r}}\,[A(S_m, \vec{r}\,) = h] \cdot \frac{1}{m}\sum_i I(h(x_i) \neq y_i) && (29)\\
&= \frac{1}{m}\sum_i \sum_{h\in H} \Pr_{\vec{r}}\,[A(S_m) = h] \cdot I(h(x_i) \neq y_i) && (30)\\
&= \frac{1}{m}\sum_i p_i. && (31)
\end{aligned}$$

Denote $(1/m)\sum_i p_i$ by $\bar{p}$, and $(1/m)\sum_i p'_i$ by $\bar{p}'$. Let e_i be a Bernoulli random variable determined by $\vec{r}_i$ which is 1 if $A(S^i_m, \vec{r}_i)$ errs on $\langle x_i, y_i\rangle$ and 0 otherwise. By definition, $\hat{\epsilon}^A_{\mathrm{CV}}(S_m, \vec{r}_1, \ldots, \vec{r}_m) = (1/m)\sum_i e_i$, and

$$\mathrm{E}_{\vec{r}_1,\ldots,\vec{r}_m}\,[\hat{\epsilon}^A_{\mathrm{CV}}(S_m, \vec{r}_1, \ldots, \vec{r}_m)] = \mathrm{E}_{\vec{r}_1,\ldots,\vec{r}_m}\,[(1/m)\sum_i e_i] = \bar{p}' \geq \bar{p} = \mathrm{E}_{\vec{r}}\,[\hat{\epsilon}(A(S_m, \vec{r}\,))]. \tag{32}$$

By Chernoff's inequality, for any α,

$$\Pr_{\vec{r}_1,\ldots,\vec{r}_m}\,[(1/m)\sum_i e_i \leq \bar{p}' - \alpha] < \exp(-2\alpha^2 m) \tag{33}$$

By setting $\alpha = (1/2)\sqrt{\log(1/\delta)/m}$, we have that with probability at least $1-\delta$ over the choice of $\vec{r}_1, \ldots, \vec{r}_m$,

$$\hat{\epsilon}^A_{\mathrm{CV}}(S_m, \vec{r}_1, \ldots, \vec{r}_m) \geq \mathrm{E}_{\vec{r}}\,[\hat{\epsilon}(A(S_m, \vec{r}\,))] - (1/2)\sqrt{\log(1/\delta)/m}. \tag{34}$$

□(Lemma 4.5)

Now we can give the proof of Theorem 4.5.

PROOF (Theorem 4.5): Because A has error stability (β_1, β_2), if we draw S_{m-1} and $\langle x, y\rangle$ at random we have probability at least $1 - \sqrt{\beta_1}$ of obtaining an S_m such that

$$\Pr_{\vec{r},\vec{r}\,'}\,[|\epsilon(A(S_m, \vec{r}\,)) - \epsilon(A(S_{m-1}, \vec{r}\,'))| \geq \beta_2] \leq \sqrt{\beta_1}. \tag{35}$$

Equation (35) relates the error when A is called on S_m and S_{m-1}. We would like to translate this to a statement relating the error when A is called on S_m twice. But if S_m satisfies Equation (35), it follows that

$$\Pr_{\vec{r},\vec{r}\,'}\left[|\epsilon(A(S_m,\vec{r}\,)) - \epsilon(A(S_m,\vec{r}\,'\,))| \geq 2\beta_2\right] \;\leq\; 2\sqrt{\beta_1}. \tag{36}$$

The reason is that if $|\epsilon(A(S_m,\vec{r}\,)) - \epsilon(A(S_m,\vec{r}\,'\,))| \geq 2\beta_2$, then $\epsilon(A(S_{m-1},\vec{r}\,''\,))$ can be within β_2 of only one of $\epsilon(A(S_m,\vec{r}\,))$ and $\epsilon(A(S_m,\vec{r}\,'\,))$, and each is equally likely to result from a call to A on S_m. From Equation (36) and Theorem 2.1, we have that with probability at least $1-\gamma-\sqrt{\beta_1}$, S_m will satisfy

$$\Pr_{\vec{r},\vec{r}\,'}\left[|\hat{\epsilon}(A(S_m,\vec{r}\,)) - \hat{\epsilon}(A(S_m,\vec{r}\,'\,))| \geq 2\beta_2 + 2\,VC(d,m,\gamma)\right] \;\leq\; 2\sqrt{\beta_1}. \tag{37}$$

If S_m satisfies Equation (37), it follows that there must be a fixed value $\hat{\epsilon}_0 \in [0,1]$ such that

$$\Pr_{\vec{r}}\left[|\hat{\epsilon}(A(S_m,\vec{r}\,)) - \hat{\epsilon}_0| \geq 2\beta_2 + 2\,VC(d,m,\gamma)\right] \;\leq\; 2\sqrt{\beta_1}. \tag{38}$$

Assuming that Equation (38) holds, how far can $\mathrm{E}_{\vec{r}}\left[\hat{\epsilon}(A(S_m,\vec{r}\,))\right]$ be from $\hat{\epsilon}_0$? The extreme cases are

$$\mathrm{E}_{\vec{r}}\left[\hat{\epsilon}(A(S_m,\vec{r}\,))\right] = (1-2\sqrt{\beta_1})(\hat{\epsilon}_0 + 2\beta_2 + 2\,VC(d,m,\gamma)) + 2\sqrt{\beta_1}\cdot 1 \tag{39}$$

and

$$\mathrm{E}_{\vec{r}}\left[\hat{\epsilon}(A(S_m,\vec{r}\,))\right] = (1-2\sqrt{\beta_1})(\hat{\epsilon}_0 - 2\beta_2 - 2\,VC(d,m,\gamma)) + 2\sqrt{\beta_1}\cdot 0. \tag{40}$$

In either case,

$$|\mathrm{E}_{\vec{r}}\left[\hat{\epsilon}(A(S_m,\vec{r}\,))\right] - \hat{\epsilon}_0| \;\leq\; 2\sqrt{\beta_1} + 2\beta_2 + 2\,VC(d,m,\gamma)) \tag{41}$$

and thus by Equation (38), with probability at least $1-\gamma-\sqrt{\beta_1}$ over the draw of S_m, S_m will be such that the probability

$$\Pr_{\vec{r}}\left[|\hat{\epsilon}(A(S_m,\vec{r}\,)) - \mathrm{E}_{\vec{r}\,'}\left[\hat{\epsilon}(A(S_m,\vec{r}\,'\,))\right]| \geq (2\beta_2 + 2\,VC(d,m,\gamma)) + (2\sqrt{\beta_1} + 2\beta_2 + 2\,VC(d,m,\gamma))\right] \tag{42}$$

is at most $2\sqrt{\beta_1}$. Combined with Lemma 4.5, we obtain that with probability at least $1-2\gamma-3\sqrt{\beta_1}$ over $S_m, \vec{r}_1,\ldots,\vec{r}_m$ and $\vec{r}$,

$$\hat{\epsilon}^A_{\mathrm{cv}}(S_m,\vec{r}_1,\ldots,\vec{r}_m\,) \;\geq\; \hat{\epsilon}(S_m,\vec{r}\,) - 2\sqrt{\beta_1} - 4\beta_2 - 4\,VC(d,m,\gamma) - \sqrt{\log(1/\gamma)/m} \tag{43}$$

as desired. □(Theorem 4.5)

Now we can give the main result of this section.

THEOREM 4.6 *Let A be a Bayesian algorithm (or any randomized algorithm satisfying the conclusion of Lemma 4.4) that has error stability (β_1,β_2). Then for any $\delta > 0$, with probability at least $1-\delta$,*

$$|\hat{\epsilon}^A_{\mathrm{cv}}(S_m,\vec{r}_1,\ldots,\vec{r}_m\,) - \epsilon(A(S_m,\vec{r}\,))| \;\leq\; \left(10\sqrt{\frac{(d+1)\,(\ln(9m/d)+1)}{m}} + 8\sqrt{\beta_1} + 5\beta_2\right)/\delta. \tag{44}$$

Here the probability is taken over the choice of S_m, and over the coin flips $\vec{r}_1,\ldots,\vec{r}_m$ and $\vec{r}$ of A on the S^i_m and S_m.

Thus, Theorem 4.6 relates the error of leave-one-out to the stability of a Bayesian algorithm: as $\beta_1, \beta_2 \to 0$, we obtain a $\tilde{O}(\sqrt{d/m})$ bound. Again, in Section 5 we show that some dependence on β_1 and β_2 is required.

4.3 Application to Linear Functions and Squared Error

In this section, we briefly describe an extension of the ideas developed so far to problems in which the outputs of both the target function and the hypothesis functions are real-valued, and the error measure is squared loss. The importance of this extension is due to the fact that for squared error, there is a particularly nice case (linear hypothesis functions) for which empirical error minimization can be efficiently implemented, and the leave-one-out estimate can be efficiently computed.

Our samples S_m now consist of examples $\langle x_i, y_i \rangle$, where $x_i \in \Re^d$ and $y_i \in [-1, 1]$. For any function $h : \Re^d \to [-1, 1]$, we now define the generalization error by $\epsilon(h) = \mathrm{E}_{\langle x,y \rangle}[(h(x) - y)^2]$, and similarly the training error becomes $\hat{\epsilon}(h) = \sum_{\langle x_i, y_i \rangle \in S} (h(x_i) - y_i)^2$. For any algorithm A, if h^i denotes $A(S^i_m)$, the leave-one-out estimate is now $\hat{\epsilon}^A_{\mathrm{CV}}(S_m) = \sum_{\langle x_i, y_i \rangle \in S_m} (h^i(x_i) - y_i)^2$.

It can be verified that in such situations, provided that a uniform convergence result analogous to Theorem 2.1 can be proved, then the analogue to Theorem 4.2 can be obtained (with essentially the same proof), where the expression $VC(d, m, \delta)$ in the bound must be replaced by the appropriate uniform convergence expression. We will not state the general theorem here, but instead concentrate on an important special case. It can easily be verified that Lemma 4.3 still holds in the squared error case: that is, if A performs (squared) training error minimization, then for any sample S_m, $\hat{\epsilon}^A_{\mathrm{CV}}(S_m) \geq \hat{\epsilon}(A(S_m))$. Furthermore, if the hypothesis space H consists of only *linear* functions $w \cdot x$, then provided the squared loss is bounded for each w, nice uniform convergence bounds are known.

THEOREM 4.7 *Let the target function be an arbitrary mapping $\Re^d \to [-B, B]$, where $B > 0$ is a constant, and let P be any input distribution over $[-B, B]^d$. Let A perform squared training error minimization over the class of all linear functions $w \cdot x$ obeying $||w|| \leq B$. Then for every $\delta > 0$, with probability at least $1 - \delta$,*

$$|\hat{\epsilon}^A_{\mathrm{CV}}(S_m) - \epsilon(A(S_m))| = O\left(\sqrt{(d/m)(\log(d/m)/\delta}\right). \tag{45}$$

Two very fortunate properties of the combination of linear functions and squared error make the sanity-check bound given in Theorem 4.7 of particular interest:

- There exist polynomial-time algorithms for performing minimization of squared training error [4] by linear functions. These algorithms do not necessarily obey the constraint $||w|| \leq B$, but we suspect this is not an obstacle to the validity of Theorem 4.7 in most practical settings.
- There is an efficient procedure for computing the leave-one-out estimate for training error minimization of squared error over linear functions [14]. Thus, it is not necessary to run the error minimization procedure m times; there is a closed-form solution for the leave-one-out estimate that can be computed directly from the data much more quickly.

More generally, many of the results given in this paper can be generalized to other loss functions via the proper generalizations of uniform convergence [7].

4.4 Other Algorithms

We now comment briefly on the application of Theorem 4.1 to algorithms other than error minimization and Bayesian procedures. As we have already noted, the only barrier to applying Theorems 4.1 to obtain bounds on the leave-one-out error that depend only on the error stability and $\tilde{O}(\sqrt{d/m})$ lies in proving that leave-one-out sufficiently overestimates the training error (or more precisely, that with high probability it does not underestimate the training error by much). We believe that while it may be difficult to prove this property in full generality for many types of algorithms, it may nevertheless often hold for natural algorithms running on natural problems.

For instance, note that in the deterministic case, leave-one-out will $(0,0)$-overestimate the training error as long as A has the stronger property that if $A(S_m)$ erred on an example $\langle x, y\rangle \in S_m$, then $A(S_m - \{\langle x, y\rangle\})$ errs on $\langle x, y\rangle$ as well. In other words, the *removal* of a point from the sample cannot *improve* the algorithm's performance on that point. This stronger property is exactly what was proven in Lemma 4.3 for training error minimization, and its randomized algorithm analogue was shown for Bayesian algorithms in Lemma 4.4. To see why this property is plausible for a natural heuristic, consider (in the squared error case) an algorithm that is performing a gradient descent on the training error over some continuous parameter space $\vec{w}$. Then the gradient with respect to $\vec{w}$ can be written as a sum of gradients, one for each example in S_m. The gradient term for $\langle x, y\rangle$ gives a force on $\vec{w}$ in a direction that causes the error on $\langle x, y\rangle$ to decrease. Thus, the main effect on the algorithm of removing $\langle x, y\rangle$ is to remove this term from the gradient, which intuitively should cause the algorithm's performance on $\langle x, y\rangle$ to degrade. (The reason why this argument cannot be turned into a proof of training error overestimation is that it technically is valid only for one step of the gradient descent.) It is an interesting open problem to prove that the required property holds for widely used heuristics.

5 Lower Bounds

In this section, we establish the following:

- That the dependence on $1/\delta$ is in general unavoidable for the leave-one-out estimate;
- That in the case of algorithms that perform error minimization, the dependence of the error of leave-one-out on the VC dimension cannot be removed without additional assumptions on tbe algorithm.
- That for *any* algorithm, the error of the leave-one-out estimate is lower bounded by the error stability;
- That there exist algorithms with *perfect* error stability for which the leave-one-out estimate is *arbitrarily* poor, and furthermore, these algorithms use a hypothesis class with *constant VC* dimension.

These last two points are especially important: they establish that while error stability is a *necessary* condition for nontrivial bounds on the leave-one-out error, it is not by itself *sufficient* even when the hypothesis class has very small *VC* dimension. Therefore, additional assumptions on the algorithm must be made. The additional assumptions made in Theorem 4.1 were sufficient training error overestimation and bounded VC dimension. In contrast, hypothesis stability alone is a sufficient condition for nontrivial bounds, but is far from necessary.

We begin with the lower bound giving an example where there is an $\Omega(1/\sqrt{m})$ chance of constant error for the leave-one-out estimate. Setting $d = 1$ in Theorem 4.1 shows that the dependence on δ given there is tight (upto logarithmic factors).

THEOREM 5.1 *There exists an input distribution P, a target function f, a hypothesis class H of VC dimension 1, and an algorithm A that minimizes the training error over H such that with probability* $\Omega(1/\sqrt{m})$, $|\hat{\epsilon}^A_{\mathrm{CV}}(S_m) - \epsilon(A(S_m))| = \Omega(1)$.

PROOF: Let the input space X consist of a single point x, and let the target function f be the probabilistic function that flips a fair coin on each trial to determine the label to be given with x. Thus, the generalization error of any hypothesis is exactly $1/2$. The algorithm A simply takes the majority label of the sample as its hypothesis. Now with probability $\Omega(1/\sqrt{m})$, the sample S_m will have a balanced number of positive and negative examples, in which case $\hat{\epsilon}^A_{\mathrm{CV}}(S_m) = 1$, proving the theorem. □(Theorem 5.1)

The following theorem shows that in the case of algorithms that perform training error minimization, the dependence of the error of the leave-one-out estimate on the VC dimension is unavoidable without further assumptions on the algorithm.

THEOREM 5.2 *For any d, there exists an input distribution P, a target function f, a hypothesis class H of VC dimension d, and an algorithm A that minimizes the training error over H such that with probability* $\Omega(1)$, $|\hat{\epsilon}^A_{\mathrm{CV}}(S_m) - \epsilon(A(S_m))| = \Omega(d/m)$.

PROOF: Let $X = [0, 1] \cup z_1, z_2$, where z_1 and z_2 are "special" points. Each of z_1 and z_2 will have weight $1/4$ under P, while the interval [0,1] will have weight $1/2$ uniformly distributed. Each function in H must label exactly one of z_1 and z_2 positively, and may be any d-switch function over [0,1]. Let h_d be the d-switch function over [0,1] in which the switches are evenly spaced $1/d$ apart.

The algorithm A behaves as follows: if the sample size m is even, A first checks if h_d minimizes the training error on those sample points in [0,1]. If so, A chooses the hypothesis that labels z_1 negatively, z_2 positively, and is h_d on [0,1]. Otherwise, A chooses to label z_1 negatively, z_2 positively, and on [0,1] chooses the "leftmost" hypothesis that minimizes the training error over [0,1] (that is, the hypothesis that minimizes the training error and always chooses its switches to be as far to the left as possible between the two sample points where the switch occurs). If the sample size is odd, A chooses the hypothesis that labels z_1 positively, z_2 negatively, and is the leftmost hypothesis minimizing the training error over [0,1]. Thus, on even samples, A has a strong bias towards choosing h_d over [0,1], but on odd samples, has no such bias.

Now suppose that the target function labels both z_1 and z_2 negatively, and labels [0,1] according to h_d. Then it is clear that with high probability, A comes within $O(1/\sqrt{m})$ of minimizing the training error (since A must label either z_1 or z_2 positively, and each of these choices will incur approximately the same number of training errors, and A always minimizes the training error on [0,1]). Furthermore, if m is even, then the true error of A's hypothesis will be within $O(1/\sqrt{m})$ of $1/4$. But is easy to verify that $\hat{\epsilon}_{\mathrm{cv}}^A(S_m)$ will exceed $1/4 + d/m$ with high probability, as desired.

Finally, the behavior of A on the points z_1 and z_2 gives the desired $\Omega(1)$ lower bound on $P[A(S_m)\Delta A(S_m^i)]$. □(Theorem 5.2)

We next show that error stability is essential for providing upper bounds on the error of the leave-one-out estimate.

THEOREM 5.3 *Let A be any algorithm which does* not *have error stability* (β_1, β_2). *Then for any* $\tau \geq 0$,

$$Pr_{S_m}[|\hat{\epsilon}_{\mathrm{cv}}^A(S_m) - \epsilon(A(S_m))| \geq \tau] \;\;\geq\;\; \frac{\beta_1 \cdot \beta_2}{2} - \tau. \tag{46}$$

PROOF: Since A does not have error stability (β_1, β_2), it is either the case that with probability at least $\beta_1/2$, $\epsilon(A(S_m)) - \epsilon(A(S_{m-1})) \geq \beta_2$, or that with probability at least $\beta_1/2$, $\epsilon(A(S_{m-1})) - \epsilon(A(S_m)) \geq \beta_2$. Without loss of generality, assume the latter is true. Let $\chi(S_m)$ be a random variable which is defined as follows: $\chi(S_m) = \hat{\epsilon}_{\mathrm{cv}}^A(S_m) - \epsilon(A(S_m))$. Thus,

$$\chi(S_m) = \hat{\epsilon}_{\mathrm{cv}}^A(S_m) - \epsilon(A(S_{m-1})) + \epsilon(A(S_{m-1})) - \epsilon(A(S_m)) \tag{47}$$

and

$$\mathrm{E}_{S_m}[\chi(S_m)] = \mathrm{E}_{S_{m-1},\langle x,y\rangle}[\hat{\epsilon}_{\mathrm{cv}}^A(S_m) - \epsilon(A(S_{m-1}))] + \mathrm{E}_{S_{m-1},\langle x,y\rangle}[\epsilon(A(S_{m-1})) - \epsilon(A(S_m))] \tag{48}$$

where $S_m = S_{m-1} \cup \{\langle x, y\rangle\}$. By Lemma 4.1 and the fact that with probability at least $\beta_1/2$, $\epsilon(A(S_{m-1})) - \epsilon(A(S_m)) \geq \beta_2$, we get that

$$\mathrm{E}_{S_m}[\chi(S_m)] \geq \frac{\beta_1}{2} \cdot \beta_2 \tag{49}$$

Let ρ be the exact probability that $|\chi(S_m)| \leq \tau$. Then

$$\begin{aligned} \frac{\beta_1\beta_2}{2} &\leq \mathrm{E}_{S_m}[\chi(S_m)] && (50)\\ &\leq \rho \cdot \tau + (1-\rho) \cdot 1 && (51)\\ &= \rho(\tau - 1) + 1 && (52) \end{aligned}$$

Thus, $\rho \leq (1 - (\beta_1\beta_2/2))(1 - \tau)$, and equivalently,

$$1 - \rho \geq \frac{(\beta_1\beta_2/2) - \tau}{1 - \tau} \geq \frac{\beta_1\beta_2}{2} - \tau \tag{53}$$

which means that with probability at least $\beta_1\beta_2/2 - \tau$, $\hat{\epsilon}_{\mathrm{cv}}^A(S_m) - \epsilon(A(S_m)) \geq \tau$. □(Theorem 5.3)

As an example, consider the application of the theorem to the case in which the probability that $|\epsilon(A(S_m)) - \epsilon(A(S_{m-1}))| \geq \beta_2$ is greater than $1/2$ for some β_2. Then by setting τ to be $\beta_2/8$, we get that with probability at least $\beta_2/8$, the error of the leave-one-out estimate is at least $\beta_2/8$.

Finally, we show that, unlike hypothesis stability, error stability alone is not sufficient to give nontrivial bounds on the error of leave-one-out even when the hypothesis class has very small *VC* dimension, and hence additional assumptions are required.

THEOREM 5.4 *There exists an input distribution P, a target function f, a hypothesis class H with constant VC dimension, and an algorithm A such that A has error stability* $(0,0)$ *with respect to P and f, but with probability* 1, $|\hat{\epsilon}^A_{\mathrm{CV}}(S_m) - \epsilon(A(S_m))| = 1/2$.

PROOF: Let $X = \{0, \ldots, N-1\}$ where N is even, let f be the constant 0 function, let P be the uniform distribution on X, and let H be the following class of (boolean) threshold functions:

$$H \stackrel{\text{def}}{=} \{h_t : t \in \{0, \ldots, N-1\}, \text{ where } h_t(x) = 1 \text{ \textit{iff} } (t + x) \bmod N < N/2\} \tag{54}$$

Clearly, the *VC* dimension of H is 2. Furthermore, for every $h \in H$, the error of h with respect to f and P is exactly $1/2$, and hence any algorithm using hypothesis class H has error stability $(0,0)$. It thus remains to show that there exists an algorithm A for which the leave-one-out estimate always has large error.

For a given sample $S_m = \{\langle x_1, y_1\rangle, \ldots, \langle x_m, y_m\rangle\}$, let $t = (\sum_{i=1}^m x_i) \bmod N$, and let $A(S_m) = h_t$, where h_t is as defined in Equation (54). Thus, the algorithm's hypothesis is determined by the sum of the (unlabeled) examples. We next compute the leave-one-estimate of the algorithm on S_m. Assume first that S_m is such that $(\sum_{i=1}^m x_i) \bmod N < N/2$. Then, by definition of A, for each x_i, the hypothesis $h^i = A(S^i_m)$ will label x_i by 1, whereas $f(x_i) = 0$. Similarly, if S_m is such that $(\sum_{i=1}^m x_i) \bmod N \geq N/2$, then for each x_i, $h^i(x_i) = 0$, which is the correct label according to f. In other words, for half of the samples S_m we have $\hat{\epsilon}^A_{\mathrm{CV}}(S_m) = 1$, which means that leave-one-out overestimates $\epsilon(A(S_m)) = 1/2$ by $1/2$, and for half of the sample it underestimates the error by $1/2$. □(Theorem 5.4)

6 Extensions and Open Problems

It is worth mentioning explicitly that in the many situations when uniform convergence bounds better than $VC(d, m, \delta)$ can be obtained [16, 6] our resulting bounds for leave-one-out will be correspondingly better as well. In the full paper we will also detail the generalizations of our results for other loss functions, and give results for k-fold cross-validation as well.

There are a number of interesting open problems, both theoretical and experimental. On the experimental side, it would be interesting to determine the "typical" dependence of the leave-one-out estimate's performance on the VC dimension for various commonly used algorithms, and also to establish the extent to which leave-one-out overestimates the training error. On the theoretical side, it would be nice to prove sanity-check bounds for leave-one-out for popular heuristics like C4.5 and backpropagation. Also, it is an open problem whether error stability together with limited VC dimension of the hypothesis class suffice to prove sanity-check bounds. Finally, there is almost certainly room for improvement in both our upper and lower bounds: our emphasis has been on the qualitative behavior of leave-one-out in terms of a number of natural parameters of the problem, not the quantitative behavior.

Acknowledgements

Thanks to Avrim Blum and Sean Holden for interesting discussions on cross-validation. We are grateful to Nabil Kahale for improving the construction in the proof of Theorem 5.4.

References

[1] L. Devroye, L. Gyröfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer Verlag, 1996.

[2] L. P. Devroye and T. J. Wagner. Distribution-free inequalities for the deleted and holdout error estimates. *IEEE Transactions on Information Theory*, IT–25(2):202–207, 1979.

[3] L. P. Devroye and T. J. Wagner. Distribution-free performance bounds for potential function rules. *IEEE Transactions on Information Theory*, IT–25(5):601–604, 1979.

[4] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.

[5] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[6] D. Haussler, M. Kearns, H.S. Seung, and N. Tishby. Rigourous learning curve bounds from statistical mechanics. *Machine Learning*, 25:195–236, 1996.

[7] David Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.

[8] S. B. Holden. Cross-validation and the PAC learning model. Research Note RN/96/64, Dept. of CS, Univ. College, London, 1996.

[9] S. B. Holden. PAC-like upper bounds for the sample complexity of leave-one-out cross validation. In *Proceedings of the Ninth Annual ACM Workshop on Computational Learning Theory*, pages 41–50, 1996.

[10] M. Kearns. A bound on the error of cross validation, with consequences for the training-test split. In *Advances in Neural Information Processing Systems 8*, pages 183–189, 1996. To Appear in *Neural Computation*.

[11] M. Kearns, R. Schapire, and L. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17:115–141, 1994.

[12] M. J. Kearns, Y. Mansour, A. Ng, , and D. Ron. An experimental and theoretical comparison of model selection methods. In *Proceedings of the Eighth Annual ACM Workshop on Computational Learning Theory*, pages 21–30, 1995. To Appear in Machine Learning, COLT95 Special Issue.

[13] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *the International Joint Conference on Artifical Intelligence*, 1995.

[14] A.J. Miller. *Subset Selection in Regression*. Chapman and Hall, 1990.

[15] W. H. Rogers and T. J. Wagner. A fine sample distribution-free performance bound for local discrimination rules. *The Annals of Statistics*, 6(3):506–514, 1978.

[16] H. S. Seung, H. Sompolinsky, and N. Tishby. Statistical mechanics of learning from examples. *Physical Review*, A45:6056–6091, 1992.

[17] V.N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, New York, 1982.