A Permutation Approach to Validation

Malik Magdon-Ismail[*] Konstantin Mertsalov[†]

Abstract

We give a permutation approach to validation (estimation of out-sample error). One typical use of validation is model selection. We establish the legitimacy of the proposed permutation complexity by proving a uniform bound on the out-sample error, similar to a VC-style bound. We extensively demonstrate this approach experimentally on synthetic data, standard data sets from the UCI-repository, and a novel diffusion data set. The out-of-sample error estimates are comparable to cross validation (CV); yet, the method is more efficient and robust, being less susceptible to overfitting during model selection.

1 Introduction

The holy grail when learning from data is an in-sample estimate of the out-sample error, i.e. *model validation.* Assume a standard setting with data

$$D = \{\mathbf{x}_i, y_i\}_{i=1}^n,$$

where $(\mathbf{x}_i, y_i)$ are sampled *iid* from the joint distribution $p(\mathbf{x}, y)$ on $\mathbb{R}^d \times \mathbb{R}$ (for regression) or on $\mathbb{R}^d \times \{\pm 1\}$ (for binary classification). Let $\mathcal{H}$ be a learning model (e.g. decision tree, k-nearest neighbor, linear regression) which produces a hypothesis $g \in \mathcal{H}$ when given D. Our discussion, though mostly generalizable, will assume that $\mathcal{H}$ is closed under negation, and that the risk is the squared error (the misclassification error rate and the least squares regression error can both be written as the squared error). Denote by e_{in} the in-sample error,

$$e_{\text{in}}(h) = \frac{1}{n}\sum_{i=1}^{n}(h(\mathbf{x}_i) - y_i)^2,$$

and by e_{out} the out-sample error,

$$e_{\text{out}}(h) = \mathbb{E}[(h(\mathbf{x}) - y)^2].$$

The expectation is over the joint distribution $p(\mathbf{x}, y)$. We wish to estimate $e_{\text{out}}(g)$, and typically $e_{\text{in}}(g)$ is not an unbiased estimate of e_{out}. For example, when g minimizes the in-sample error over $\mathcal{H}$, then for a small data set, e_{in} will typically have a large optimistic bias precisely because you are minimizing e_{in}. Instead of e_{out}, it is equally good to get an estimate of the *generalization error*[1]

$$e_{\text{gen}}(g) = e_{\text{out}}(g) - e_{\text{in}}(g),$$

which is typically positive and explicitly measures the optimism of the in-sample error.

Our goal is to present a method for estimating the generalization error, in particular, we present a permutation estimate for the generalization error. Loosely speaking, it considers learning problems related to permutations of the realized data. These permutation problems can be explicitly generated from the realized data set. For each permutation-transformed problem, we compute the expected optimism of the in-sample error. The average of this optimism over permutations is the permutation estimate for $e_{\text{gen}}(g)$. To make the idea more concrete, we will illustrate on a toy vision problem of learning to distinguish between male and female faces, using a labeled data set of images. You are using your favorite algorithm to produce a classifier. A small data set is shown in Figure 1. After learning on

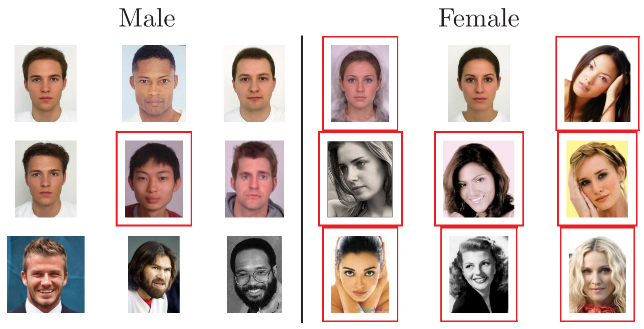


Figure 1: A sample data set for learning to distinguish male from female faces.

the data, your algorithm (hypothetically) classified as females, the images in red (in this example, the learned

[*]Rensselaer Ploytechnic Institute, Computer Science Department. magdon@cs.rpi.edu

[†]Rensselaer Ploytechnic Institute, Computer Science Department. mertsk2@cs.rpi.edu

[1]some authors use generalization error to denote what we call the out-sample error.

rule is "roundish face or long hair is female"). After learning this rule, you can classify the in-sample data. If your rule were perfect in-sample, all the faces on the right would be boxed in red. This rule is not quite perfect, and your in-sample classification error on this example data set is about 11% (2 errors). How reliable is this error of 11%? Do we expect that this rule will generalize well to out-sample, and achieve $e_{\text{out}} \approx 11\%$?

The permutation estimate would provide one estimate of the reliability. To apply the permutation estimate, we first generate a random permutation of the data, i.e. permute the labels (male or female) randomly, to obtain a permuted data set. One such realized random permutation is shown in Figure 2. This data with

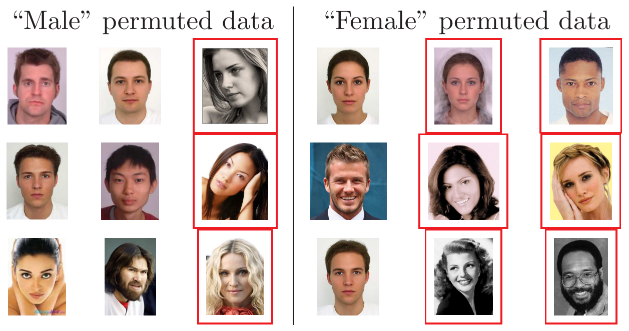


Figure 2: A permuted version of the same sample data in Figure 1. Learning on this data set should not yield anything meaningful.

the randomly permuted labels is a very confusing data set. Clearly you should not expect better than 50% performance from any algorithm out-of-sample, if the real data distribution was like this. Nevertheless, we can go ahead and learn with our learning algorithm. Since, in-sample, your learning algorithm is fitting the data, it may overfit even this randomly permuted data. In our example, suppose you learned the rule "dark skin or long hair is female"; this gives the in-sample classifications of females shown above (red boxes). In this case, on the randomly permuted data, we have made 6 errors (the 3 boxed images on the left and three unboxed images on the right), or roughly 33% classification error rate. As we already argued, for such randomly permuted data distribution, no algorithm can give out-sample error better than 50% (assuming that there are equal numbers of male and female faces), and so this achieved in-sample error rate of 33% is optimistic by about 17%. That is, the learning algorithm has overfit the permuted data by 17% (your generalization error is 17%). This level of overfitting is a single sample estimate of the overfitting capability of the model on data "of this type" – for another type of data, say digit classification, your level of overfitting on permuted data could be different. Thus, this permutation estimate of generalization error is *data dependent*, and certainly depends on the complexity of your model and learning algorithm. It seems reasonable to guess that this same level of overfitting might have affected the actual fitting on the actual data, since the actual data is, at least on the surface, of this "type". We thus apply this 17% optimism of the in-sample error obtained from looking at the permuted data to the learning on the unpermuted data; we conclude that our original classifier "roundish face or long hair is female", which had in-sample error of 11% would have an out-sample error more like 28%. The goal of this paper is to first define the permutation estimate for both regression and classification; provide theoretical justification for its use, and experimentally compare it with some other methods for validation.

Our empirical comparisons will use leave-one-out cross validation, LOO-CV, as the strawman benchmark. This is a general validation method, which is in common use. It has also been compared with most other validation methods, and hence is a valid benchmark. Our permutation estimate (as with LOO-CV) can be applied to any learning model or error metric, requiring only the ability to run the model. However, it is more efficient than LOO-CV and suffers less from the potential to be overfit during model selection, especially in regression.

Our Contribution. We give a permutation estimate for validation. Validation is one of the most important tasks when learning from data. Corresponding to the permutation estimate, we define a "permutation complexity" measure for the complexity of a learning model. We theoretically justify the permutation estimate in the context of empiral error minimization in classification problems by proving a *uniform bound* for the out-sample error in terms of this *data dependent* permutation complexity. The bound is uniform in that it applies to any joint input-output distribution.

On the practical side, we give a detailed experimental investigation to support the permutation estimate. We show that it is more efficient than the leave-one-out cross validation method, with comparable or better performance. We use LOO-CV as our strawman benchmark, since most validation methods have been compared to this benchmark, including the leave-K-out methods, and other statistical estimators. leave-K-out methods are also more efficient than LOO CV, but they will sacrifice on accuracy. Such a more extensive comparison is left for future work.

Next, we review some relevant literature, followed by a description of the permutation method and the uniform bound which justifies its use. We then give a detailed experimental investigation and conclusions. In general, detailed proofs are postponed to a full version of this paper.

Related Work. Out-sample error estimation has extensive coverage in the literature, both in the statistics community and the learning commuity. Broadly speaking there are three approaches:

(i) *Statistical methods* which try to estimate the out-sample error asymptotically in n, giving consistent estimates under certain model assumptions (for example final prediction error (FPE) [2], Generalized Cross Validation (GCV) [8] or covariance-type penalties [9, 29]). Statistical methods tend to work well when the model has been well specified. Such methods are not our primary focus. However, we do show that for linear models in the statistical regression setting, the permutation estimate reduces to an AIC-type prediction error estimate, a well known estimate.

(ii) *Bounds*. The most celebrated uniform bound on generalization error is the distribution independent bound of Vapnik-Chervonenkis (VC-bound) [27]. Since the VC-dimension may be hard to compute, empirical estimates have been suggested, [28]. The VC-bound is optimal among distribution independent bounds; however, for a particular distribution, it could be sub-optimal. Several data dependent bounds have been proposed, which can typically be estimated in-sample via optimization: maximum discrepancy [4]; Rademacher-style penalties [5, 10, 15, 16, 17, 18, 19, 21]; margin based bounds, for example [25]. Relevant to this work are Rademacher penalties. Let $\mathbf{r}$ be a sequence of *iid* binary random variables with $\mathbb{P}[+1] = \frac{1}{2}$. The Rademacher complexity is

$$\mathcal{R}(\mathcal{H}|D) = \mathbb{E}_{\mathbf{r}}\left[\frac{1}{n}\max_{h\in\mathcal{H}}\sum_{i=1}^{n} r_i h(\mathbf{x}_i)\right].$$

Generalizations to Gaussian and symmetric, bounded variance $\mathbf{r}$ have also been suggested, [5, 10] . The permutation estimate is related to a "permutation complexity" which is a Rademacher-like complexity where the r_i are obtained via a random permutation of the observed target values in D, instead of via independent uniform Bernoulli trials. Thus, the permutation estimate more closely mimics the distribution supported by the data. It also has the advantage that the estimate can be used with multi-class and regression problems, and can accomodate regularized learning algorithms.

(iii) *Sampling methods*, such as leave-one-out cross validation (LOO-CV), try to estimate the out-sample error directly. The permutation estimate uses a "sampled" data set on which to run the model and obtain the estimate; other than this superficial similarity, the estimates are inherently different. The permutation estimate is more like a Rademacher penalty in spirit.

Permutation Methods are not new to statistics [13]. They have been suggested as tests of significance for specific model selection tasks [12, 30]. In [12], the authors give a concentration inequality for such a test which involves the Rademacher complexity. We directly give a uniform bound for the out-sample error in terms of a permutation complexity, which answers a question posed in [12] which suggests that there should be a direct link between permutation statistics and generalization errors.

2 The Permutation Estimate

Consider a new learning problem, for which the input space is exactly the data examples, and the outputs are a random permutation of the observed target values. This problem mimics the learning problem at hand, in that the targets have the same joint distribution but for which there is no input-output relationship. For this new permutation-learning problem, we can compute the "out-sample" error for a test example drawn from this same random permutation distribution. If we fit the model to the data, we obtain the in-sample error of the learned function which will generally be lower than this computed out-sample error. The expected difference between the in-sample error and the computed out-sample error for the learned function is the estimate of the optimistic bias. We use this optimistic bias as the estimate of the generalization error for the particular data set D (after fitting the model to it). There is a leap of faith here: we are using the average level of optimism for this random class of problems as the measure of optimism for the *single* actual realized problem. We will provide justification for this leap of faith by relating the permutation estimate (for binary classification) to a uniform upper bound on the out-sample error. We now describe the details.

In order to specify the random permutation learning problem, $p^{\boldsymbol{\pi}}(\mathbf{x}, y)$, we will take an operational route. (We use the superscript $(\cdot)^{\boldsymbol{\pi}}$ for quantities relevant to the permutation problem.) Let

$$D^{\boldsymbol{\pi}} = (\mathbf{x}_1, y_{\pi_1}), \ldots, (\mathbf{x}_n, y_{\pi_n}),$$

be a random permutation of the data, where $\boldsymbol{\pi}$ is a random permutation of $1, \ldots, n$. This is a new learning problem in which the $\mathbf{x}$ values are unchanged but the target function $f^{\boldsymbol{\pi}}$ is a random function with $\mathbb{P}[f^{\boldsymbol{\pi}}(\mathbf{x}_i) = y_j] = \frac{1}{n}$ for $j = 1, \ldots, n$, independent of the particular $\mathbf{x}_i$. This target function has the same joint distribution of outputs on the data as the true target function, but otherwise it is independent of the input, and so there is no input-output relationship to be learned. Let $g^{\boldsymbol{\pi}}$ be the function output by the model (e.g. via empirical risk minimization). If the model "learns" a relationship, it has overfit the permuted data. In order to compute the level to which it has overfit the data, we need to first compute the out-sample error for this permuted learning problem.

2.1 Out-Sample Error for the Random Permutation Problem The out-sample error of any function $h \in \mathcal{H}$ can be computed because we know $f^{\boldsymbol{\pi}}$: $p^{\boldsymbol{\pi}}(y|\mathbf{x})$ is uniform on $\{y_i\}$. Let $\bar{y}$ and s_y^2 be the sample mean and sample variance of the target values.

Theorem 2.1.

$$e_{\text{out}}^{\boldsymbol{\pi}}(h) = s_y^2 + \frac{1}{n}\sum_{i=1}^{n}(h(\mathbf{x}_i) - \bar{y})^2.$$

Proof.

$$\begin{aligned} e_{\text{out}}^{\boldsymbol{\pi}}(h) &= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[(h(\mathbf{x}_i) - f^{\boldsymbol{\pi}}(\mathbf{x}_i))^2] \\ &= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n}\sum_{j=1}^{n}(h(\mathbf{x}_i) - y_j)^2. \end{aligned}$$

Adding and subtracting $\bar{y}$ in the summand, we have:

$$\begin{aligned} e_{\text{out}}^{\boldsymbol{\pi}}(h) &= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(h(\mathbf{x}_i) - \bar{y} + \bar{y} - y_j)^2 \\ &= s_y^2 + \frac{1}{n}\sum_{i=1}^{n}(h(\mathbf{x}_i) - \bar{y})^2. \end{aligned}$$

Notice that for this random learning problem, the best one can do is achieve an out-sample error of s_y^2 which is attained by the function which predicts the mean target value for all data points.

2.2 Estimating the Generalization Error We now compute the bias of the in-sample error (typically obtained via empirical risk minimization) on a random permutation problem. On the permuted data, the in-sample error is:

$$e_{\text{in}}^{\boldsymbol{\pi}}(g^{\boldsymbol{\pi}}) = \frac{1}{n}\sum_{i=1}^{n}(g^{\boldsymbol{\pi}}(\mathbf{x}_i) - y_{\pi_i})^2.$$

Since the generalization error is $e_{\text{gen}}^{\boldsymbol{\pi}}(g^{\boldsymbol{\pi}}) = e_{\text{out}}^{\boldsymbol{\pi}}(g^{\boldsymbol{\pi}}) - e_{\text{in}}^{\boldsymbol{\pi}}(g^{\boldsymbol{\pi}})$, we have the following result.

Theorem 2.2.

$$\begin{aligned} e_{\text{gen}}^{\boldsymbol{\pi}}(g^{\boldsymbol{\pi}}) &= \frac{2}{n}\sum_{i=1}^{n}(y_{\pi_i} - \bar{y})g^{\boldsymbol{\pi}}(\mathbf{x}_i) \\ &= \frac{2}{n}\sum_{i=1}^{n}y_{\pi_i}g^{\boldsymbol{\pi}}(\mathbf{x}_i) - 2\bar{y}\bar{g}^{\boldsymbol{\pi}}. \end{aligned}$$

Proof. Using Theorem 2.1,

$$e_{\text{gen}}^{\boldsymbol{\pi}}(g^{\boldsymbol{\pi}}) = s_y^2 + \frac{1}{n}\sum_{i=1}^{n}(g^{\boldsymbol{\pi}}(\mathbf{x}_i) - \bar{y})^2 - \frac{1}{n}\sum_{i=1}^{n}(g^{\boldsymbol{\pi}}(\mathbf{x}_i) - y_{\pi_i})^2,$$

and the result follows after some elementary algebra.

Note that $e_{\text{gen}}^{\boldsymbol{\pi}}$ is twice the covariance between the learned hypothesis and the randomly permuted target values. We wish to estimate $\mathbb{E}_{\boldsymbol{\pi}}[e_{\text{gen}}^{\boldsymbol{\pi}}(g^{\boldsymbol{\pi}})]$. Naturally we have to do this by sampling. Unfortunately, y_{π_i} are dependent (sampled without replacement), and so it is not easy to obtain a concentration result around the expectation. For *iid* sampling with replacement (see later) the generalization error on the random problem concentrates about its expectation (via McDiarmid's Inequality), so computing the generalization penalty for just a single random sample would suffice. For the permutation estimate, it should also be possible to prove a concentration inequality, however we recommend to average over M random permutations, for some reasonably sized M – even though one should be able to get an asymptotic concentration result, the real value of validation estimates is for small n when overfitting is a real concern. We used $M = 100$ for our experimental results. The results were not significantly different from $M = 1$ when n, the number of data points is large.

$$(2.1)\quad \widehat{e}_{\text{gen}}(\mathcal{H}|D) = \mathbb{E}_{\boldsymbol{\pi}}[e_{\text{gen}}^{\boldsymbol{\pi}}(g^{\boldsymbol{\pi}})] \approx \frac{1}{M}\sum_{m=1}^{M}e_{\text{gen}}^{\boldsymbol{\pi}_m}(g^{\boldsymbol{\pi}_m}).$$

(we use hat notation $\widehat{(\cdot)}$ for estimates of quantities relevant to the data D.) As with cross validation, the drawback with using more samples (larger M) is that one has to learn a final hypothesis for each permuted data set. Luckily, small M is enough for most practical purposes. The benefit over cross-validation is that for each iteration of training, one gets n estimates of the generalization error, i.e., there should be concentration around the expectation.

We now estimate the out-sample error for the distribution supported by the realized data D by

$$(2.2)\qquad \widehat{e}_{\text{out}}(g) = e_{\text{in}}(g) + \widehat{e}_{\text{gen}}(\mathcal{H}|D).$$

This generalization error estimate is data dependent, because the randomly permuted learning problems were modeled after the original data set D. This is one of the advantages of such a method over distribution independent methods (such as VC) which may not be optimal for a particular problem.

Example: Kernel Based Classification. For kernel classification, it is possible to bound the permutation estimate in terms of the trace of the kernel matrix. The techniques are exactly in correspondence with those used for the Rademacher complexity (see for example [5, 26]). This is because of the the relationship between the permutation estimate we have presented, $\widehat{e}_{\text{gen}}$, and a permutation complexity which we introduce in the next section; this permutation complexity resembles a Rademacher-like complexity.

Example: Linear Ridge Regression. Ridge regression (regression with weight decay) is popular in statistics, and many statistical estimates of the out-sample error exist. Our permutation estimate closely resembles the AIC criterion [2] in this setting.

A linear model has the form

$$\mathcal{H} = \left\{ \mathbf{w}^{\mathrm{T}}\mathbf{z} | \mathbf{w} \in \mathbb{R}^{d+1} \right\},$$

where $\mathbf{z}^{\mathrm{T}} = [1, \mathbf{x}^{\mathrm{T}}]$ is the original input prepended by a 1 for the constant. Let $\mathrm{X}^{\mathrm{T}} = [\mathbf{z}_1, \ldots, \mathbf{z}_n]$ be the input data matrix and $\mathbf{y}^{\mathrm{T}} = [y_1, \ldots, y_n]$ be the target values. Assume for simplicity that X has full column rank. If λ is the ridge regression parameter, then the optimal regularized fit minimizing $e_{\text{in}}(\mathbf{w}) + \lambda \mathbf{w}^{\mathrm{T}}\mathbf{w}$ is given by

$$\mathbf{w}_{\text{in}} = (\mathrm{X}^{\mathrm{T}}\mathrm{X} + \lambda \mathrm{I})^{-1}\mathrm{X}^{\mathrm{T}}\mathbf{y},$$

and the in-sample predictions are $\hat{\mathbf{y}} = \mathrm{S}(\lambda)\mathbf{y}$, where,

$$\mathrm{S}(\lambda) = \mathrm{X}(\mathrm{X}^{\mathrm{T}}\mathrm{X} + \lambda \mathrm{I})^{-1}\mathrm{X}^{\mathrm{T}}.$$

(We will typically suppress the dependence on λ.)

For linear models, we can obtain the permutation estimate without sampling, via an analytic computation. We want $\mathbb{E}_{\boldsymbol{\pi}}[e^{\boldsymbol{\pi}}_{\text{gen}}(g^{\boldsymbol{\pi}})]$. The predictions on the permuted data are $\hat{\mathbf{y}}^{(\boldsymbol{\pi})} = \mathrm{S}\mathbf{y}^{(\boldsymbol{\pi})}$, where S is independent of the permutation $\boldsymbol{\pi}$ because it only depends on the $\mathbf{x}$ values in the data. Using Theorem 2.2:

$$e^{\boldsymbol{\pi}}_{\text{gen}}(g^{\boldsymbol{\pi}}) = \frac{2}{n}\sum_{i,j=1}^{n} \mathrm{S}_{ij}(y_{\pi_i}y_{\pi_j} - \bar{y}y_{\pi_i}) \tag{2.3}$$

To compute $\mathbb{E}_{\boldsymbol{\pi}}[e^{\boldsymbol{\pi}}_{\text{gen}}(g^{\boldsymbol{\pi}})]$ for Theorem 2.3, we will need the next result on the correlations.

Lemma 2.1.

$$\mathbb{E}_{\boldsymbol{\pi}}[y_{\pi_i}] = \bar{y}, \qquad \mathbb{E}_{\boldsymbol{\pi}}[y_{\pi_i}y_{\pi_j}] = \begin{cases} \bar{y}^2 + s_y^2 & i = j, \\ \bar{y}^2 - \frac{1}{n-1}s_y^2 & i \neq j. \end{cases}$$

Proof. The first equality follows immediately because y_{π_i} is uniform over $\mathbf{y}$. To prove the second, note that

$$\begin{aligned} \mathbb{E}[y_{\pi_i}, y_{\pi_j}] &= \sum_{i=1}^{n}\sum_{j\neq i} \frac{1}{n(n-1)} y_i y_j, \\ &= \frac{1}{n(n-1)}\sum_{i=1}^{n}\left(\sum_{j=1}^{n} y_i y_j - y_i^2\right), \\ &= \frac{n}{n-1}\bar{y}^2 - \frac{1}{n-1}\overline{y^2}, \end{aligned}$$

from which the result follows, because $s_y^2 = \overline{y^2} - \bar{y}^2$.

Define $\hat{\sigma}_y^2 = \frac{n}{n-1}s_y^2$, the unbiased variance estimate for y, and let $\mathbf{1}$ be the n–vector of ones.

Theorem 2.3.

$$\widehat{e}_{\text{out}}(g) = e_{\text{in}}(g) + \frac{2\hat{\sigma}_y^2}{n}\left(\text{trace}(\mathrm{S}) - \frac{\mathbf{1}^{\mathrm{T}}\mathrm{S}\mathbf{1}}{n}\right).$$

Proof. Using (2.3),

$$\begin{aligned} \widehat{e}_{\text{gen}}(\mathcal{H}|D) &= \mathbb{E}_{\boldsymbol{\pi}}[e^{\boldsymbol{\pi}}_{\text{gen}}(g^{\boldsymbol{\pi}})] \\ &= \frac{2}{n}\sum_{i,j=1}^{n} \mathrm{S}_{mn}(\mathbb{E}_{\boldsymbol{\pi}}[y_{\pi_i}y_{\pi_j}] - \bar{y}\,\mathbb{E}_{\boldsymbol{\pi}}\,[y_{\pi_i}]). \end{aligned}$$

The result follows using Lemma 2.1 after some algebra.

With no regularization ($\lambda = 0$), S is a projection matrix (projecting onto the columns of X). Since the first column of X is a column of ones (representing a constant term in the regression), then $\mathrm{S}\mathbf{1} = \mathbf{1}$ and the permutation estimate becomes

$$\widehat{e}_{\text{out}} = e_{\text{in}} + \frac{2\hat{\sigma}_y^2 d}{n}.$$

Thus, with no regularization, the permutation estimate reduces to an AIC-like criterion, and the term $2\hat{\sigma}_y^2 d/n$ is a penalty for model complexity. Observe that $(\text{trace}(\mathrm{S}) + 1 - \frac{1}{n}\mathbf{1}^{\mathrm{T}}\mathrm{S}\mathbf{1})$ is an "effective number of parameters". The choice $d_{\text{eff}} = \text{trace}(\mathrm{S})$ has been proposed in the literature (see for example [6]).

Extensions: A simpler alternative to the permutation estimate (sampling the y_i without replacement) is to sample with replacement. Computationally, this is slightly simpler, but it does not preserve the joint distribution of the target values (they are now independent at two different points $\mathbf{x}_i, \mathbf{x}_j$). There are some advantages to this approach, for example one can obtain easy concentration results for $e^{\text{samp}}_{\text{gen}}$ about its expectation. This implies that even one random learning problem could be enough for estimating the optimism penalty. Our entire discussion will generalize easily to this setting. For linear ridge regression,

$$e^{\text{samp}}_{\text{out}}(g) = e_{\text{in}}(g) + \frac{2s_y^2\text{trace}(\mathrm{S})}{n},$$

and when $\lambda = 0$, $e^{\text{samp}}_{\text{out}}(g) = e_{\text{in}}(g) + \frac{2s_y^2(d+1)}{n}$.

The permutation estimate can be extended to different risk metrics, multi-class and regression problems in a seemless way. The analogues of Theorems 2.1 and 2.2 would need to be computed; their form may not be as simple, but nonetheless it is a straightforward task. It is not immediate how to extend the Rademacher penalty to regression, since, for example, the maximizer of covariance for a linear model is not well defined.

3 Bounding Out-Sample Classification Error

For classification ($y \in \{\pm 1\}$), the estimate $\widehat{e}_{\text{gen}}$ is closely related to a Rademacher-like "permutation complexity". We now give a bound for the out-sample error using this permutation complexity for empirical risk minimization. We will adapt the standard ghost sample approach in VC-type proofs and the symmetrization trick in [11] which has greatly simplified VC-style proofs. In general, high probability results are with respect to the distribution over data sets. Our main bounding tool will be McDiarmid's inequality:

LEMMA 3.1. (MCDIARMID, [22]) *Let $X_i \in A_i$ be independent; suppose $f : \prod_i A_i \mapsto \mathbb{R}$ satisfies*

$$\sup_{\substack{\mathbf{x}\in\prod_i A_i \\ z\in A_j}} |f(\mathbf{x}) - f(x_1,\ldots,x_{j-1},z,x_{j+1},\ldots,x_n)| \le c_j,$$

for $j = 1,\ldots,n$. Then, for all $t > 0$, $\mathbb{P}[f(X_1,\ldots,X_n) - \mathbb{E}f(X_1,\ldots,X_n) \ge t] \le e^{-2t^2/\sum_{i=1}^n c_i^2}$.

3.1 Permutation Complexity

The out-sample permutation complexity of a model is:

$$\mathcal{P}_{\text{out}}(\mathcal{H}) = \mathbb{E}_{D,\boldsymbol{\pi}}\left[\max_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^n y_{\pi_i} h(\mathbf{x}_i)\right],$$

where the expectation is over the data $D = (\mathbf{x}_1, y_1),\ldots,(\mathbf{x}_n, y_n)$ and a random permutation $\boldsymbol{\pi}$. For a particular sample D, the in-sample permutation complexity is

$$\mathcal{P}_{\text{in}}(\mathcal{H}|D) = \mathbb{E}_{\boldsymbol{\pi}}\left[\max_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^n y_{\pi_i} h(\mathbf{x}_i)\right].$$

We note that the above definitions can be extended to models not closed under negation by considering $\max_{h\in\mathcal{H}} \left|\frac{1}{n}\sum_{i=1}^n y_{\pi_i} h(\mathbf{x}_i)\right|$. Let D' differ from D only in one example, $(\mathbf{x}_j, y_j) \to (\mathbf{x}'_j, y'_j)$.

LEMMA 3.2. $|\mathcal{P}_{\text{in}}(\mathcal{H}|D) - \mathcal{P}_{\text{in}}(\mathcal{H}|D')| \le \frac{4}{n}$.

Proof. Consider any permutation $\boldsymbol{\pi}$; the sum $\sum_{i=1}^n y_{\pi_i} h(\mathbf{x}_i)$ changes by at most 4 (only two points are affected). Thus, the maximum over $\mathcal{H}$ changes by at most 4 and the lemma follows.

Lemma 3.2 together with McDiarmid's inequality implies a concentration of $\mathcal{P}_{\text{in}}$ about $\mathcal{P}_{\text{out}}$, which means we can work with $\mathcal{P}_{\text{in}}$ instead of the unknown $\mathcal{P}_{\text{out}}$.

COROLLARY 3.1. *With probability at least $1-\delta$,*

$$\mathcal{P}_{\text{out}}(\mathcal{H}) \le \mathcal{P}_{\text{in}}(\mathcal{H}|D) + 4\sqrt{\frac{1}{2n}\ln\frac{1}{\delta}}.$$

Since $e_{\text{in}}(h) = 2 - \frac{2}{n}\sum_{i=1}^n y_i h(\mathbf{x}_i)$, the empirical risk minimizer maximizes the correlation:

$$\max_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^n y_{\pi_i} h(\mathbf{x}_i) = \frac{1}{n}\sum_{i=1}^n y_{\pi_i} g^{\boldsymbol{\pi}}(\mathbf{x}_i).$$

Thus, using Theorem 2.2:

THEOREM 3.1. $\mathcal{P}_{\text{in}}(\mathcal{H}|D) = \frac{1}{2}\widehat{e}_{\text{gen}}(\mathcal{H}|D) + \bar{y}\,\mathbb{E}_{\boldsymbol{\pi}}\left[\bar{g}^{\boldsymbol{\pi}}\right]$

3.2 Uniform Bound for the Out-Sample Error

LEMMA 3.3. *Let $\mathbf{r}$ be an arbitrary ± 1 sequence. Then, with probability at least $1-\delta$:*

$$\begin{aligned} &\max_{h\in\mathcal{H}}\{e_{\text{out}}(h) - e_{\text{in}}(h)\} \\ \le\ &\mathbb{E}_{D,D'}\left[\max_{h\in\mathcal{H}} \frac{2}{n}\sum_{i=1}^n r_i(y_i h(\mathbf{x}_i) - y'_i h(\mathbf{x}'_i))\right] + \sqrt{\frac{8\ln\frac{1}{\delta}}{n}}. \end{aligned}$$

Proof. The proof uses the standard ghost sample and symmetrization arguments typical of modern generalization error proofs, so we do not give the details (see for example [5, 26]).

Lemma 3.3 holds for an *arbitrary* sequence $\mathbf{r}$ which is independent of D, D'. An immediate corollary is that we can take the expectation with respect to $\mathbf{r}$, for *arbitrarily* distributed $\mathbf{r}$.

For each permutation $\mathbf{y}^{\boldsymbol{\pi}}$, let $r_i^{\boldsymbol{\pi}} = y_i^{\boldsymbol{\pi}} y_i$; then, $y_i^{\boldsymbol{\pi}} = r_i^{\boldsymbol{\pi}} y_i$, i.e., we have a mapping from permutations to a multiset of ± 1 sequences $S_{\mathbf{y}} = \{\mathbf{r}^{\boldsymbol{\pi}}\}_{\boldsymbol{\pi}}$. If the distribution of $\mathbf{r}$ is uniform on $S_{\mathbf{y}}$, then $\mathbf{ry}$ is uniform over the permutations of $\mathbf{y}$. We say that $S_{\mathbf{y}}$ generates the permutations on $\mathbf{y}$. Similarly, we can define $S_{\mathbf{y}'}$, the generator of permutations on the ghost sample $\mathbf{y}'$. Unfortunately, $S_{\mathbf{y}}, S_{\mathbf{y}'}$ depend on D, D'. We can overcome this by introducing a second ghost sample D'' to "approximately" generate the permutations for $\mathbf{y}, \mathbf{y}'$, ultimately allowing us to prove the main theorem.

THEOREM 3.2. *With probability at least $1-\delta$,*

$$\max_{h\in\mathcal{H}}\{e_{\text{out}}(h) - e_{\text{in}}(h)\} \le 4\mathcal{P}_{\text{out}}(\mathcal{H}) + O\left(\sqrt{\tfrac{1}{n}\ln\tfrac{1}{\delta}}\right).$$

The bound, being uniform, applies to the empirical risk minimizer g.

COROLLARY 3.2.

$$e_{\text{out}}(g) \le e_{\text{in}}(g) + 2\widehat{e}_{\text{gen}}(\mathcal{H}|D) + 4\bar{y}\mathbb{E}_{\boldsymbol{\pi}}[\bar{g}^{\boldsymbol{\pi}}] + O\left(\sqrt{\tfrac{1}{n}\ln\tfrac{1}{\delta}}\right)$$

Commentary: The bound in Theorem 3.2 is with $\mathcal{P}_{\text{out}}$ (or $\mathcal{P}_{\text{in}}$ for n large). Corollary 3.2 justifies using the permutation estimate $\widehat{e}_{\text{gen}}(\mathcal{H}|D)$ *for near empirical risk minimizers* – this bound can fail for non-empirical risk minimizers, though in practice it works well for regularized algorithms. For non-balanced data, $\widehat{e}_{\text{gen}}(\mathcal{H}|D)$ may not be a good estimate of the permutation complexity, where as $\mathcal{P}_{\text{in}}$ always works. The permutation complexity is a complexity measure for the model. However, as the example with linear ridge regression illustrates, the permutation estimate uses a penalty which is algorithm specific, and can directly account for regularization in the learning algorithm. We show empirically that the permutation estimate works even for regularized algorithms.

Proof. (sketch) Let D'' be a second ghost data set and let $S_{\mathbf{y}''}$ be the generator of permutations for $\mathbf{y}''$. From Lemma 3.3, taking the expectation w.r.t $\mathbf{r}'' \in S_{\mathbf{y}''}$, the first term on the RHS becomes

$$\mathbb{E}_{D,D'} \frac{1}{n!} \sum_{\boldsymbol{\pi}_m} \left[\max_{h \in \mathcal{H}} \frac{2}{n} \sum_{i=1}^{n} r''_i (y_i h(\mathbf{x}_i) - y'_i h(\mathbf{x}'_i)) \right].$$

The first summation is over the $n!$ permutations, each permutation $\boldsymbol{\pi}_m$ inducing a particular sequence $\mathbf{r}''(\boldsymbol{\pi}_m)$. Consider the sequences $\mathbf{r}, \mathbf{r}'$ corresponding to the permutations $\{\boldsymbol{\pi}_m\}_m$. The next lemma (proof omitted) will relate the expectation over permutations in the second ghost data set to the permutations over D, D'.

LEMMA 3.4. *There is a one-to-one mapping from the sequences $\{\mathbf{r}''(\boldsymbol{\pi}_m)\}_m$ to $\{\mathbf{r}(\boldsymbol{\pi}_m)\}_m$ such that with probability at least $1 - \delta$,*

$$\left| \frac{1}{n} \sum_{i=1}^{n} (r''_i - r_i(\mathbf{r}'')) y_i h(\mathbf{x}_i) \right| \le c \sqrt{\frac{\ln 1/\delta}{n}},$$

for every $\mathbf{r}''(\boldsymbol{\pi}_m)$ and every $h \in \mathcal{H}$. Similarly, there exists such a mapping from $\{\mathbf{r}''\}$ to $\{\mathbf{r}'\}$.

We can rewrite the internal summand as

$$(r''_i - r_i(\mathbf{r}'') + r_i(\mathbf{r}'')) y_i h(\mathbf{x}_i) - (r''_i - r'_i(\mathbf{r}'') + r'_i(\mathbf{r}'')) y'_i h(\mathbf{x}'_i).$$

By Lemma 3.4, up to error terms which are $O(\sqrt{\ln(1/\delta)/n})$, we have:

$$\mathbb{E}_{D,D'} \frac{1}{n!} \sum_{\boldsymbol{\pi}_m} \left[\max_{h \in \mathcal{H}} \frac{2}{n} \sum_{i=1}^{n} (r_i(\mathbf{r}'') y_i h(\mathbf{x}_i) - r'_i(\mathbf{r}'') y'_i h(\mathbf{x}'_i)) \right].$$

The summation is still over permutations (because the mapping is one-to-one). Since $\mathcal{H}$ is closed under negation, we bound by

$$\mathbb{E}_D \frac{1}{n!} \sum_{\boldsymbol{\pi}_m} \left[\max_{h \in \mathcal{H}} \frac{2}{n} \sum_{i=1}^{n} r_i y_i h(\mathbf{x}_i) \right] + \mathbb{E}_{D'} \frac{1}{n!} \sum_{\boldsymbol{\pi}_m} \left[\max_{h \in \mathcal{H}} \frac{2}{n} \sum_{i=1}^{n} r'_i y'_i h(\mathbf{x}'_i) \right],$$

which is just $4\mathcal{P}_{\text{out}}(\mathcal{H}|D)$. We get the result after collecting error terms.

Extensions. The analysis can accomodate models not closed under negation by using absolute values in the permutation complexity. All the analysis can be extended to sampling with replacement rather than permutations (sampling without replacement). One could bound $\mathcal{P}_{\text{out}}$ for VC function classes, showing that this data dependent bound is no worse than a VC-type bound. One could also use techniques for bounding Rademacher complexity to bounding permutation complexity on specific domains (eg for decision trees, neural networks, kernel methods, [5]).

Any approximate estimate of the out-sample error (which satisfies some bound of this form) can be used for model selection, after adding a (small) penalty for the "complexity of model selection" (see [4]). In practice, this penalty for the complexity of model selection is ignored (as in [4]).

4 Experiments

We compared the permutation estimate in a variety of settings to test both its validation capability, and its performance during model selection. For regression, we used linear models and conducted an extensive comparison of several methods, including several statistical estimates on simulated data (we averaged results over more than 1 million experiments). For classification on real data, we used a held out test set for evaluation (averaged results over 1,000 random test-training splits). In general we compare LOO-CV and the permutation estimate, which are both generally applicable, requiring only the ability to learn with a model. For classification, we also compared the Rademacher penalty, used in a similar vein to the permutation complexity by applying the algorithm to the Rademacher variables (it is not possible to compute the empirical Rademacher complexity as that would entail an empirical risk minimization). We use LOO-CV as the strawman benchmark because it is widely used; it should be noted that some of the pitfalls of the LOO-CV are its computation complexity, which is addressed by the leave-K-out CV estimate (with $K < n$); we also compared with 10-fold cross validation when learning was costly. The tradeoffs between leave-K-out and LOO have been studied, and in general most validation methods have been compared to LOO. Hence LOO is a valid benchmark. For linear regression, we compared some statistical estimates (eg. Akaike's final prediction error (FPE)), as well as the VC penalty measure.

4.1 Data

We considered a number of data sets:

Simulated Data – Regression. The input $\mathbf{x}$ is uniform on $[-1, 1]$. The target function is a polynomial of degree d_f, which we write as

$$f(x) = \sum_{i=0}^{d_f} a_i L_i(x),$$

where $L_i(x)$ are the Legendre polynomials:

$$y_i = f(x_i) + \sigma\epsilon_i,$$

where ϵ_i are *iid* standard Normals and σ is the noise variance. We consider polynomial models of different order (order selection) and different regularization parameters λ in the ridge regression (regularization parameter selection).

Simulated Data – Classification. We considered the 2-D problem in Figure 4 (green is +1).

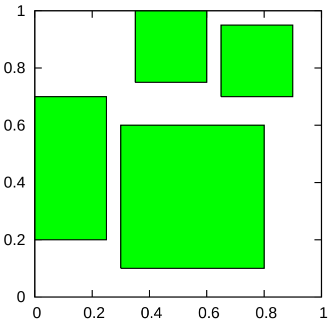


Figure 4: Simulated classification problem.

We used k-nearest neighbor (complexity parameter k) and decision trees with greedy splitting according to information gain (complexity is number of leaves). This is a variation of ID3 [24].

In the simulated setting we are able to generate data points controling the amount of noise and the proportion of positive and negative examples in the data. This allows us to conduct a systematic comparison of the different validation methods in different settings.

Real Data – Classification. We evaluated the methods on various UCI ML repository data sets [3], using decision trees (DT) and k-nearest neighbors (k-NN). We used Abalone, Mammographic Mass, Spambase, Transfusion and WDBC data (see Table 1 for some statistics on the data sets). For Abalone data the decision task was to predict [age $\geq$ 10]. We also used a novel large data set for predicting the diffusion of YouTube videos in the LiveJournal blogosphere – the task is to determine if a video will spread to at least M blogs, based on features of the early diffusion (e.g. the in/out-degrees of initially affected bloggers). This is an extremely hard and noisy prediction problem. For a detailed description of the data set, see [20]

Data	$\lvert D\rvert$	Pos./Neg.	Dim.
Abalone	4,177	0.50/0.50	7
Ionosphere	351	0.64/0.36	34
M.Mass	830	0.51/0.49	5
Parkinsons	195	0.75/0.25	22
P.I.D.	768	0.35/0.75	8
Spambase	4,601	0.39/0.61	57
Transfusion	748	0.23/0.77	4
WDBC	569	0.37/0.63	30
Diffusion	3,554	0.22/0.78	16

Table 1: Real datasets used in experiments.

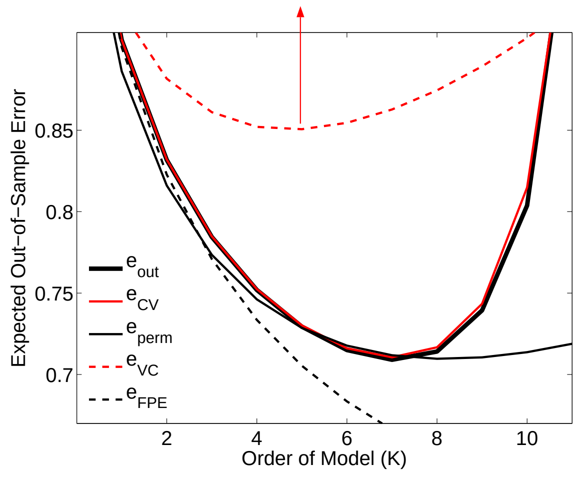
4.2 Results on Regression

In the simulated linear ridge regression setting, we compared the permutation and LOO-CV estimates together with Akaike's FPE [1] (a statistical estimate from information theory) and a VC-penalty for regression([7, (4.27b)] – all these estimates are computed analytically. These penalties are defined in terms of an effective model complexity, d_{eff} (we used $d_{\text{eff}} = \text{trace}(\mathrm{S}(\lambda))$). The graphs in Figure 3 compare the validation estimates as the model varies along two dimensions: (a) polynomial regression models with different polynomial order and no regularization; (b) polynomial ridge regression with 5th order polynomials, and different regularization parameters λ.

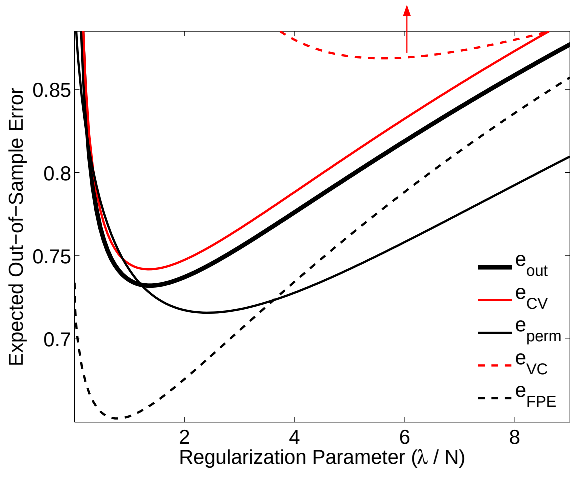
LOO-CV is very hard to beat as an estimate of the out-sample error. However, for model selection, by far the most important use of validation, the LOO-CV estimate performs poorly. We tabulate the performance of model selection below. We evaluate the validation estimates using the fractional regret (as compared to the optimal model). Table 2 summarizes the performance of model selection using each of the validation estimates. LOO-CV experiences huge regret. This is because in practice it pays to be conservative: LOO-CV pays too big a price when it is wrong (erring on the side of too complex a model). It is already known that LOO-CV can suffer from overfitting during model selection, and

Validation	Order Selection		λ Selection	
Estimate	Regret	Avg Order	Regret	Avg. $\frac{\lambda}{N}$
LOO-CV	540	9.29	18.8	23.1
Perm.	**185**	**7.21**	5.96	9.57
VC	508	5.56	**3.50**	**125**
FPE	9560	11.42	51.3	18.1

Table 2: Comparison of validation estimates for polynomial order selection and regularization parameter selection. The regret is the percentage loss versus the optimal selection which minimizes e_{out}.



(a) Different Polynomial Order.



(b) Different Regularization Parameter.

Figure 3: Performance of validation measures as compared to the true expected out-sample error. LOO-CV gives very good estimates of the out-sample error on average.

several approaches to "regularizing" cross validation have been suggested (see for example [23],[14, 1-σ rule, pg 216]). We take a simple approach to regularizing model selection by removing the choice $\lambda = 0$ from the set of models. This corresponds to "regularizing" the model selection methods, because by removing the $\lambda = 0$ model, we are not allowing any validation method to be too aggressive. This makes sense, because whenever noise is present in the data, which is the typical case, one expects regularization to help. The results of model selection among models not containing $\lambda = 0$ is shown in the Table 3. Note that all methods benefit from a more

Validation Estimate	Regularized λ Selection Regret
LOO-CV	0.44
Perm.	**0.39**
VC	0.42
FPE	0.87

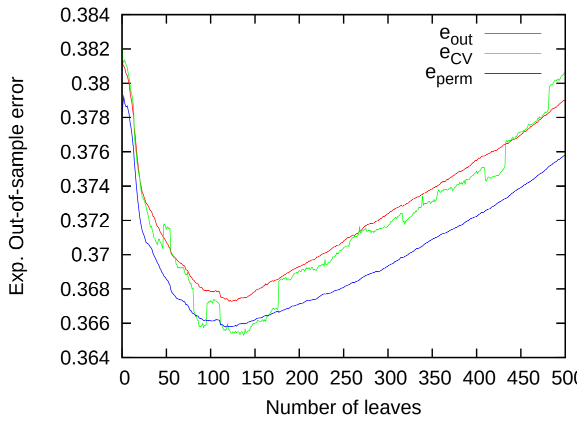
Table 3: Comparison of validation methods for selecting the regularization parameter when the unstable case $\lambda = 0$ is excluded from the set of possible models.

regularized model selection which excludes the $\lambda = 0$ model. The permutation estimate is the best performer.

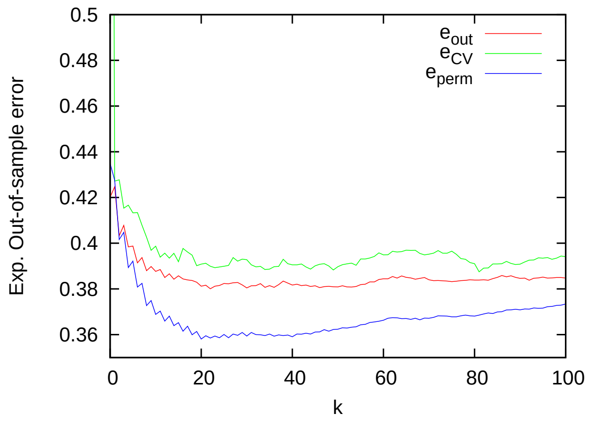
4.3 Results on Classification We compared LOO-CV, the permutation, and the Rademacher approaches to validation. We used a single permutation or and Rademacher sequence to evaluate the estimates. When learning is costly (decision trees), all validation estimates had a fixed number of learning episodes (1000 in our case). We also considered another extreme where only a very few learning episodes are allowed (10 in our case); in such a setting, 10-fold cross validation is a common alternative to LOO-CV, so we also compared 10-fold validation. When learning is cheap (e.g. k-NN), we do not limit the number of cross validation points. The plots in Figure 5 illustrate the validation estimates for the diffusion prediction data set.

In general, we observe that the CV estimate is more accurate but less stable than the permutation estimate. Our experimental results when using the respective estimates for model selection are shown in Table 4. The permutation and Rademacher approaches are comparable, and generally better than LOO-CV in decision tree learning. In case of k-NN classification LOO-CV showed best results and permutation estimate was a close second. For simulated data, we used a balanced data set with 300 data points and 40% noise. We used the simulated example as a testbed to study how the noise level affected the validation estimates. For small noise, the LOO-CV estimate suffers most, and for large noise, approaching 50%, all methods equalize with LOO-CV slightly better. The permutation estimate slightly dominates the Rademacher estimate. These results are shown in Table 6. We also did a systematic study of how the positive to negative example imbalance affects the estimates. We found that when the imbalance in the data increases, the permutation estimate dominates the Rademacher estimate.

For DTs, learning is costly, so we fixed the number of learning episodes, and LOO-CV is generally inferior.



(a) LOO-CV vs. Permutation (DT)



(b) LOO-CV vs. Permutation (k-NN).

Figure 5: Comparison of the permutation and LOO-CV estimates on a diffusion prediction data set with the decision tree and nearest neighbor learning models. The permutaion estimate is well behaved and smooth where as the LOO-CV estimate is very unstable.

Data	Decision Trees			k-Nearest Neighbor		
	LOO-CV	Perm.	Rad.	LOO-CV	Perm.	Rad.
Abalone	0.05	**0.02**	**0.02**	0.04	0.04	0.04
Ionosphere	0.17	**0.16**	0.17	**0.17**	0.70	0.83
M.Mass	0.09	**0.05**	**0.05**	**0.09**	0.11	0.11
Parkinsons	**0.24**	0.34	0.41	**0.25**	0.33	0.43
Pima Diabetes	0.09	**0.07**	**0.07**	**0.11**	**0.11**	0.14
Spambase	0.07	**0.06**	0.07	**0.19**	0.43	0.55
Transfusion	0.10	**0.08**	0.09	**0.09**	0.12	0.19
WDBC	**0.20**	0.23	0.34	**0.21**	0.34	0.51
Diffusion	0.04	0.03	**0.02**	0.04	0.06	**0.03**
Simulated	0.16	**0.15**	**0.15**	0.21	0.21	0.21

Table 4: Average regret after model selection. For decision trees, the permutation estimate performs best, and for k-NN, LOO-CV seems best, with the permutation estimate a clear second.

For k-NNs, when learning is not costly, LOO-CV works well. For both DTs and k-NNs, the permutation estimate generally dominates the Rademacher approach. We have also tried using the uniform bounds for model selection instead of the estimates, and that is generally inferior.

Table 5 shows the performance of model selection when the number of learning episodes was limited to 10. The permutation and Rademacher results are similar to using a large number of learning episodes (Table 4), a by-product of the concentration of the empirical complexity measure about its out-sample expectation. The performance of CV significantly deteriorated this is to be expected, because the validation estimate is computed using only 10 test points in total; naturally it will be a very unstable estimate. The alternative to this type of validation using cross validation is to use 10-fold validation; this has the advantage of using all the data as test points. The tradeoff is that this now estimates the out-sample performance of learning with $\frac{9}{10}$th of the data, which is only an approximation to learning with all the data. From Table 5, it is not clear which of the two CV measures is better, but the permutation estimate is now dominant. The ability of a method to perform under such limited conditions (very few learning episodes) is important when model construction is expensive and model application is cheap (e.g. Decision Trees). The permutation method works well with just one additional learning episode, producing an estimate using all the data. The permutation estimate also

Noise(%)	LOO-CV	Perm.	Rad.
5	0.30	**0.28**	**0.28**
10	0.28	**0.27**	**0.27**
15	0.28	**0.25**	**0.25**
20	0.28	**0.26**	**0.26**
25	0.26	**0.25**	**0.25**
30	**0.24**	**0.24**	**0.24**
35	0.24	**0.23**	0.24
40	**0.25**	0.26	0.26
45	**0.24**	0.26	0.26

Table 6: Regret for model selection on simulated data as a function of noise level. For small noise, the permutation and Rademacher methods dominate, with the permutation method slightly better. For large noise, all methods significantly deteriorate to large regret with LOO-CV slightly better.

seems to dominate the Rademacher approach because it takes more properties about the distribution into account, without sacrificing stability.

5 Discussion

Cross validation is general, but may be unstable and computationally intensive, which has led to a variety of alternate validation methods. We presented a permutation estimate (Equations (2.1), (2.2) and Theorem 2.2), which, as with cross validation, is generally applicable, requiring only the ability to learn with a model. It can be applied to: classification or regression; general models and error metrics; any data distribution; and, most importantly to any learning algorithm for a given model. Other data dependent estimates, such as the Rademacher penalty, penalize the complexity of a hypothesis set; the permutation complexity which we also introduced to justify the permutation estimate for empirical error minimization is similar to the Rademacher complexity in that it is a complexity measure for a hypothesis set. The permutation estimate, however, also captures the properties of the learning algorithm (for example regularized empirical risk minimization). The permutation estimate is general, whereas the Rademacher complexity only applies to classification and empirical error minimization.

For classification, with empirical risk minimization, the permutation estimate gives similar guarantees as the Rademacher penalty, and indeed the empirical performances are comparable, though the permutation estimate appears slightly better. We gave a detailed empirical comparison of the permutation estimate with other estimates, including statistical estimates which are taylored for the linear regression setting. The permutation estimate, in most cases, was either the best or comparable to the best for model selection.

When learning is costly, cross validation is prohibitively expensive; we have demonstrated on simulated data, the diffusion data and some standard UCI data sets that the permutation approach gives better results for a fixed (small) number of costly learning attempts. Another way to improve upon LOO-CV when learning is too costly is the K-fold CV procedure; 10-fold is a typical choice. We explicitly compared the 10-fold CV with LOO-CV with 10 learning episodes and the permutation estimate. The permutation estimate dominates. Naturally, our experimental study is nowhere near exhaustive. The tradeoffs between K-fold CV and LOO-CV have been extensively studied elsewhere; the conclusions of these studies are not cut and dry; for example, the choice of K is a sticky point, and there are no clear guidelines. We see here that it is not even clear whether a LOO-CV estimate from only 10 left out test points (corresponding to 10 learning episodes) is dominated by 10-fold CV. Fortunately, with the permutation estimate, we can use all the data in constructing the validation estimate. Further, we gave theoretical guarantees, and in experiments the permutation estimate comparable or better than CV, especially when model construction is costly. Conclusion: the permutation estimate provides a more stable alternative to CV which is as generally applicable, more efficient, and (for classification) comes with a uniform bound. There are numerous directions for further investigation, such as more detailed comparison K-fold validation methods, the dependence on n, etc. These are directions for future work.

References

[1] H. Akaike. Statistical predictor identification. *Annals Inst. Stat. Math.*, 22:203–217, 1970.

[2] H. Akaike. A new look at the statistical model identification. *IEEE Trans. Aut. Cont.*, 19:716–723, 1974.

[3] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[4] P. L. Bartlett, S. Boucheron, and G. Lugosi. Model selection and error estimation. *Machine Learning*, 48:85–113, 2002.

[5] P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

[6] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[7] V. Cherkassky and F. Mulier. *Learning From Data: Concepts, Theory, and Methods.* Wiley, 2007.

[8] P. Craven and G. Wahba. Smoothing noisy data with

Data	n	Decision Trees				k-Nearest Neighbor		
		LOO-CV	10-fold	Perm.	Rad.	LOO-CV	Perm.	Rad.
Abalone	3,132	0.12	0.13	**0.02**	**0.02**	0.24	**0.09**	0.12
Ionosphere	263	0.24	0.21	**0.18**	0.19	**0.49**	0.75	0.84
M.Mass	667	0.23	0.13	**0.06**	**0.06**	0.15	**0.11**	0.12
Parkinsons	144	**0.25**	0.31	0.34	0.40	0.34	**0.32**	0.44
Pima Diabetes	576	0.18	0.18	**0.07**	**0.07**	0.16	0.12	0.15
Spambase	3,450	0.28	0.09	**0.07**	**0.07**	0.44	**0.43**	0.54
Transfusion	561	0.19	0.13	**0.08**	0.09	0.17	**0.12**	0.19
WDBC	426	0.31	0.40	**0.24**	0.37	0.55	**0.33**	0.50
Diffusion	2,665	0.13	0.04	0.03	**0.02**	0.09	0.06	**0.04**

Table 5: Average regret for model selection with number of learning episodes limited to 10. The permutation estimate is now clearly dominant. There is no clear pattern between 10-fold CV and 10 learning episodes being evaluated on one point each.

spline functions. *Numerische Mathematik*, 31:377–403, 1979.

[9] B. Efron. The estimation of prediction error: Covariance penalties and cross-validation. *Journal of the American Statistical Association*, 99(467):619–632, September 2004.

[10] M. Fromont. Model selection by bootstrap penalization for classification. *Machine Learning*, 66(2-3):165–207, 2007.

[11] E. Giné and J. Zinn. Some limit theorems for empirical processes. *Annals of Prob.*, 12:929–989, 1984.

[12] P. Golland, F. Liang, S. Mukherjee, and D. Panchenko. Permutation tests for classification. *Learning Theory*, pages 501–515, 2005.

[13] P. Good. *Permutation, parametric, and bootstrap tests of hypotheses*. Springer, 2005.

[14] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[15] M. Kääriäinen and T. Elomaa. Rademacher penalization over decision tree prunings. In *In Proc. 14th European Conference on Machine Learning*, pages 193–204, 2003.

[16] V. Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, 2001.

[17] V. Koltchinskii and D. Panchenko. Rademacher processes and bounding the risk of function learning. In E. Gine, D. Mason, and J. Wellner, editors, *High Dimensional Prob. II*, volume 47, pages 443–459, 2000.

[18] F. Lozano. Model selection using rademacher penalization. In *Proc. 2nd ICSC Symp. on Neural Comp.*, 2000.

[19] G. Lugosi and A. Nobel. Adaptive model selection using empirical complexities. *Annals of Statistics*, 27:1830–1864, 1999.

[20] M. Magdon-Ismail, K. Mertsalov, and M. Goldberg. Learning to predict diffusion in the blogosphere. In *submitted to ICMLA*, 2009.

[21] P. Massart. Some applications of concentration inequalities to statistics. *Annales de la Faculté des Sciencies de Toulouse*, X:245–303, 2000.

[22] C. McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

[23] A. Ng. Preventing "overfitting" of cross-validation data. In *Proc. 14th International Conference on Machine Learning (ICML)*, pages 245–253, 1997.

[24] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, March 1986.

[25] J. Shawe-Taylor, P. L. Bartlett, R. C. Williamson, and M. Anthony. Structural risk minimization over data dependent hierarchies. *IEEE Transactions on Information Theory*, 44:1926–1940, 1998.

[26] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Camb. Univ. Press, June 2004.

[27] V. N. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their pr obabilities. *Theory of Probability and its Applications*, 16:264–280, 1971.

[28] V. N. Vapnik, E. Levin, and Y. Le Cun. Measuring the VC-dimension of a learning machine. *Neural Computation*, 6(5):851–876, 1994.

[29] J. Wang and X. Shen. Estimation of generalization error: random and fixed inputs. *Statistica Sinica*, 16:569–588, 2006.

[30] S. Wiklund, D. Nilsson, L. Eriksson, M. Sjostrom, S. Wold, and K. Faber. A randomization test for pls component selection. *Journal of Chemometrics*, 21(10,11), 2007.