EECS 553: Machine Learning (ECE) **University of Michigan**

Model Selection

Winter 2023 **Clayton Scott**

1 Tuning Parameters

Many machine learning algorithms have parameters that are not determined by the algorithm itself. Examples include regularization parameters, kernel parameters, and the k in k-nearest neighbors. These *tuning parameters*, or *hyperparameters*, influence algorithm performance, and therefore it is important to set them correctly.

The problem of setting these parameters is called *(hyper)parameter tuning* or *model selection*. A model in this context is the class of functions that our machine learning algorithm selects a function from. For example, consider the constrained version of ridge regression: $\min_{\boldsymbol{w},b} \frac{1}{n}\sum_i (y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b)^2 \quad s.t. \quad \|\boldsymbol{w}\|_2^2 \leq s$. Each value of s determines a model, namely the set of function $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$ where $\|\boldsymbol{w}\|_2^2 \leq s$. Parameters usually determine the complexity of a model, and therefore model selection amounts to striking the right balance between *overfitting* and *underfitting*. Model selection is an issue in all machine learning. In these notes, we will focus on supervised learning problems in which the performance is measured by a risk. The ideas presented here can be applied in other learning settings as long as it is possible to estimate performance for each model under consideration.

2 Estimating Risk

Let $\boldsymbol{\theta}$ denote the parameter vector to be tuned. Let $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$ denote the training data, and for each $\boldsymbol{\theta}$ under consideration, let $\widehat{f}_{\boldsymbol{\theta}}$ denote the learned model when trained on the full training data. Throughout these notes, we imagine that the same learning algorithm is used to compute $\widehat{f}_{\boldsymbol{\theta}}$ for all $\boldsymbol{\theta}$. Typically, $\boldsymbol{\theta}$ is selected from a pre-determined grid of values. For example, if the algorithm is a support vector machine with Gaussian kernel, then $\boldsymbol{\theta} = [C\ \sigma]^T$, and we might want to select the best $\boldsymbol{\theta}$ from the grid $(C, \sigma) \in \{1, 10, 100, 1000\} \times \{10^{-2}, 10^{-1}, 1, 10^1, 10^2\}$.

We would ideally like to solve

$$\min_{\boldsymbol{\theta}} R(\widehat{f}_{\boldsymbol{\theta}}).$$

where R denotes a risk of interest. Not knowing the joint distribution of $(\boldsymbol{X}, Y)$, we can't compute R and so we aim to estimate it instead. Therefore the crux of model selection is estimating the risk.

If $R(f) = \mathbb{E}[L(\boldsymbol{Y}, f(\boldsymbol{X}))]$, a natural estimate is the *training error*:

$$\widehat{R}_{TR}(\widehat{f}_{\boldsymbol{\theta}}) := \frac{1}{n}\sum_{i=1}^{n} L(y_i, \widehat{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\ )$$

which is also known as the *apparent error*, *resubstitution error*, or empirical risk. However, if $\boldsymbol{\theta}$ determines model complexity, then solving

$$\min_{\boldsymbol{\theta}} \widehat{R}_{TR}(\widehat{f}_{\boldsymbol{\theta}})$$

selects a complex model because that will lead to the smallest training error. For example, if the parameter is the degree of the polynomial in least squares polynomial regression, the largest allowed degree will be selected since this leads to the smallest sum of squared errors. Thus, minimizing the training error may lead to overfitting, and so we must look for other error estimates.

Note 1. In classification, even if our algorithm uses a surrogate loss, our ultimate interest is risk based on the 0-1 loss, and this is the risk we aim to esimate.

Note 2. In the example of least squares polynomial regression, if the degree exceeds the sample size, there are actually infinitely many empirical risk minimizers. If an appropriate empirical risk minimizer is selected, such as the one with minimum norm $\|\boldsymbol{w}\|_2$, the solution will non overfit.

3 Holdout

The holdout error estimate is defined as follows. Partition the training data as

$$\underbrace{(\boldsymbol{x}_1, y_1),\ \ldots\ , (\boldsymbol{x}_m, y_m)}_{\text{used to fit the model } \widehat{f}_{\boldsymbol{\theta}}}\ \underbrace{(\boldsymbol{x}_{m+1}, y_{m+1}),\ \ldots\ , (\boldsymbol{x}_n, y_n)}_{\text{used to estimate } R(\widehat{f}_{\boldsymbol{\theta}})},$$

perhaps after randomly shuffling the data. The second part is often called the *validation dataset*.

For each candidate $\boldsymbol{\theta}$, let $\widehat{f}_{\boldsymbol{\theta}}$ be the model learned using $(\boldsymbol{x}_i, y_i),\ 1 \le i \le m$. The *holdout error estimate* is

$$\widehat{R}_{HO}(\widehat{f}_{\boldsymbol{\theta}}) = \frac{1}{n-m} \sum_{i=m+1}^{n} L(y_i, \widehat{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)).$$

The holdout method selects $\boldsymbol{\theta}$ by solving

$$\min_{\boldsymbol{\theta}} \widehat{R}_{HO}(\widehat{f}_{\boldsymbol{\theta}}).$$

4 Cross-Validation

If data is scarce, the holdout error has the disadvantage that it limits the training examples that are used for fitting the model. *Cross-validation* attempts to improve the sample size used for model fitting.

Let K be an integer, $1 \le K \le n$. Let $I_1, \ldots, I_K$ be a partition of $\{1, 2, \ldots, n\}$ such that $|I_j| \approx \frac{n}{K}\ \forall j$. For example, if $n = 10, K = 3$, a partition might be

$$I_1 = \{2, 6, 7\}$$
$$I_2 = \{1, 3, 5, 10\}$$
$$I_3 = \{4, 8, 9\}.$$

Typically the partition is obtained by randomly shuffling the training data and then grouping consecutive batches of $\lceil \frac{n}{K} \rceil$ points. The sets I_k are called *folds* or *validation sets*.

Let $\widehat{f}_{\boldsymbol{\theta}}^{(j)}$ denote the fitted model based on parameter $\boldsymbol{\theta}$ and training data $\{(\boldsymbol{x}_i, y_i)\}_{i \notin I_j}$, and define an error estimate based on I_j as

$$\widehat{R}^{(j)}(\widehat{f}_{\boldsymbol{\theta}}^{(j)}) = \frac{1}{|I_j|} \sum_{i \in I_j} L(y_i, \widehat{f}_{\boldsymbol{\theta}}^{(j)}(\boldsymbol{x}_i)).$$

Each $\widehat{R}^{(j)}$ is a holdout estimate.

The *K-fold cross-validation error estimate* is

$$\widehat{R}_{CV}(\widehat{f}_{\boldsymbol{\theta}}) := \frac{1}{K} \sum_{j=1}^{K} \widehat{R}^{(j)}(\widehat{f}_{\boldsymbol{\theta}}^{(j)}).$$

Although each $\widehat{R}^{(j)}$ is unbiased as an estimate of $R(\widehat{f}_{\boldsymbol{\theta}}^{(j)})$, $\widehat{R}_{CV}(\widehat{f}_{\boldsymbol{\theta}})$ is not an unbiased estimate of $R(\widehat{f}_{\boldsymbol{\theta}})$ (recall $\widehat{f}_{\boldsymbol{\theta}}$ is the model trained on the *full* training data). Larger K tends to produce error estimates with

lower bias,[1] while smaller K tends to produce error estimates with smaller variance. Higher variance is more likely to lead to overfitting (because it is more likely that a suboptimal parameter value will appear to be the best), so smaller values of K are preferable.[2] Common choices of K are 5 and 10 although these values are somewhat arbitrary.

For support vector machines, an extensive experimental study has argued that 2-fold CV is appropriate for data sets of 1000 or more data points, while a 3-fold CV is appropriate for smaller data sets.[3]

Remark 1. When $K = n$ it's called *leave-one-out cross-validation* (LOOCV). One attractive feature of LOOCV is that it can be directly incorporated into the learning algorithm for some methods such as kernel ridge regression[4]. On the other hand, LOOCV has high variance and is therefore more likely to lead to overfitting.

Remark 2. To reduce the variance of the estimate and if the resources are available, it is good to compute several CV estimates based on different random partitions and average them.

Remark 3. In classification, the sets I_k should be chosen so that the proportions of different classes in each fold are the same as in the full sample. This is known as *stratified* cross-validation.

5 The Bootstrap

Let $B \geq 1$ be an integer. For $b = 1, \ldots, B$, let I_b be a subset of $\{1, 2, ..., n\}$ of size n obtained by *uniformly sampling with replacement*. Such a set is called a *bootstrap sample*.

Example 1. If $n = 6$ and $B = 2$, we might randomly draw the sets

$$I_1 = \{3, 4, 5, 4, 1, 2\}$$
$$I_2 = \{1, 2, 6, 6, 2, 5\}.$$

Define $\widehat{f}_{\boldsymbol{\theta}}^{(b)}$ to be the fitted model based on the data $\{(\boldsymbol{x}_i, y_i)\}_{i \in I_b}$, where $\{(\boldsymbol{x}_i, y_i)\}_{i \in I_b}$ includes the repeated elements from I_b. Also define the error estimate based on I_b to be

$$\widehat{R}^{(b)}(\widehat{f}_{\boldsymbol{\theta}}^{(b)}) = \frac{1}{|\{j : j \notin I_b\}|} \sum_{i \notin I_b} L(y_i, \widehat{f}_{\boldsymbol{\theta}}^{(b)}(\boldsymbol{x}_i))$$

Essentially, the model $\widehat{f}_{\boldsymbol{\theta}}^{(b)}$ is based on the bth bootstrap sample, and evaluated on points not in that sample. Averaging these estimates across all boostrap samples gives the *bootstrap error estimate*,

$$\widehat{R}_{BS}(\widehat{f}_{\boldsymbol{\theta}}) = \frac{1}{B} \sum_{b=1}^{B} \widehat{R}^{(b)}(\widehat{f}_{\boldsymbol{\theta}}^{(b)}).$$

Larger B typically leads to better performance, and so the value of B is typically based on computational considerations. $B = 200$ is a recommended minimum. For large scale applications the "bag of little bootstraps" is a computationally efficient alternative.[5]

Remark 4. $\widehat{R}_{BS}$ tends to be pessimistic, so it is common to combine the bootstrap error estimate with the training error estimate, which tends to be optimistic. A recommendation is

$$0.632\widehat{R}_{BS}(\widehat{f}_{\boldsymbol{\theta}}) + 0.368\widehat{R}_{TR}(\widehat{f}_{\boldsymbol{\theta}})$$

called the "0.632 bootstrap."

[1] The *bias* of an estimator is the difference between the true parameter and the expected value of the estimator. If the bias is zero, the estimator is said to be *unbiased*, otherwise it is said to be *biased*.

[2] https://jmlr.org/papers/volume11/cawley10a/cawley10a.pdf.

[3] https://www.jmlr.org/papers/volume18/16-174/16-174.pdf.

[4] https://jmlr.org/papers/volume11/cawley10a/cawley10a.pdf.

[5] https://people.eecs.berkeley.edu/∼jordan/papers/blb_icml2012.pdf.

Remark 5. The *balanced bootstrap* chooses $I_1, \ldots, I_B$ such that each $i \in \{1, 2, \ldots, n\}$ occurs exactly B times.

For more on the bootstrap, see Efron and Tibshirani, *An Introduction to the Bootstrap*.

6 Final Fitting

With all of the above approaches, once $\boldsymbol{\theta}$ has been selected, the final model $\widehat{f_{\boldsymbol{\theta}}}$ is obtained by applying the learning algorithm on the full training data, using the selected $\boldsymbol{\theta}$.

7 Beyond Grid Search

The term "grid search" refers to computing an error estimate for a *predetermined* grid of parameter values, and selecting the parameter minimizing that estimate. While grid search is the most common approach, recent research has argued that the set of parameter values can be refined as an algorithm is trained. For example, one might start with a coarse grid, run an iterative algorithm for a few iterations, and refine the search around those parameter values that seem most promising. Search on "Bayesian optimization" and the Hyperband algorithm for more information.