Machine Learning Course - CS-433

Generalization, Model Selection, and Validation

Oct 8, 2019

EPFL

Motivation

Assume that your friend has trained a model on some data and now claims to have found the "perfect" regression function f. How can you verify this claim and have confidence that f will have good performance? This leads us to the question of ==*generalization.*==

As a second motivation consider the following problem. We have seen in ridge regression that the regularization parameter $\lambda > 0$ can be tuned to reduce overfitting by reducing model complexity,

$$
\min_{\mathbf{w}} \quad \frac{1}{2N} \sum_{n=1}^{N} (y_n - \mathbf{x}_n^\top \mathbf{w})^2 \quad + \quad \lambda \|\mathbf{w}\|^2
$$

The parameter λ is a *hyper*-parameter.

In a similar manner, we can enrich the model complexity, by augmenting the feature vector $\mathbf{x}$. E.g., consider a polynomial feature expansion. Here the degree d is a hyperparameter.

To see a final example consider neural nets. Here we have tens or hundreds of hyperparameters: architechture, width, depth, type of the network etc.

In all these cases we are faced with the same problem: how do we choose these hyperparameters? This is the ==*model selection*== problem.

Data Model and Learning Algorithm

In order to give a meaningful answer to the above questions we first need to specify our data model.

We assume that there is an (unknown) underlying distribution $\mathcal{D}$, with range $\mathcal{X} \times \mathcal{Y}$. The data set we see, call it S, consists of independent samples from $\mathcal{D}$:

$$S = \{(\mathbf{x}_n, y_n) \text{ i.i.d. } \sim \mathcal{D}\}_{n=1}^{N}.$$

The *learning algorithm* takes the data and outputs a model within the class of models that it is given. Often this is done by optimising a cost function. We have seen that we can use e.g. (stochastic) gradient descent or least-squares for the ridge-regression model as an efficient way of implementing this learning. Write $f_S = \mathcal{A}(S)$, where $\mathcal{A}$ denotes the learning algorithm.

If we want to indicate that f_S also depends on parameters of the model, e.g., the λ in the ridge regression model, we can add a subscript to write $f_{S,\lambda}$.

True Error, Emperical Error, and Training Error

Given a model f, how can we assess if f is any good? We should compute the *expected* error over all samples chosen according to $\mathcal{D}$, i.e., we should compute

$$L_{\mathcal{D}}(f) = \mathbb{E}_{\mathcal{D}}[\ell(y, f(\mathbf{x}))],$$

where $\ell(\cdot, \cdot)$ is our loss function. E.g., for ridge regression

$$\ell(y, f(\mathbf{x})) = \tfrac{1}{2}(y - f(\mathbf{x}))^2,$$

The quantity $L_{\mathcal{D}}(f)$ has many names: *(true/expected) (risk/loss/*
This is the quantity we are fundamentally interested in, but
we cannot compute it since $\mathcal{D}$ is not known.

But we are given some data S. It is therefore natural to
compute the equivalent *empirical* quantity

$$L_S(f) = \frac{1}{|S|} \sum_{(\mathbf{x}_n, y_n) \in S} \ell(y_n, f(\mathbf{x}_n)). \qquad (1)$$

This is called the *(empirical) (risk/loss/error)*.

There is one added complication. Assume that we are given
the data S. If we first learn the model from S, i.e., we
compute $f_S = \mathcal{A}(S)$ and then we compute the empirical risk
of f_S using the *same data S* then in fact we are computing

$$L_S(f_S) = \frac{1}{|S|} \sum_{(\mathbf{x}_n, y_n) \in S} \ell(y_n, f_S(\mathbf{x}_n)).$$

This is called the *training (risk/loss/error)* and we have
already discussed in previous lectures that this training error
might not be representative of the error we see on "fresh"
samples.

The reason that $L_S(f_S)$ might not be close to $L_{\mathcal{D}}(f_S)$ is of
course overfitting.

So let us summarize. If somebody hands us a function f then
there is the true risk $L_{\mathcal{D}}(f)$ associated to f, but we cannot
in general compute it. If we have some data S that was not
used to determine f then we can compute the empirical risk
of f with respect to S. This is denoted by $L_S(f)$. We will
shortly discuss the relationship between $L_{\mathcal{D}}(f)$ and $L_S(f)$.
And if S was used to learn f, then we denote it by f_S and
call the resulting empirical error the training error, $L_S(f_S)$.

Splitting the Data and Test Error

Before we go on and explore the relationship between the true risk and the empirical risk, let us first see how we can adress the potential overfitting problem that ocurs if we train and test the model on the same data.

Problem: Validating model on the same data subset we trained it on!

Fix: Split the data into a *training* and a *test* set (a.k.a. *validation* set), call them S_{train} and S_{test}, respectively.

We apply the learning algorithm $\mathcal{A}$ to the training set S_{train} and compute the function $f_{S_{\text{train}}}$. We then compute the error on the test set, i.e.,

$$L_{S_{\text{test}}}(f_{S_{\text{train}}}) = \frac{1}{|S_{\text{test}}|} \sum_{(y_n, \mathbf{x}_n) \in S_{\text{test}}} \ell(y_n, f_{S_{\text{train}}}(\mathbf{x}_n)).$$

This is called the *(test/validation) (risk/loss/error)*. Since S_{test} is a "fresh" sample we can hope that $L_{S_{\text{test}}}(f_{S_{\text{train}}})$ is close to the quantity $L_{\mathcal{D}}(f_{S_{\text{train}}})$.

But we payed a price. We had to split the data and now have less data both for the learning as well as the validation (test) task.

"Cross validation" as described below is more efficient in using data, but hard to analyze.

True Error, Test Error, and Generalization Error

We now get back to our original question. Assume that we have a model f, and that our loss function $\ell(\cdot, \cdot)$ is bounded,

lets say in $[a, b]$. We are given a test set S_{test} chosen i.i.d. from the underlying distribution $\mathcal{D}$ (and this test set was *not* used to train the model). The test/empnrial error is

$$L_{S_{\text{test}}}(f) = \frac{1}{|S_{\text{test}}|} \sum_{(\mathbf{x}_n, y_n) \in S_{\text{test}}} \ell(y_n, f(\mathbf{x}_n)).$$

The true error is

$$L_{\mathcal{D}}(f) = \mathbb{E}_{(y,\mathbf{x}) \sim \mathcal{D}}[\ell(y, f(\mathbf{x}))].$$

How far are these apart? This is called the *generalization error* and it is given by

$$|L_{\mathcal{D}}(f) - L_{S_{\text{test}}}(f)|.$$

First note that in expectation they are the same, i.e.,

$$L_{\mathcal{D}}(f) = \mathbb{E}_{S_{\text{test}} \sim \mathcal{D}}[L_{S_{\text{test}}}(f)], \tag{2}$$

where the expectation is over the samples of the test set. But we need to worry about the variation. We claim that

$$\mathbb{P}\left[\left| L_{\mathcal{D}}(f) - L_{S_{\text{test}}}(f) \right| \geq \sqrt{\frac{(b-a)^2 \ln(2/\delta)}{2|S_{\text{test}}|}} \right] \leq \delta. \tag{3}$$

Insights: The error decreases as $\mathcal{O}(1/\sqrt{|S_{\text{test}}|})$ with the number test points. The more data points we have therefore, the more confident we can be that the empirical loss we measure is close to the true loss. If we want δ to be smaller we only need to increase the size of the test set slightly. *Proof of* (3):

Since we assumed that each data sample $(\mathbf{x}_n, y_n)$ in the test set S_{test} is chosen independently, the associated losses $\ell(y_n, f(\mathbf{x}_n))$, given a fixed model f, are also i.i.d. random variables, taking values in $[a, b]$ by assumption. Call each such loss Θ_n. The expected value of $\Theta_n = \ell(y_n, f(\mathbf{x}_n))$ is equal to the true loss

$$L_{\mathcal{D}}(f) = \mathbb{E}[\ell(y_n, f(\mathbf{x}_n))].$$

The empirical loss on the other hand is equal to the average of $|S_{\text{test}}|$ such i.i.d. values.

We want to know the chance that the empirical loss $L_{S_{\text{test}}}(f)$ deviates from its true value by more than a given constant. This is a classical problem addressed in the following lemma.

Lemma 0.1 (Chernoff Bound). *Let $\Theta_1, \ldots, \Theta_N$ be a sequence of i.i.d. random variables with mean $\mathbb{E}[\Theta]$ and range $[a, b]$. Then, for any $\varepsilon > 0$,*

$$\mathbb{P}\left[\left|\frac{1}{N}\sum_{n=1}^{N}\Theta_n - \mathbb{E}[\Theta]\right| \geq \varepsilon\right] \leq 2e^{-2N\varepsilon^2/(b-a)^2}.$$

Using Lemma 0.1 let us show (3). Equating $2e^{-2|S_{\text{test}}|\varepsilon^2/(b-a)^2}$ with δ we get that $\varepsilon = \sqrt{\frac{(b-a)^2\ln(2/\delta)}{2|S_{\text{test}}|}}$ as claimed.

How to Use This Bound

Let us summarize. Assume at first that somebody hands you a function f and you have a data set S. Then you can assert

that

$$\mathbb{P}\left[L_{\mathcal{D}}(f) > L_S(f) + \sqrt{\frac{(b-a)^2 \ln(2/\delta)}{2|S|}} \right] \leq \delta.$$

In words, we can compute a probabilistic upper bound on the true risk since both $L_S(S)$ as well as the error term can be computed with the data at hand.

In the more general case, we are given a data set S and we want to first learn a function and then compute an upper bound on the true risk of the learned function. In this case we should split S into $S = S_{\text{train}} \cup S_{\text{test}}$. We then let $f_{S_{\text{train}}} = \mathcal{A}(S_{\text{train}})$. Then we can assert that

$$\mathbb{P}\left[L_{\mathcal{D}}(f_{S_{\text{train}}}) > L_{S_{\text{test}}}(f_{S_{\text{train}}}) + \sqrt{\frac{(b-a)^2 \ln(2/\delta)}{2|S_{\text{test}}|}} \right] \leq \delta.$$

So also in this case do we get a a computable probabilistic upper bound. In the second case we pay a price for splitting the data and the bound will in general be less tight.
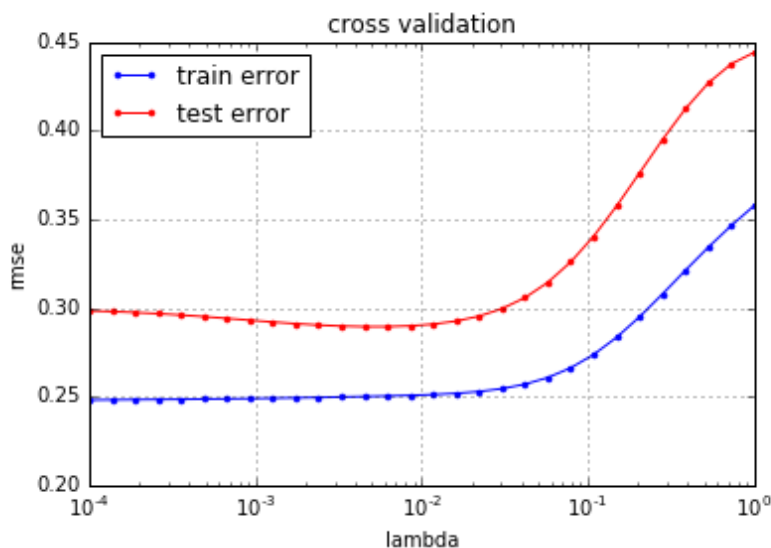
Model selection

Let us now get to our second, related, problem. We are looking for a way to select the hyperparameters of our model, like the parameter λ for the ridge regression problem.

We split our data into a training set S_{train} and a test set S_{test} and we think of them as having been generated independently and sampled according to the underlying but unknown distribution $\mathcal{D}$. We have in addition a set of values for a parameter of the model, e.g., the parameter λ in the ridge

regression problem. Let these values be λ_k, $k = 1, \ldots, K$. To keep things simple we assume that K is some finite value. We run the learning algorithm K times on the same training set S_{train} to compute the K prediction functions $f_{S_{\text{train}}, \lambda_k}$. For each such prediction function we compute the test error $L_{S_{\text{test}}}(f_{S_{\text{train}}, \lambda_k})$. We then choose that value of the parameter λ which gives us the smallest such test error.

In the figure below, we plot the test error (red) as well as the training error (blue) for many values of λ (grid search).

The same procedure can is applied to select other model hyperparameters, such as the degree in case of a polynomial feature expansion.

Model Selection Based on Test Error

If for all models $f_{S_{\text{train}}, \lambda_k}$ the test error was exactly equal to the true error then it makes sense that we pick the best of these models. But we have seen that even if we compare the

test to the true error of a single function in general there is a difference (the generalization error). And in the model selection problem we use the same test set K times to compute the test error for each of the K models. How confident can we be that the suggested procedure gives us meaningful results? This is a slight extension of our previous analysis. Assume hence that we have K models f_k given as candidates, $k = 1, \ldots, K$. Again assume that our loss function $\ell(\cdot, \cdot)$ is bounded in $[a, b]$. Given a test set S_{test} chosen i.i.d. from $\mathcal{D}$. what is the maximum generalization error, i.e., what is the maximum difference between the true error and the test error?

Similarly as in the case of a single model (3), we claim that we can now bound the maximum deviation for all K candidates, by

$$\mathbb{P}\left[\max_k \left| L_{\mathcal{D}}(f_k) - L_{S_{\text{test}}}(f_k) \right| \geq \sqrt{\frac{(b-a)^2 \ln(2K/\delta)}{2|S_{\text{test}}|}} \right] \leq \delta. \tag{4}$$

Insights: The error decreases as $\mathcal{O}(1/\sqrt{|S_{\text{test}}|})$ with the number test points. Now that we test K hyper-parameters, our error only goes up by a very small factor which is proportional to $\sqrt{\ln(K)}$. So we can test many different models without incurring a large penalty.

The proof of this statement follows (3), which has answered the special case of $K = 1$.

For a general K, if we check the deviations for K independent samples and ask for the probability that for at least one such

sample we get a deviation of at least ε then by the union bound this probability is at most K times as large as in the case where we are only concerned with a single instance. I.e., the upper bound becomes $2Ke^{-2|S_{\text{test}}|\varepsilon^2/(b-a)^2}$.

Hence, equating now $2Ke^{-2|S_{\text{test}}|\varepsilon^2/(b-a)^2}$ with δ we get that $\varepsilon = \sqrt{\frac{(b-a)^2\ln(2K/\delta)}{2|S_{\text{test}}|}}$ as stated.

Let

$$k^* = \operatorname{argmin}_k L_{\mathcal{D}}(f_k).$$

In words, f_{k^*} is that function that has the smallest *true* risk. Further, let

$$\hat{k} = \operatorname{argmin}_k L_{S_{\text{test}}}(f_k).$$

In words, $\hat{k}$ is that function that has the smallest *empirical* risk. Then a little thought shows that

$$\mathbb{P}\left[L_{\mathcal{D}}(f_{\hat{k}}) > L_{\mathcal{D}}(f_{k^*}) + 2\sqrt{\frac{(b-a)^2\ln(2K/\delta)}{2|S_{\text{test}}|}} \right] \le \delta. \quad (5)$$

In words, if we choose the "best" function according to the empirial risk then its true risk is not too far away from the true risk of the optimal choice.

Extension to Infinitely Many Model Choices The basic idea of this bound can be carried over to the case where we have infinitely many models. In this case a more sophisticated concept, called the VC-dimension, is used: as long as we have models with a *finite* VC-dimension then the bound one can show has the same form, with K replaced by the VC dimension.

Cross-validation

Splitting the data once into two parts (one for training and one for testing) is not the most efficient way to use the data. Cross-validation is a better way:

K-fold cross-validation is a popular variant. Randomly partition the data into K groups. Now train K times. Each time leave out exactly one of the K groups for testing and use the remaining $K-1$ groups for training. Average the K results.

Note: Have used all data for training, and all data for testing, and used each data point the same number of times.
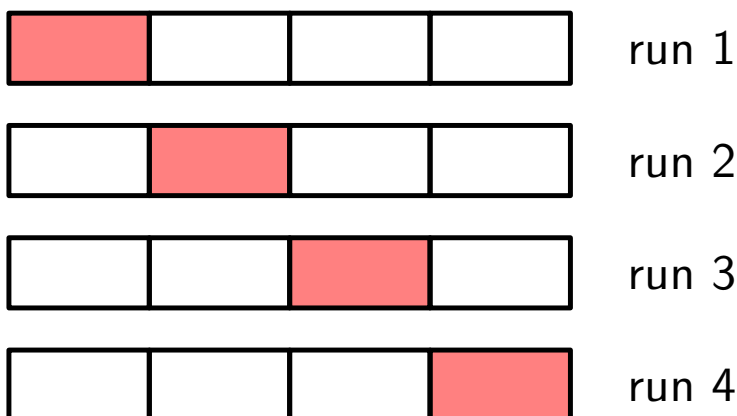
Cross-validation returns an unbiased estimate of the *generalization error* and its variance.

Additional Notes

Proof of Lemma 0.1

Instead of considering the setup in the lemma we can equivalently assume that $\mathbb{E}[\Theta] = 0$ and that the Θ_n take values in $[a, b]$, where $a \leq 0 \leq b$. We will show that

$$\mathbb{P}\{\frac{1}{N} \sum_{n=1}^{N} \Theta_n \geq \varepsilon\} \leq e^{-2N\varepsilon^2/(b-a)^2}.$$

This, together with the equivalent bound

$$\mathbb{P}\{\frac{1}{N} \sum_{n=1}^{N} \Theta_n \leq -\varepsilon\} \leq e^{-2N\varepsilon^2}$$

will prove the claim. We have

$$\mathbb{P}\{\frac{1}{N} \sum_{n=1}^{N} \Theta_n \geq \varepsilon\} \overset{s \geq 0}{=} \mathbb{P}\{e^{s\frac{1}{N} \sum_{n=1}^{N} \Theta_n} \geq e^{s\varepsilon}\}$$

$$\overset{(a)}{\leq} \min_{s>0} \mathbb{E}[e^{s\frac{1}{N} \sum_{n=1}^{N} \Theta_n}] e^{-s\varepsilon}$$

$$\overset{(b)}{=} \min_{s>0} \prod_{n=1}^{N} \mathbb{E}[e^{\frac{s\Theta_n}{N}}] e^{-s\varepsilon}$$

$$= \min_{s>0} \mathbb{E}[e^{\frac{s\Theta}{N}}]^N e^{-s\varepsilon}$$

$$\overset{(c)}{\leq} \min_{s>0} e^{s^2(b-a)^2/(8N)} e^{-s\varepsilon}$$

$$\overset{s=4N\varepsilon/(b-a)^2}{=} e^{-2N\varepsilon^2/(b-a)^2}.$$

Here, step (a) follows from the Markov inequality. In step (b) we have used the fact that the random variables Θ_n, and

hence the random variables $e^{\frac{s\Theta_n}{N}}$, are independent so that the expectation of the product is equal to the product of the expectations. Finally, in step (c) we have used the so-called Hoeffding lemma. It states that for any random variable X, with $\mathbb{E}[X] = 0$ and $X \in [a, b]$ we have

$$\mathbb{E}[e^{sX}] \leq e^{\frac{1}{8}s^2(b-a)^2}.$$

To give a rough outline, consider the convex function e^{sx}, $s \geq 0$. In the range $[a, b]$ it is upper bounded by the chord (the line that is equal to the function at the two boundaries)

$$e^{sx} \leq \frac{x-a}{b-a}e^{sb} + \frac{b-x}{b-a}e^{sa}.$$

If we now take the expectation with respect to X and recall that $\mathbb{E}[X] = 0$ by assumption then we get

$$\mathbb{E}[e^{sX}] \leq \frac{b}{b-a}e^{sa} - \frac{a}{b-a}e^{sb} \leq e^{s^2(b-a)^2/8}.$$

The last step on the right requires several steps but this is now a pure calculus problem and we skip the details.