TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG
Die Ressourcenuniversität. Seit 1765.

Prof. Dr. Björn Sprungk
Faculty of Mathematics and Computer Science
Institute of Stochastics

Mathematics of machine learning

5. Training

Winter term 2024/25

5. Training

Content

5. Training

What's it about?

We will look at some aspects of optimization in machine learning:

1. How should you use the available data also for validation and testing?
2. How can empirical risk minimization be performed in practice using numerical optimization?

   $\Rightarrow$ Gradient descent

3. How is learning done in practice with large or even huge amounts of data?

   $\Rightarrow$ Stochastic gradient descent!

5.1 Training, validation, testing

- One should use the available data for different purposes:

  1. **Training:** Determine hypothesis h_s via numerical computation of a learning rule $h_s \approx A(s)$, e.g. ERM rule $A = \mathrm{ERM}_{\mathcal{H}}$

  2. **Validation or model selection:** Choose the most appropriate hypothesis class $\mathcal{H}$ or hyperparameter such as regularization parameter λ to train

  3. **Testing:** Estimate the generalization error of the learned hypothesis

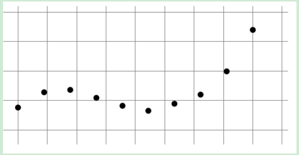
- To this end, we divide the data $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)$ **randomly** into a training sample s of m_S points, a validation sample v of m_V points, and a test set t of m_T points

- **Rule of thumb:** $m_V = m_T$ and $m_S \in [0.6m, 0.8m]$ – i.e. 60% to 80% percent of data for training and 20% and 10% for validation and testing.

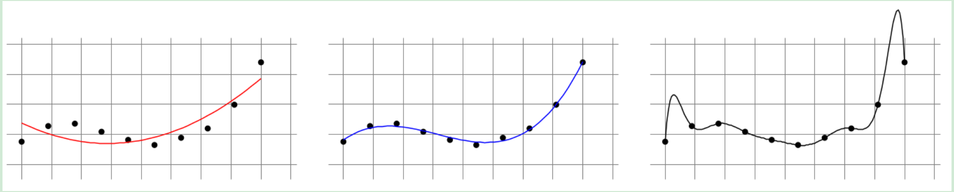
Validation

By validation we identify the best hyperparameters for the learning task, e.g., the regularization parameter λ or a complexity parameter p of the hypothesis class.

Example 5.1:



- Given $m = 10$ data pairs (x_i, y_i) we want to fit or learn a polynom $h_p(x) = \sum_{j=0}^{p} w_j x^p$
- Hypotheses $h_{s,p}$ determined by ERM for $p = 2$, $p = 3$ and $p = 10$:



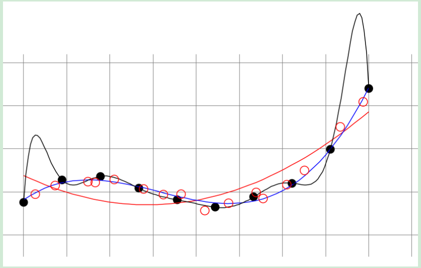
- Which of these leads to the smallest generalization error $\mathcal{R}_\mu(h_{s,p})$?

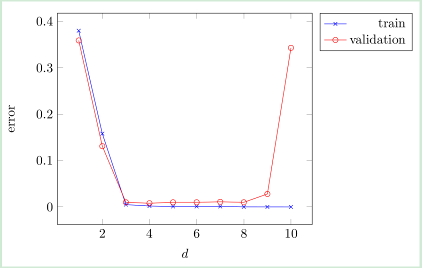
We compare different hypotheses $h_{s,1}, \dots, h_{s,J}$ learned by (regularized) ERM from J classes or models $\mathcal{H}_1, \dots, \mathcal{H}_J$ by their validation error based on the validation dataset $v = ((\mathbf{x}_1, y_1), \dots, (\mathbf{x}_{m_V}, y_{m_V}))$:

$$\mathcal{R}_v(h_{s,j}) = \frac{1}{m_V} \sum_{i=1}^{m_V} \ell(h_{s,j}(\mathbf{x}_i), y_i).$$

Example 5.1:

For the given example of polynomial fitting we obtain with $m_V = 16$ additional validation points:





Source: "Understanding Machine Learning" (2014)

$\Rightarrow$ We choose $h_{s,p}$ for $p = d = 3$ since there the validation error is smallest

Quality of the validation error

- The validation error $\mathcal{R}_v(h_s)$ shall estimate the expected risk $\mathcal{R}_\mu(h_s)$ analogously to $\mathcal{R}_s$
- Can we bound the difference $|\mathcal{R}_\mu(h_s) - \mathcal{R}_v(h_s)|$?
- We can apply Hoeffding's inequality and obtain for bounded loss $\ell \le c$:

$$\mathbb{P}_{\mu^{m_V}} \left(|\mathcal{R}_V(h_s) - \mathcal{R}_\mu(h_s)| > \epsilon\right) \le 2\exp(-2m_V\ \epsilon^2/c^2)$$

- For $\epsilon = \sqrt{\frac{c^2 \log(2/\delta)}{2m_V}}$ we get $\mathbb{P}\left(|\mathcal{R}_V(h_s) - \mathcal{R}_\mu(h_s)| \le \epsilon\right) \ge 1-\delta$
- By Bonferroni's correction we obtain for J hypotheses to be compared

$$\mathbb{P}_{\mu^{m_V}} \left(|\mathcal{R}_V(h_{s,j}) - \mathcal{R}_\mu(h_{s,j})| \le \sqrt{\frac{c^2 \log(2J/\delta)}{2m_V}} \quad \forall j = 1, \dots, J \right) \ge 1-\delta$$

Cross validation

1. We divide the m data points into K blocks s_k, $k = 1, \dots, K$, of $\frac{m}{K}$ points – assuming $\frac{m}{K} \in \mathbb{N}$

2. For each of blocks s_k compute for the J hyperparameters (e.g., for classes $\mathcal{H}_1, \dots, \mathcal{H}_J$ or regularization parameters $\lambda_1, \dots, \lambda_J$) the hypotheses

$$h_{k,j} = A_j((s_k^*)), \qquad s_k^* = (s_1, \dots, s_{k-1}, s_{k+1}, \dots, s_K),$$

where A_j denotes the learning algorithm with respect to the j-th hyperparameter, and calculate

$$\mathrm{err}_{k,j} := \mathcal{R}_{s_k}(h_{k,j})$$

3. The cross validation error for the j-th learning model or the j-th hyperparameter is then given by

$$\mathrm{err}_j^{\mathsf{CV}} := \frac{1}{K} \sum_{k=1}^{K} \mathrm{err}_{k,j} = \frac{1}{K} \sum_{k=1}^{K} \mathcal{R}_{s_k}(h_{k,j})$$

4. We then choose $j^* \in \operatorname{argmin}_j \mathrm{err}_j^{\mathsf{CV}}$ and compute $h_s = A_{j^*}(s)$

Notes

- Cross validation is particularly applied if there is not enough data to split it into a training and a validation set
- With the Hoeffding inequality one can control again for each j and k $|\mathcal{R}_\mu(h_{k,j}) - \mathcal{R}_{s_k}(h_{k,j})|$ but no longer $|\mathcal{R}_\mu(h_s) - \mathrm{err}_{j^*}^{\mathsf{CV}}|$ due to the dependencies of the errors $\mathrm{err}_{k,j}$.
- In practice, cross validation usually works very well, but in theory it can also lead to a wrong choice.
- The case $K = m$ is also called Leave-one-out (LOO) cross validation.

Test error

- Once one has decided for learned hypothesis h_s, a test dataset $t = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{m_T}, y_{m_T}))$ can be used to determine the test error

$$\mathcal{R}_T(h_s) = \frac{1}{m_T} \sum_{i=1}^{m_T} \ell(h_s(\mathbf{x}_i), y_i),$$

as an estimate of the generalization error $\mathcal{R}_\mu(h_s)$.

- Again, using the Hoeffding inequality for restricted loss functions we have

$$\mathbb{P}_{\mu^{m_T}}\left(|\mathcal{R}_T(h_s) - \mathcal{R}_\mu(h_s)| \leq \sqrt{\frac{c^2 \log(2/\delta)}{2m_T}} \right) \geq 1 - \delta.$$

- If a validation error has already been calculated for h_s based on a validation data set v, then this can also be used as an estimate.

Training via numerical optimization

- We will deal with numerical optimziation for computing the outcome of the (regularized) ERM rule

$$h_s = A(s) = \operatorname*{argmin}_{h \in \mathcal{H}} \mathcal{R}_s(h) + \lambda R(h)$$

  consisting of an empirical risk $\mathcal{R}_s(h)$ and, if necessary, a regularization $R(h)$ in the remainder of this chapter.

- In practice, the minimizer of $\mathcal{R}_s + \lambda R$ is calculated via numerical optimization methods.

- There are nowadays many such methods adapted to machine learning like AdaGrad or ADAM.

- We restrict ourselves here to the two classical and simplest ones, gradient decent and stochastic gradient descent.

Parametrized hypothesis classes

- We assume subsequently that the hypothesis classes under consideration $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ are parameterized.
- That is, there exists a parameter set $\mathcal{W} \subseteq \mathbb{R}^p$, $p \in \mathbb{N}$, and each hypothesis $h \in \mathcal{H}$ corresponds to a parameter(vector) $\mathbf{w} \in \mathcal{W}$: $h = h_{\mathbf{w}}$.
- The mapping $\mathbf{w} \mapsto h_{\mathbf{w}}$ does not need to be injective, e.g., for linear hypotheses $h_{(\mathbf{w},b)} \in \mathcal{L}_d$ we have $h_{\lambda(\mathbf{w},b)} \equiv h_{(\mathbf{w},b)}$ for all $\lambda > 0$.
- Furthermore, we assume now a loss function $\ell\colon \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \to [0, \infty)$ stated on the parameter set w. r. t which we have want to compute

$$\mathbf{w}_S \in \operatorname*{argmin}_{\mathbf{w} \in \mathcal{W}} \mathcal{R}_s(\mathbf{w}) + \lambda R(\mathbf{w}), \qquad \mathcal{R}_s(\mathbf{w}) := \frac{1}{m} \sum_{i=1}^{m} \ell(\mathbf{w}, \mathbf{x}_i, y_i)$$

which includes,e .g., all learning rules and approaches from Chapter 3