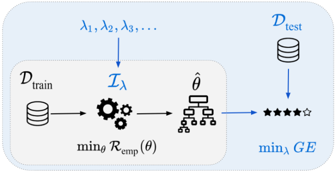
Introduction to Machine Learning

Hyperparameter Tuning - Problem Definition



Learning goals

- Understand the definition of *c* and know its properties
- Know the components of a tuning problem
- Be able to explain what makes tuning a complex problem

HPO AS BLACK-BOX OPTIMIZATION

Recall: **Hyperparameters (HP)** $\boldsymbol{\lambda}$ are parameters that are *inputs* to the training problem in which a learner $\mathcal{I}$ minimizes the empirical risk on a training data set in order to find optimal **model parameters** $\boldsymbol{\theta}$ which define the fitted model $\hat{f}$. Usually HPs influence the generalization performance in a non-trivial and subtle way.

Hyperparameter optimization (HPO) / **Tuning** is the process of finding good model hyperparameters $\boldsymbol{\lambda}$.

HPO AS BLACK-BOX OPTIMIZATION

- HPO algorithms automatically identify a well-performing hyperparameter configuration (HPC) $\boldsymbol{\lambda} \in \tilde{\boldsymbol{\Lambda}}$ for an learner $\mathcal{I}_{\boldsymbol{\lambda}}$.
- As input they take the search space $\tilde{\boldsymbol{\Lambda}} \subset \boldsymbol{\Lambda}$ which contains all considered HPs for optimization and their respective ranges:

$$\tilde{\boldsymbol{\Lambda}} = \tilde{\boldsymbol{\Lambda}}_1 \times \tilde{\boldsymbol{\Lambda}}_2 \times \dots \times \tilde{\boldsymbol{\Lambda}}_l$$

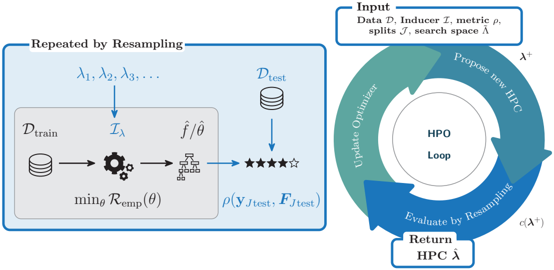
  where $\tilde{\boldsymbol{\Lambda}}_i$ is a bounded subset of the domain of the i-th HP $\boldsymbol{\Lambda}_i$, and can be either continuous, discrete, or categorical.

HPO AS BLACK-BOX OPTIMIZATION

The general HPO problem is defined as:

$$\boldsymbol{\lambda}^* \in \arg\min_{\boldsymbol{\lambda} \in \tilde{\boldsymbol{\Lambda}}} c(\boldsymbol{\lambda}) = \arg\min_{\boldsymbol{\lambda} \in \tilde{\boldsymbol{\Lambda}}} \widehat{\mathrm{GE}}(\mathcal{I}, \mathcal{J}, \rho, \boldsymbol{\lambda})$$

where $\boldsymbol{\lambda}^*$ denotes the theoretical optimum, and $c(\boldsymbol{\lambda})$ is a shorthand for the estimated generalization error when $\mathcal{I}$, resampling splits $\mathcal{J}$, performance measure ρ are fixed.



HPs and their respective evaluations successively are stored in the *archive* $\mathcal{A} = ((\boldsymbol{\lambda}^{(1)}, c(\boldsymbol{\lambda}^{(1)})), (\boldsymbol{\lambda}^{(2)}, c(\boldsymbol{\lambda}^{(2)})), \ldots)$, with $\mathcal{A}^{[t+1]} = \mathcal{A}^{[t]} \cup (\boldsymbol{\lambda}^+, c(\boldsymbol{\lambda}^+))$ if a single point is proposed by the tuner.

HPO AS BLACK-BOX OPTIMIZATION

- This means we estimate and optimize the generalization error

$$c(\boldsymbol{\lambda}) = \widehat{\mathrm{GE}}(\mathcal{I}, \mathcal{J}, \rho, \boldsymbol{\lambda}) = \mathrm{agr}\Bigg(\rho\Big(\mathbf{y}_{J_{\mathrm{test},1}}, \boldsymbol{F}_{J_{\mathrm{test},1}, \mathcal{I}(\mathcal{D}_{\mathrm{train},1}, \boldsymbol{\lambda})}\Big), \\ \vdots \\ \rho\Big(\mathbf{y}_{J_{\mathrm{test},1}}, \boldsymbol{F}_{J_{\mathrm{test},B}, \mathcal{I}(\mathcal{D}_{\mathrm{train},B}, \boldsymbol{\lambda})}\Big)\Bigg),$$

  of a learner $\mathcal{I}_{\boldsymbol{\lambda}}$, w.r.t. an HPC $\boldsymbol{\lambda}$
- $c(\boldsymbol{\lambda})$ is a black-box, as it usually has no closed-form mathematical representation, and hence no analytic gradient information is available.
- The evaluation of $c(\boldsymbol{\lambda})$ can take a significant amount of time.
- Therefore, the minimization of $c(\boldsymbol{\lambda})$ forms an *expensive black-box* optimization problem.

TUNING COMPONENTS

The components of a tuning problem are:

- The data set $\mathcal{D}$
- A learner $\mathcal{I}_{\boldsymbol{\lambda}}$ (possibly: several competing learners?) to be tuned
- The learner's hyperparameters and their respective regions-of-interest $\tilde{\boldsymbol{\Lambda}}$ over which we optimize
- The performance measure ρ, as determined by the application. Not necessarily identical to the loss function that defines the risk minimization problem for the learner!
- A (resampling) procedure for estimating the predictive performance which gives rise to the splits $\mathcal{J}$

We can thus define the HP tuner $\tau : (\mathcal{D}, \mathcal{I}, \tilde{\boldsymbol{\Lambda}}, \rho) \mapsto \hat{\boldsymbol{\lambda}}$ that proposes its estimate $\hat{\boldsymbol{\lambda}}$ of the true optimal configuration $\boldsymbol{\lambda}^*$ given $\mathcal{D}, \mathcal{I}_{\boldsymbol{\lambda}}$ with corresponding search space $\tilde{\boldsymbol{\Lambda}}$ to optimize, and a target measure ρ.

WHY IS TUNING SO HARD?

- Tuning is derivative-free ("black box problem"): It is usually impossible to compute derivatives of the objective (i.e., the resampled performance measure) that we optimize with regard to the HPs. All we can do is evaluate the performance for a given hyperparameter configuration.
- Every evaluation requires one or multiple train and predict steps of the learner. I.e., every evaluation is very **expensive**.
- Even worse: the answer we get from that evaluation is **not exact, but stochastic** in most settings, as we use resampling.
- Categorical and dependent hyperparameters aggravate our difficulties: the space of hyperparameters we optimize over has a non-metric, complicated structure.