The surprising effect of reshuffling the data during hyperparameter tuning

LMU Munich
Thomas Nagler <t.nagler@lmu.de>

Hyperparameter Optimization in ML

- **Goal**: Fit a model to data that predicts new observations well.

Hyperparameter Optimization in ML

- **Goal**: Fit a model to data that predicts new observations well.
- An **algorithm** g with hyperparameter $\boldsymbol{\lambda} \in \Lambda$ maps a data set $\mathcal{D}$ to a model $h = g_{\boldsymbol{\lambda}}(\mathcal{D}) \in \mathcal{H}$.
- Examples for **hyperparameter $\boldsymbol{\lambda}$**:
  - Penalty parameter in Lasso regression.
  - Depth and size of a tree ensemble.
  - Width and depth of neural network.

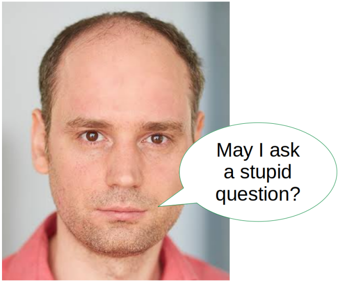
Hyperparameter Optimization in ML

- **Goal**: Fit a model to data that predicts new observations well.
- An **algorithm** g with hyperparameter $\boldsymbol{\lambda} \in \Lambda$ maps a data set $\mathcal{D}$ to a model $h = g_{\boldsymbol{\lambda}}(\mathcal{D}) \in \mathcal{H}$.
- Examples for **hyperparameter $\boldsymbol{\lambda}$**:
  - Penalty parameter in Lasso regression.
  - Depth and size of a tree ensemble.
  - Width and depth of neural network.
- **Hyperparameter optimization** aims to find a $\boldsymbol{\lambda}^*$ minimizing the expected generalization error:

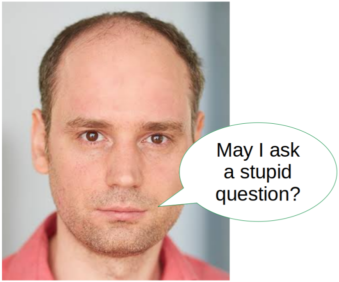
$$\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda} \in \Lambda} \mu(\boldsymbol{\lambda}), \quad \text{where} \quad \mu(\boldsymbol{\lambda}) = \mathbb{E}[\ell(\boldsymbol{Z}, g_{\boldsymbol{\lambda}}(\mathcal{D}))],$$

where $\ell(\boldsymbol{Z}, h)$ denotes the loss of model h on a new observation $\boldsymbol{Z}$.

The problem



The problem



- Standard approach:
  1. Split data $\mathcal{D}$ into training and validation set $\mathcal{D} = (\mathcal{T}, \mathcal{V})$.
  2. Train models $g_{\boldsymbol{\lambda}_1}(\mathcal{T}), \dots, g_{\boldsymbol{\lambda}_J}(\mathcal{T})$.
  3. Evaluate models on $\mathcal{V}$.
  4. Choose $\boldsymbol{\lambda}_j$ with best validation loss.
- *Why don't we reshuffle the data between evaluations of $\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_J$?*

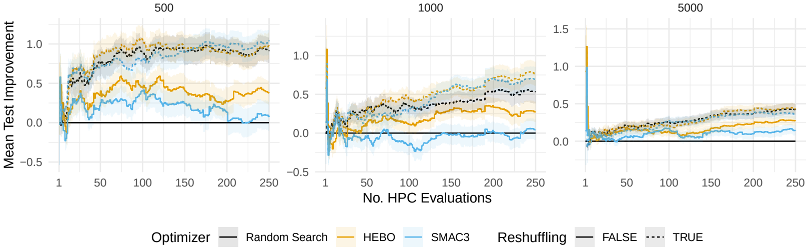
The problem

The problem

We ran some experiments…

Sounds like a terrible idea

The experiments



Agenda

Mathematical framework

- Dataset $\mathcal{D} = \{\boldsymbol{Z}_i\}_{i=1}^n$ of *i.i.d.* random variables from distribution P, where $\boldsymbol{Z}_i = (\boldsymbol{X}_i, Y_i)$ in the supervised setting.
- A finite set $\Lambda = \{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_J\} \subseteq \mathbb{R}^d$ of HPCs to be evaluated.

Mathematical framework

- Dataset $\mathcal{D} = \{\boldsymbol{Z}_i\}_{i=1}^n$ of *i.i.d.* random variables from distribution P, where $\boldsymbol{Z}_i = (\boldsymbol{X}_i, Y_i)$ in the supervised setting.
- A finite set $\Lambda = \{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_J\} \subseteq \mathbb{R}^d$ of HPCs to be evaluated.
- To estimate the generalization error, construct a resampling:
  - Draw M sets $\mathcal{I}_{1,j}, \ldots, \mathcal{I}_{M,j} \subset \{1, \ldots, n\}$ of validation indices with $n_{\mathrm{valid}} = \lceil \alpha n \rceil$ instances.
  - Validation and training sets $\mathcal{V}_{m,j} = \{\boldsymbol{Z}_i\}_{i \in \mathcal{I}_{m,j}}$, $\mathcal{T}_{m,j} = \{\boldsymbol{Z}_i\}_{i \notin \mathcal{I}_{m,j}}$.

Mathematical framework

- Dataset $\mathcal{D} = \{\boldsymbol{Z}_i\}_{i=1}^n$ of *i.i.d.* random variables from distribution P, where $\boldsymbol{Z}_i = (\boldsymbol{X}_i, Y_i)$ in the supervised setting.
- A finite set $\Lambda = \{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_J\} \subseteq \mathbb{R}^d$ of HPCs to be evaluated.
- To estimate the generalization error, construct a resampling:
  - Draw M sets $\mathcal{I}_{1,j}, \ldots, \mathcal{I}_{M,j} \subset \{1, \ldots, n\}$ of validation indices with $n_{\mathrm{valid}} = \lceil \alpha n \rceil$ instances.
  - Validation and training sets $\mathcal{V}_{m,j} = \{\boldsymbol{Z}_i\}_{i \in \mathcal{I}_{m,j}}$, $\mathcal{T}_{m,j} = \{\boldsymbol{Z}_i\}_{i \notin \mathcal{I}_{m,j}}$.
- Define M-fold validation loss

$$\widehat{\mu}(\boldsymbol{\lambda}_j) = \frac{1}{M} \sum_{m=1}^{M} L(\mathcal{V}_{m,j}, g_{\boldsymbol{\lambda}_j}(\mathcal{T}_{m,j})),$$

where $\quad L(\mathcal{V}_{m,j}, g_{\boldsymbol{\lambda}_j}(\mathcal{T}_{m,j})) = \frac{1}{n_{\mathrm{valid}}} \sum_{i \in \mathcal{I}_{m,j}} \ell(\boldsymbol{Z}_i, g_{\boldsymbol{\lambda}_j}(\mathcal{T}_{m,j})),$

Mathematical framework

- Recall we want to minimize $\mu(\boldsymbol{\lambda}) = \mathbb{E}[\ell(\boldsymbol{Z}, g_{\boldsymbol{\lambda}}(\mathcal{T}))]$.
- Since μ is unknown, take

$$\widehat{\boldsymbol{\lambda}} = \arg\min_{\boldsymbol{\lambda} \in \Lambda} \widehat{\mu}(\boldsymbol{\lambda}),$$

hoping $\mu(\widehat{\boldsymbol{\lambda}})$ will be small.

Mathematical framework

- Recall we want to minimize $\mu(\boldsymbol{\lambda}) = \mathbb{E}[\ell(\boldsymbol{Z}, g_{\boldsymbol{\lambda}}(\mathcal{T}))]$.
- Since μ is unknown, take

$$\widehat{\boldsymbol{\lambda}} = \arg\min_{\boldsymbol{\lambda} \in \Lambda} \widehat{\mu}(\boldsymbol{\lambda}),$$

  hoping $\mu(\widehat{\boldsymbol{\lambda}})$ will be small.
- Typically, same splits for each HPC: $\mathcal{I}_{m,j} = \mathcal{I}_m$ for all j and m.

Mathematical framework

- Recall we want to minimize $\mu(\boldsymbol{\lambda}) = \mathbb{E}[\ell(\boldsymbol{Z}, g_{\boldsymbol{\lambda}}(\mathcal{T}))]$.
- Since μ is unknown, take

$$\widehat{\boldsymbol{\lambda}} = \arg\min_{\boldsymbol{\lambda} \in \Lambda} \widehat{\mu}(\boldsymbol{\lambda}),$$

  hoping $\mu(\widehat{\boldsymbol{\lambda}})$ will be small.
- Typically, same splits for each HPC: $\mathcal{I}_{m,j} = \mathcal{I}_m$ for all j and m.
- We analyze the effect of reshuffling train-validation splits (i.e., $\mathcal{I}_{m,j} \neq \mathcal{I}_{m,j'}$ for $j \neq j'$).
  1. How does reshuffling affect the validation loss surface $\widehat{\mu}(\boldsymbol{\lambda})$?
  2. How does this affect the generalization error $\mu(\widehat{\boldsymbol{\lambda}})$?

Effect on the validation loss surface

Theorem

Under regularity conditions,

$$\sqrt{n}\,(\widehat{\mu}(\boldsymbol{\lambda}_j) - \mu(\boldsymbol{\lambda}_j))_{j=1}^{J} \to_d \mathcal{N}(0, \Sigma),$$

where

$$\Sigma_{i,j} = \tau_{i,j,M} K(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_j),$$

$$\tau_{i,j,M} = \lim_{n\to\infty} \frac{1}{nM^2\alpha^2} \sum_{s=1}^{n} \sum_{m=1}^{M} \sum_{m'=1}^{M} \Pr(s \in \mathcal{I}_{m,i} \cap \mathcal{I}_{m',j}),$$

and

$$K(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_j) = \lim_{n\to\infty} \mathrm{Cov}[\bar{\ell}_n(\boldsymbol{Z}', \boldsymbol{\lambda}_i), \bar{\ell}_n(\boldsymbol{Z}', \boldsymbol{\lambda}_j)],$$

$$\bar{\ell}_n(\boldsymbol{z}, \boldsymbol{\lambda}) = \mathbb{E}[\ell(\boldsymbol{z}, g_{\boldsymbol{\lambda}}(\mathcal{T}))] - \mathbb{E}[\ell(\boldsymbol{Z}, g_{\boldsymbol{\lambda}}(\mathcal{T}))].$$

Effect on the validation loss surface

$$\sqrt{n}\left(\widehat{\mu}(\boldsymbol{\lambda}_j) - \mu(\boldsymbol{\lambda}_j)\right)_{j=1}^{J} \to_d \mathcal{N}(0, \Sigma), \quad \Sigma_{i,j} = \tau_{i,j,M} K(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_j).$$

Most common examples

$$\tau_{i,j,M} = \begin{cases} \sigma^2, & i = j \\ \tau^2\sigma^2, & i \neq j. \end{cases},$$

Effect on the validation loss surface

$$\sqrt{n}\left(\widehat{\mu}(\boldsymbol{\lambda}_j) - \mu(\boldsymbol{\lambda}_j)\right)_{j=1}^{J} \to_d \mathcal{N}(0, \Sigma), \quad \Sigma_{i,j} = \tau_{i,j,M} K(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_j).$$

Most common examples

$$\tau_{i,j,M} = \begin{cases} \sigma^2, & i = j \\ \tau^2\sigma^2, & i \neq j. \end{cases},$$

Method	σ^2	τ^2
holdout (HO)	$1/\alpha$	1
reshuffled HO	$1/\alpha$	α
M-fold CV	1	1
reshuffled M-fold CV	1	1
M-fold HO	$1 + (1-\alpha)/M\alpha$	1
reshuffled M-fold HO	$1 + (1-\alpha)/M\alpha$	$1/(1 + (1-\alpha)/M\alpha)$

Effect on the validation loss surface

$$\sqrt{n}\,(\widehat{\mu}(\boldsymbol{\lambda}_j) - \mu(\boldsymbol{\lambda}_j))_{j=1}^{J} \to_d \mathcal{N}(0, \Sigma), \quad \Sigma_{i,j} = \tau_{i,j,M} K(\boldsymbol{\lambda}_i, \boldsymbol{\lambda}_j).$$

Takeaways

- Reshuffling ...
  - has no effect on the variance of individual validation losses,
  - decreases the correlation between validation losses of distinct $\boldsymbol{\lambda}$.
- Distant $\boldsymbol{\lambda} \neq \boldsymbol{\lambda}'$ are only weakly correlated anyway.
- No effect on M-fold CV (information in data used exhaustively)

Effect on generalization error

- To simplify the analysis, we work in the limit regime.
- Let $\epsilon(\boldsymbol{\lambda})$ be a zero-mean Gaussian process and

$$\widehat{\mu}(\boldsymbol{\lambda}_j) = \mu(\boldsymbol{\lambda}_j) + \epsilon(\boldsymbol{\lambda}_j), \quad \mathsf{Cov}(\epsilon(\boldsymbol{\lambda}), \epsilon(\boldsymbol{\lambda}')) = \begin{cases} K(\boldsymbol{\lambda}, \boldsymbol{\lambda}) & \text{if } \boldsymbol{\lambda} = \boldsymbol{\lambda}', \\ \tau^2 K(\boldsymbol{\lambda}, \boldsymbol{\lambda}') & \text{else,} \end{cases}$$

  with $0 < \underline{\sigma}^2 \leq \mathsf{Var}[\epsilon(\boldsymbol{\lambda})] \leq \sigma^2 < \infty$ for all $\boldsymbol{\lambda} \in \Lambda$.

Effect on generalization error

- To simplify the analysis, we work in the limit regime.
- Let $\epsilon(\boldsymbol{\lambda})$ be a zero-mean Gaussian process and

$$\widehat{\mu}(\boldsymbol{\lambda}_j) = \mu(\boldsymbol{\lambda}_j) + \epsilon(\boldsymbol{\lambda}_j), \quad \mathrm{Cov}(\epsilon(\boldsymbol{\lambda}), \epsilon(\boldsymbol{\lambda}')) = \begin{cases} K(\boldsymbol{\lambda}, \boldsymbol{\lambda}) & \text{if } \boldsymbol{\lambda} = \boldsymbol{\lambda}', \\ \tau^2 K(\boldsymbol{\lambda}, \boldsymbol{\lambda}') & \text{else,} \end{cases}$$

  with $0 < \underline{\sigma}^2 \leq \mathrm{Var}[\epsilon(\boldsymbol{\lambda})] \leq \sigma^2 < \infty$ for all $\boldsymbol{\lambda} \in \Lambda$.
- **Regret analysis**: compare
  - expected loss $\mu(\widehat{\boldsymbol{\lambda}})$ of empirically optimized $\widehat{\boldsymbol{\lambda}}$,
  - best achievable expected loss $\mu(\boldsymbol{\lambda}^*)$.

Effect on generalization error

Theorem

$$\mathbb{E}[\mu(\widehat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)] \le \sigma\sqrt{d}[8 + B(\tau) - A(\tau)].$$

Effect on generalization error

Theorem

$$\mathbb{E}[\mu(\widehat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)] \leq \sigma\sqrt{d}[8 + B(\tau) - A(\tau)].$$

- $B(\tau)$
  - quantifies how likely it is to pick a bad $\widehat{\boldsymbol{\lambda}}$ because of bad luck
  - more likely when ϵ is weakly correlated $\Rightarrow$ decreasing in τ.

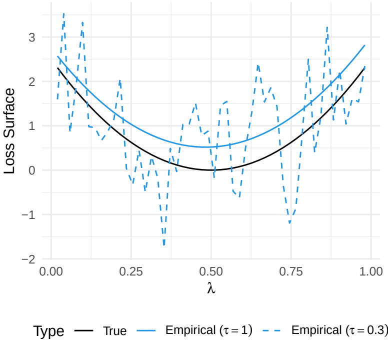
Effect on generalization error

Theorem

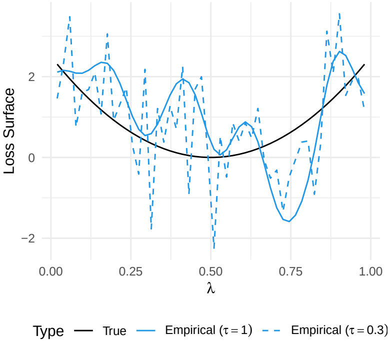
$$\mathbb{E}[\mu(\widehat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)] \leq \sigma\sqrt{d}[8 + B(\tau) - A(\tau)].$$

- $B(\tau)$
  - quantifies how likely it is to pick a bad $\widehat{\boldsymbol{\lambda}}$ because of bad luck
  - more likely when ϵ is weakly correlated $\Rightarrow$ decreasing in τ.
- $A(\tau)$
  - quantifies how likely it is to pick a good $\widehat{\boldsymbol{\lambda}}$ by luck
  - more likely when ϵ is weakly correlated $\Rightarrow$ decreasing in τ.

Effect on generalization error



(a) High signal-to-noise ratio



(b) Low signal-to-noise ratio

Proof idea

- We want to bound the probability that $\mu(\hat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)$ is large.
- Define the set of 'good' hyperparameters

$$\Lambda_\delta = \{\boldsymbol{\lambda}_j \colon \mu(\boldsymbol{\lambda}_j) - \mu(\boldsymbol{\lambda}^*) \leq \delta\}.$$

Proof idea

- We want to bound the probability that $\mu(\hat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)$ is large.
- Define the set of 'good' hyperparameters

$$\Lambda_\delta = \{\boldsymbol{\lambda}_j \colon \mu(\boldsymbol{\lambda}_j) - \mu(\boldsymbol{\lambda}^*) \leq \delta\}.$$

Proof idea

- We want to bound the probability that $\mu(\hat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)$ is large.
- Define the set of 'good' hyperparameters

$$\Lambda_\delta = \{\boldsymbol{\lambda}_j \colon \mu(\boldsymbol{\lambda}_j) - \mu(\boldsymbol{\lambda}^*) \leq \delta\}.$$

- It holds

$$\begin{aligned} &\Pr\left(\mu(\widehat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*) > \delta\right) \\ &= \Pr\left(\min_{\boldsymbol{\lambda} \notin \Lambda_\delta} \widehat{\mu}(\boldsymbol{\lambda}) < \min_{\boldsymbol{\lambda} \in \Lambda_\delta} \widehat{\mu}(\boldsymbol{\lambda})\right) \\ &\leq \Pr\left(\min_{\boldsymbol{\lambda} \notin \Lambda_\delta} \epsilon(\boldsymbol{\lambda}) < -\frac{\delta}{4}\right) + \Pr\left(\min_{\boldsymbol{\lambda} \in \Lambda_{\delta/2}} \epsilon(\boldsymbol{\lambda}) > \frac{\delta}{4}\right) \end{aligned}$$

Proof idea

- We want to bound the probability that $\mu(\hat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)$ is large.
- Define the set of 'good' hyperparameters

$$\Lambda_\delta = \{\boldsymbol{\lambda}_j \colon \mu(\boldsymbol{\lambda}_j) - \mu(\boldsymbol{\lambda}^*) \leq \delta\}.$$

- It holds

$$\begin{aligned}
&\Pr\left(\mu(\widehat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*) > \delta\right) \\
&= \Pr\left(\min_{\boldsymbol{\lambda} \notin \Lambda_\delta} \widehat{\mu}(\boldsymbol{\lambda}) < \min_{\boldsymbol{\lambda} \in \Lambda_\delta} \widehat{\mu}(\boldsymbol{\lambda})\right) \\
&\leq \Pr\left(\min_{\boldsymbol{\lambda} \notin \Lambda_\delta} \epsilon(\boldsymbol{\lambda}) < -\frac{\delta}{4}\right) + \Pr\left(\min_{\boldsymbol{\lambda} \in \Lambda_{\delta/2}} \epsilon(\boldsymbol{\lambda}) > \frac{\delta}{4}\right)
\end{aligned}$$

- Use Gaussian (anti-)concentration inequalities to bound two terms.

Effect on generalization error

Theorem

$$\mathbb{E}[\mu(\widehat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)] \leq \sigma\sqrt{d}[8 + B(\tau) - A(\tau)].$$

where

$$B(\tau) = 48\left[\sqrt{1-\tau^2}\sqrt{\log J} + \tau\sqrt{1+\log(3\kappa)_+}\right],$$

$$A(\tau) = \sqrt{1-\tau^2}(\underline{\sigma}/\sigma)\sqrt{\log\left(\frac{\sigma}{2m\eta^2}\right)_+}.$$

Effect on generalization error

Theorem

$$\mathbb{E}[\mu(\widehat{\boldsymbol{\lambda}}) - \mu(\boldsymbol{\lambda}^*)] \leq \sigma\sqrt{d}[8 + B(\tau) - A(\tau)].$$

where

$$B(\tau) = 48\left[\sqrt{1-\tau^2}\sqrt{\log J} + \tau\sqrt{1 + \log(3\kappa)_+}\right],$$

$$A(\tau) = \sqrt{1-\tau^2}(\underline{\sigma}/\sigma)\sqrt{\log\left(\frac{\sigma}{2m\eta^2}\right)_+}.$$

- Correlation constant $\kappa = \sup_{\|\boldsymbol{\lambda}\|,\|\boldsymbol{\lambda}'\| \leq 1} \frac{|K(\boldsymbol{\lambda},\boldsymbol{\lambda}) - K(\boldsymbol{\lambda},\boldsymbol{\lambda}')|}{K(\boldsymbol{\lambda},\boldsymbol{\lambda})\|\boldsymbol{\lambda}-\boldsymbol{\lambda}'\|^2}$.
- Grid density η: minimal number s.t. any η-ball in $\{\|\boldsymbol{\lambda}\| \leq 1\}$ contains at least one element of Λ.
- Curvature around the minimum: $m = \sup_{\boldsymbol{\lambda}\in\Lambda} \frac{|\mu(\boldsymbol{\lambda}) - \mu(\boldsymbol{\lambda}^*)|}{\|\boldsymbol{\lambda}-\boldsymbol{\lambda}^*\|^2}$.

Effect on generalization error

$$\text{Signal-to-noise ratio } \rho = \frac{m\eta^2}{\sigma}$$

Two regimes

- $\rho \geq 2e$:

- $\rho < 2e$:

Effect on generalization error

$$\text{Signal-to-noise ratio } \rho = \frac{m\eta^2}{\sigma}$$

Two regimes

- $\rho \geq 2e$:

- $\rho < 2e$:

Effect on generalization error

$$\text{Signal-to-noise ratio } \rho = \frac{m\eta^2}{\sigma}$$

Two regimes

- $\rho \geq 2e$:
  - $A(\tau) = 0$ and reshuffling cannot lead to an improvement of the bound.
  - Signal is much stronger than the noise, the HPO problem is so easy that reshuffling will not help.
- $\rho < 2e$:

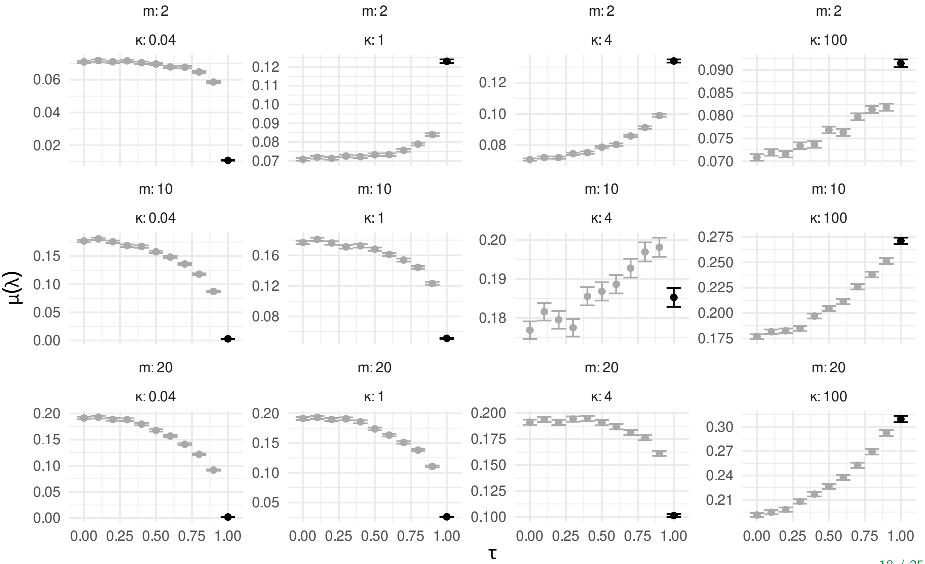
Effect on generalization error

$$\text{Signal-to-noise ratio } \rho = \frac{m\eta^2}{\sigma}$$

Two regimes

- $\rho \geq 2e$:
  - $A(\tau) = 0$ and reshuffling cannot lead to an improvement of the bound.
  - Signal is much stronger than the noise, the HPO problem is so easy that reshuffling will not help.
- $\rho < 2e$:
  - $A(\tau)$ and $B(\tau)$ enter with opposing signs.
  - If ϵ weakly correlated, gains in $A(\tau)$ may outweigh loss in $B(\tau)$.

Effect on generalization error: simulations



Agenda

Setup

- **Datasets**: 10 datasets for binary classification from OpenML (<100 features, 10,000–1,000,000 observations).
  - Subsampled sets of $500, 1000, 5000$ points.
- **ML algorithms**: CatBoost, XGBoost, Elastic Net, neural network.
- **HPO strategies**:
  - Random search ($J = 500$)
  - Bayesian optimization with HEBO or SMAC3 ($J = 250$).
- **Test performance** evaluated by re-training on combined train-validation sets, then testing out-of-sample (10 replications)

Benchmarking results: random search

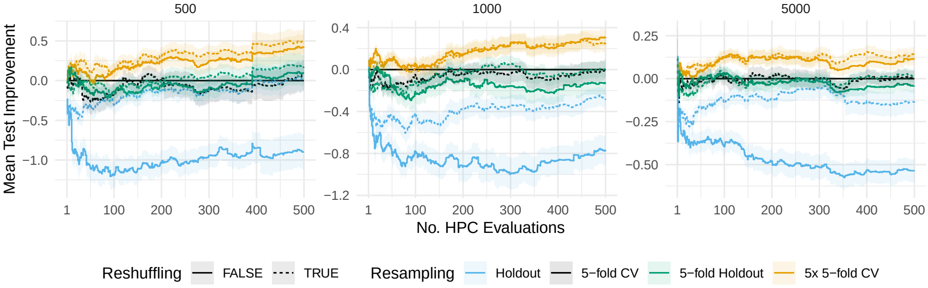


Figure: Average improvement (compared to standard 5-fold CV) with respect to test performance (ROC AUC) for increasing n.

Benchmarking results: Bayesian optimization

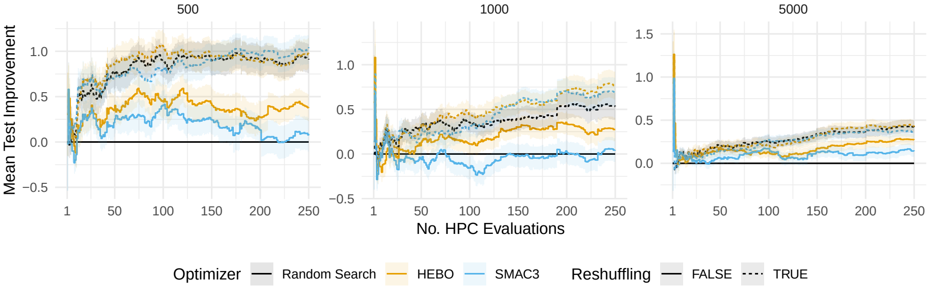


Figure: Average improvement (compared to random search/holdout) with respect to test performance (ROC AUC) for different n.

Agenda

Conclusion

- Reshuffling the data during hyperparameter optimization can improve the generalization error.
- The effect depends on the signal-to-noise ratio and the correlation of losses.
- Effects are especially large if validation sets are small (hold-out).
- Next step: Design algorithms exploiting this phenomenon.

Conclusion

- Reshuffling the data during hyperparameter optimization can improve the generalization error.
- The effect depends on the signal-to-noise ratio and the correlation of losses.
- Effects are especially large if validation sets are small (hold-out).
- Next step: Design algorithms exploiting this phenomenon.

Nagler, Schneider, Bischl, Feurer. *Reshuffling resampling splits can improve generalization of hyperparameter optimization.* (NeurIPS '24)

Common resampling strategies

1. **(holdout)** Let $M = 1$ and $\mathcal{I}_{1,j} = \mathcal{I}_1$ for all $j = 1, \ldots, J$, and some size-$\lceil \alpha n \rceil$ index set $\mathcal{I}_1$.

2. **(reshuffled holdout)** Let $M = 1$ and $\mathcal{I}_{1,1}, \ldots, \mathcal{I}_{1,J}$ be independently drawn from the uniform distribution over all size-$\lceil \alpha n \rceil$ subsets from $\{1, \ldots, n\}$.

3. **(M-fold CV)** Let $\alpha = 1/M$ and $\mathcal{I}_1, \ldots, \mathcal{I}_M$ be a disjoint partition of $\{1, \ldots, n\}$, and $\mathcal{I}_{m,j} = \mathcal{I}_m$ for all $j = 1, \ldots, J$.

4. **(reshuffled M-fold CV)** Let $\alpha = 1/M$ and $(\mathcal{I}_{1,j}, \ldots, \mathcal{I}_{M,j}), j = 1, \ldots, J$, be independently drawn from the uniform distribution over disjoint partitions of $\{1, \ldots, n\}$.

5. **(M-fold holdout)** Let $\mathcal{I}_m, m = 1, \ldots, M$, be independently drawn from the uniform distribution over size-$\lceil \alpha n \rceil$ subsets of $\{1, \ldots, n\}$ and set $\mathcal{I}_{m,j} = \mathcal{I}_m$ for all $m = 1, \ldots, M, j = 1, \ldots, J$.

6. **(reshuffled M-fold holdout)** Let $\mathcal{I}_{m,j}, m = 1, \ldots, M, j = 1, \ldots, J$, be independently drawn from the uniform distribution over size-$\lceil \alpha n \rceil$ subsets of $\{1, \ldots, n\}$.