LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Introduction to Machine Learning

Hyperparameter Tuning

Bernd Bischl, Christoph Molnar, Daniel Schalk, Fabian Scheipl

Department of Statistics – LMU Munich

HYPERPARAMETER TUNING

- Many parameters or decisions for an ML algorithm are not decided by the fitting procedure
- **Model parameters** are optimized during training, by some form of loss minimization. They are an **output** of the training. E.g., the coefficients of a linear model or the optimal splits of a tree learner.
- **Hyperparameters** must be specified before the training phase. They are an **input** of the training. E.g., how small a leaf can become for a tree; k and which distance measure to use for kNN
- HPs have to be set either by the user or by (smart) default values
- Our goal is to optimize these w.r.t. the estimated prediction error; this implies an independent test set, or cross-validation
- The same applies to preprocessing, feature construction and other model-relevant operations. In general we might be interested in optimizing an entire ML "pipeline"

WHY TUNING IS IMPORTANT

- Hyperparameters control the complexity of a model, i.e., how flexible the model is
- If a model is too flexible so that it simply “memorizes” the training data, we will face the dreaded problem of overfitting
- Hence, control of capacity, i.e., proper setting of hyperparameters can prevent overfitting the model on the training set
- Many other factors like optimization control settings, distance functions, scaling, algorithmic variants in the fitting procedure can heavily influence model performance in non-trivial ways. It is extremely hard to guess the correct choices here.

TYPES OF HYPERPARAMETERS

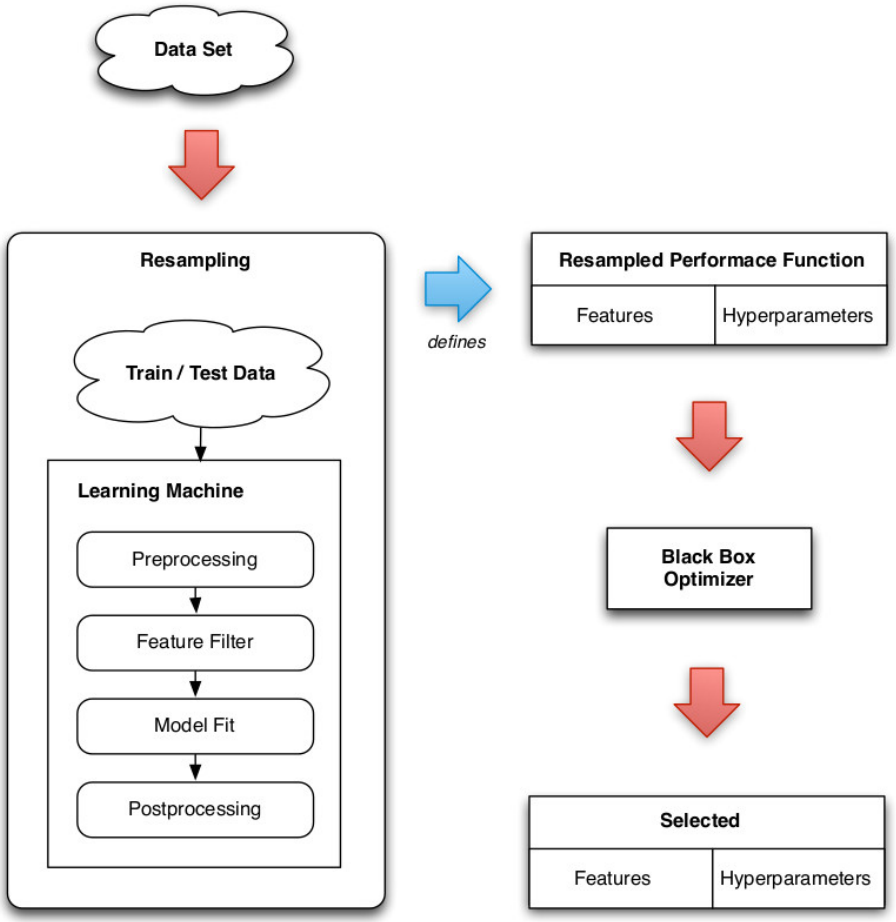
- Numerical parameters (real valued / integers)
  - *mtry* in a random forest
  - Neighborhood size k for kNN
- Categorical parameters:
  - Which split criterion for classification trees?
  - Which distance measure for kNN?
- Ordinal parameters:
  - $\{\texttt{low}, \texttt{medium}, \texttt{high}\}$
- Dependent parameters:
  - If we use the Gaussian kernel for the SVM, what is its width?

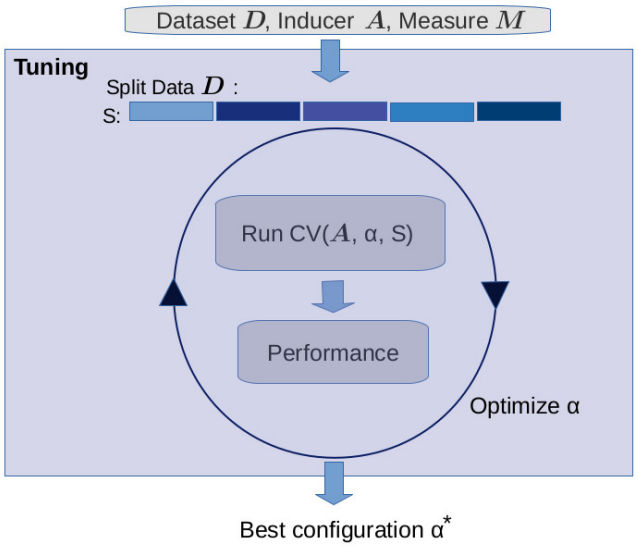
COMPONENTS OF TUNING PROBLEM

- The learner (possibly: several competing learners?)
- The performance measure. Determined by the application. Not necessarily identical to the loss function that the learner tries to minimize. We could even be interested in multiple measures simultaneously, e.g., accuracy and sparseness of our model, TPR and PPV, etc.
- A (resampling) procedure for estimating the predictive performance
- The learner's hyperparameters and their respective regions-of-interest over which we optimize

HYPERPARAMETER TUNING

Tuner proposes configuration, eval by resampling, tuner receives performance, iterate



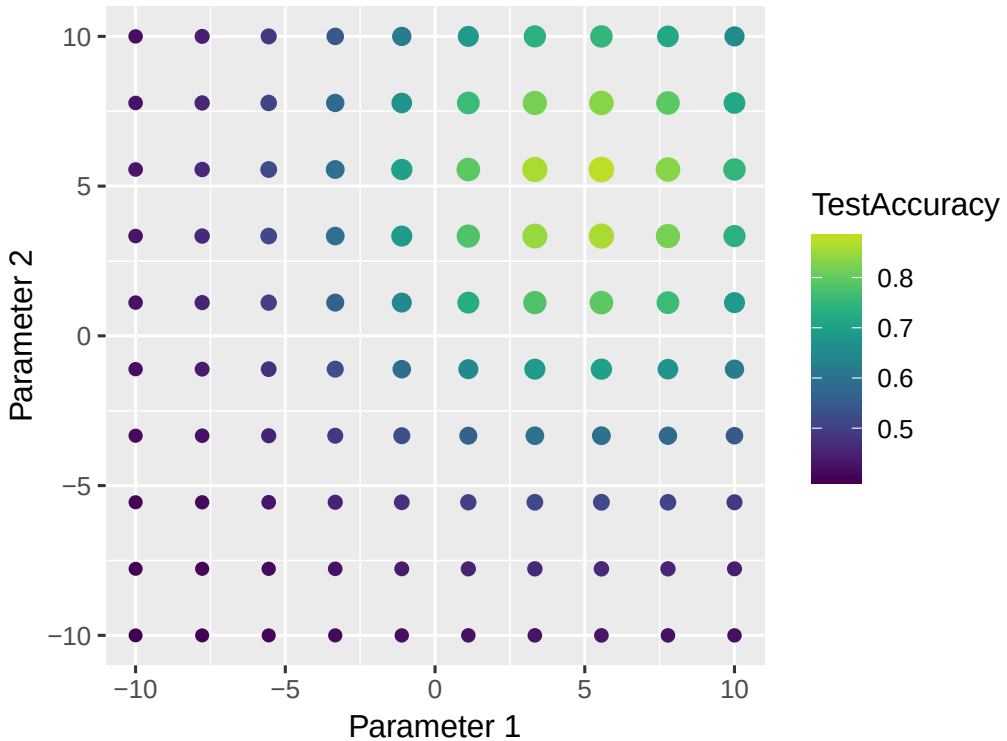


WHY IS TUNING SO HARD?

- 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.
- For large and difficult problems parallelizing the computation seems relevant, to evaluate multiple HP configurations in parallel or to speed up the resampling-based performance evaluation

GRID SEARCH

- Simple technique which is still quite popular, tries all HP combinations on a multi-dimensional discretized grid
- For each hyperparameter a finite set of candidates is predefined
- Then, we simply search all possible combinations in arbitrary order



Grid search over 10x10 points

GRID SEARCH

Advantages

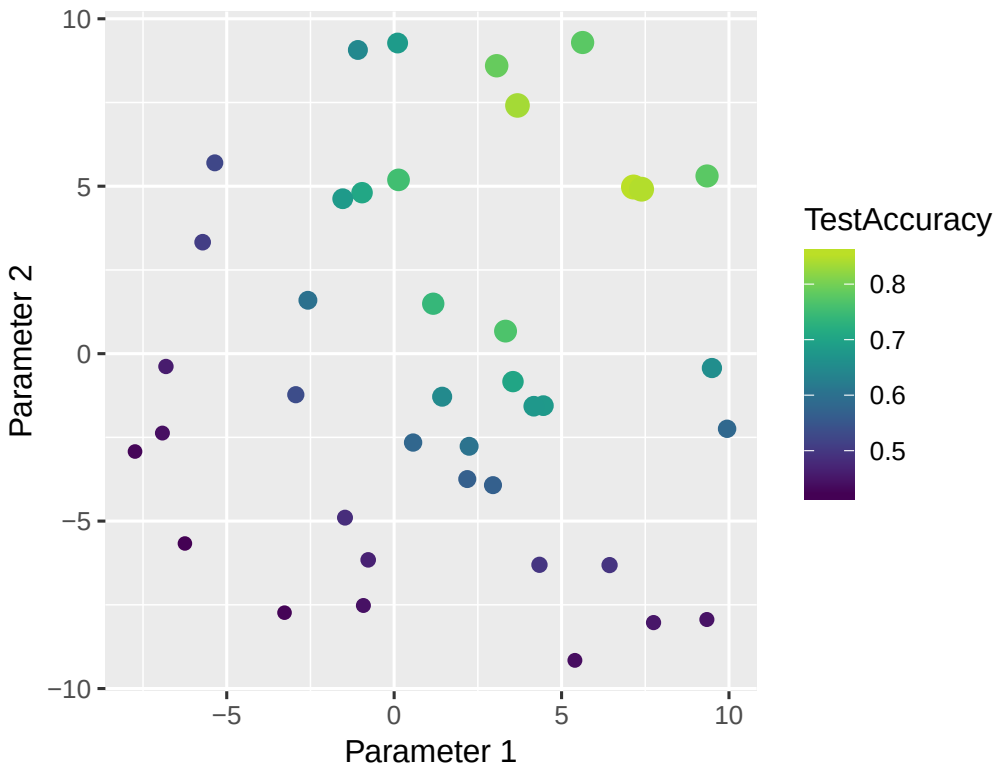
- Very easy to implement, therefore very popular
- All parameter types possible
- Parallelization is trivial

Disadvantages

- Combinatorial explosion, inefficient
- Searches large irrelevant areas
- Which values / discretization?

RANDOM SEARCH

- Small variation of grid search
- Uniformly sample from the region-of-interest



Random search over 100 points

RANDOM SEARCH

Advantages

- Very easy to implement, therefore very popular
- All parameter types possible
- Parallelization is trivial
- Anytime algorithm - we can always increase the budget when we are not satisfied
- Often better than grid search, as each individual parameter has been tried with m different values, when the search budget was m. Mitigates the problem of discretization

Disadvantages

- As for grid search, many evaluations in areas with low likelihood for improvement

ADVANCED TUNING TECHNIQUES

- Stochastic local search, e.g. simulated annealing
- Genetic algorithms / CMAES
- Iterated F-Racing
- Model-based Optimization / Bayesian Optimization
- Hyperband
- . . .