Hyperparameter Tuning with mlr3tuning::CHEAT SHEET

Class Overview

The package provides a set of R6 classes which allow to (a) define general hyperparameter (HP) tuning instances and (b) run algorithms which optimize on these. (a) is called a TuningInstanceSingleCrit or TuningInstanceMultiCrit, which defines a blackbox optimization function that maps HP candidate configurations to resampled performance values for arbitrary performance measures.

Tuners - When to stop

Construction: tune(.key, ...) • `evals(n_evals)` After a given amount of iterations. • `clock_time(secs, stop_time)` After a given absolute time. • `model_time(secs)` After a given training time. • `perf_reached(level)` After a specific performance was reached. • `stagnation(its, threshold)` After the performance stagnated for given iterations.

Transformations for Rescaling


Set: `store_benchmark_result = TRUE` to store resamplings of evaluations and `store_models = TRUE` to store associated models.

Example

```
# optimize hyperpar of RBF SVM on logscale
learner = classif.rbf, tuned = c("gamma", type = "<gamma>function")

tuner = TuneControl$new(parallel = TRUE, batch_size = 2)

tune_result = tune(learner = learner, task, resampling, measure, evals20, tuner, tune_ps)
```

Nesting Resampling

Resampling the AutoTuner results in nested resampling with an inner and outer loop.

Example

```
rr = resampling::rr(task)

tuner = AutoTuner$new(learner, resamplingInner, resamplingOuter, measure, evals20, tuner, tune_ps)

tuning_result = tuner$optimize(task)
```

Logging and Parallelization

Sets the parallelization backend. Speeds up tuning by running iterations in parallel.

AutoTuner - Tune before Train

Wraps learner and performs integrated tuning.

Example

```
at = AutoTuner$new(learner, resampling, measure, terminator, tuner, tune_ps)
```

Parameter Dependencies

Dependencies prevent invalid learner configurations.

Example

```
tune_ps$add_dep(id, on, cond)
```

Use TuningInstanceMultiCrit for multi-criteria tuning.