

# Package ‘mlr3tuningspaces’

January 3, 2022

**Title** Search Spaces for Hyperparameter Tuning

**Version** 0.1.0

**Description** Collection of search spaces for hyperparameter tuning.  
Includes various search spaces that can be directly applied on an  
‘mlr3’ learner. Additionally, meta information about the search space  
can be queried.

**License** LGPL-3

**URL** <https://mlr3tuningspaces.mlr-org.com>,  
<https://github.com/mlr-org/mlr3tuningspaces>

**BugReports** <https://github.com/mlr-org/mlr3tuningspaces/issues>

**Depends** mlr3tuning (>= 0.8.0), R (>= 3.1.0)

**Imports** checkmate (>= 2.0.0), data.table (>= 1.14.0), mlr3 (>= 0.11.0), mlr3misc (>= 0.9.1), paradox (>= 0.7.1), R6 (>= 2.5.0)

**Suggests** e1071 (>= 1.7-6), glmnet (>= 4.1-2), kknn (>= 1.3.1),  
mlr3learners (>= 0.4.5), ranger (>= 0.12.1), rpart (>= 4.1-15),  
testthat (>= 3.0.0), xgboost (>= 1.4.1.1)

**Config/testthat/edition** 3

**Encoding** UTF-8

**RoxygenNote** 7.1.2

**Collate** 'mlr\_tuning\_spaces.R' 'TuningSpace.R' 'bibentries.R' 'sugar.R'  
'tuning\_spaces\_default.R' 'tuning\_spaces\_rbv2.R' 'zzz.R'

**NeedsCompilation** no

**Author** Marc Becker [cre, aut] (<<https://orcid.org/0000-0002-8115-0400>>),  
Michel Lang [ctb] (<<https://orcid.org/0000-0001-9754-0393>>)

**Maintainer** Marc Becker <marcbecker@posteo.de>

**Repository** CRAN

**Date/Publication** 2022-01-03 17:30:02 UTC

**R topics documented:**

mlr3tuningspaces-package . . . . .	2
lts . . . . .	3
mlr_tuning_spaces . . . . .	3
TuningSpace . . . . .	4
tuning_spaces_default . . . . .	6
tuning_spaces_rbv2 . . . . .	7
<b>Index</b>	<b>9</b>

---

mlr3tuningspaces-package

*mlr3tuningspaces: Search Spaces for Hyperparameter Tuning*


---

**Description**

Collection of search spaces for hyperparameter tuning. Includes various search spaces that can be directly applied on an ‘mlr3’ learner. Additionally, meta information about the search space can be queried.

**Author(s)**

**Maintainer:** Marc Becker <marcbecker@posteo.de> ([ORCID](#))

Other contributors:

- Michel Lang <michellang@gmail.com> ([ORCID](#)) [contributor]

**See Also**

Useful links:

- <https://mlr3tuningspaces.mlr-org.com>
- <https://github.com/mlr-org/mlr3tuningspaces>
- Report bugs at <https://github.com/mlr-org/mlr3tuningspaces/issues>

---

lts	<i>Syntactic Sugar for Tuning Space Construction</i>
-----	--

---

### Description

This function complements [mlr\\_tuning\\_spaces](#) with functions in the spirit of [mlr3::mlr\\_sugar](#).

### Usage

```
lts(x)

## S3 method for class 'character'
lts(x)

## S3 method for class 'Learner'
lts(x)

ltss(x)
```

### Arguments

x	(character()   <a href="#">mlr3::Learner</a> ) If character, key passed the dictionary to retrieve the tuning space. If <a href="#">mlr3::Learner</a> , default tuning space is added to the learner.
---	--

### Value

- [TuningSpace](#) for `lts()`
- list of [TuningSpace](#) for `ltss()`

### Examples

```
lts("classif.ranger.default")
```

---

mlr_tuning_spaces	<i>Dictionary of Tuning Spaces</i>
-------------------	------------------------------------

---

### Description

A simple [mlr3misc::Dictionary](#) storing objects of class [TuningSpace](#). Each tuning space has an associated help page, see `mlr_tuning_spaces_[id]`.

### Format

[R6::R6Class](#) object inheriting from [mlr3misc::Dictionary](#).

**Methods**

See [mlr3misc::Dictionary](#).

**S3 methods**

- `as.data.table(dict)`  
[mlr3misc::Dictionary](#) -> `data.table::data.table()`  
Returns a `data.table::data.table()` with columns "key", "learner" and "n\_values".

**Examples**

```
library(data.table)
as.data.table(mlr_tuning_spaces)
mlr_tuning_spaces$get("classif.ranger.default")
```

---

TuningSpace

*Tuning Spaces*


---

**Description**

This is the abstract base class for tuning spaces which define a search space for hyperparameter tuning.

TuningSpace objects store a list of [paradox::TuneToken](#) and additional meta information. These tokens can be assigned to the \$values slot of a learner's [paradox::ParamSet](#).

**Public fields**

id (character(1)).  
values (list()).  
tags (character()).  
package (character()).  
learner (character(1)).

**Methods****Public methods:**

- [TuningSpace\\$new\(\)](#)
- [TuningSpace\\$get\\_learner\(\)](#)
- [TuningSpace\\$format\(\)](#)
- [TuningSpace\\$print\(\)](#)
- [TuningSpace\\$clone\(\)](#)

**Method** `new()`: Creates a new instance of this R6 class.

*Usage:*

```
TuningSpace$new(id, values, tags, learner, package = character())
```

*Arguments:*

id (character(1))

Identifier for the new instance.

values (list())

List of [paradox::TuneToken](#) and parameter values.

tags (character())

Tags to group and filter tuning spaces.

learner (character(1))

[mlr3::Learner](#) identifier in [mlr3::mlr\\_learners](#).

package (character())

Packages which provide the [Learner](#), e.g. [mlr3learners](#) for the learner [mlr3learners::LearnerClassifRanger](#) which interfaces the [ranger](#) package.**Method** `get_learner()`: Returns a learner with [TuneToken](#) set in parameter set.*Usage:*

TuningSpace\$get\_learner(...)

*Arguments:*

... (named 'list()')

Passed to `mlr3::lrn()`. Named arguments passed to the constructor, to be set as parameters in the [paradox::ParamSet](#), or to be set as public field. See `mlr3misc::dictionary_sugar_get()` for more details.*Returns:* [mlr3::Learner](#)**Method** `format()`:*Usage:*

TuningSpace\$format(...)

**Method** `print()`:*Usage:*

TuningSpace\$print(...)

**Method** `clone()`: The objects of this class are cloneable with this method.*Usage:*

TuningSpace\$clone(deep = FALSE)

*Arguments:*

deep Whether to make a deep clone.

**Examples**

```
library(mlr3tuning)

# get default tuning space of rpart learner
tuning_space = mlr_tuning_spaces$get("classif.rpart.default")

# get learner and set tuning space
learner = lrn("classif.rpart")
```

```
learner$param_set$values = tuning_space$values

# tune learner
instance = tune(
  method = "random_search",
  task = tsk("pima"),
  learner = learner,
  resampling = rsmpl("holdout"),
  measure = msr("classif.ce"),
  term_evals = 10)

instance$result
```

---

tuning\_spaces\_default *Default Tuning Spaces*

---

## Description

Tuning spaces from the Bischl (2021) article.

### ranger tuning space

- replace [TRUE,FALSE]
- sample.fraction [0.1, 1]
- num.trees [1, 2000]
- mtry.ratio [0, 1]

### rpart tuning space

- minsplit [2, 128]
- minbucket [1, 64]
- cp [ $1e - 04$ , 0.1]

### svm tuning space

- cost [ $1e - 04$ , 10000]
- kernel ["polynomial", "radial", "sigmoid", "linear"]
- degree [2, 5]
- gamma [ $1e - 04$ , 10000]

**xgboost tuning space**

- eta [ $1e - 04$ , 1]
- nrounds [1, 5000]
- max\_depth [1, 20]
- colsample\_bytree [0.1, 1]
- colsample\_bylevel [0.1, 1]
- lambda [0.1, 1]
- gamma [ $1e - 04$ , 1000]
- alpha [ $1e - 04$ , 1000]
- subsample [0.1, 1]

**Source**

Bischl B, Binder M, Lang M, Pielok T, Richter J, Coors S, Thomas J, Ullmann T, Becker M, Boulesteix A, Deng D, Lindauer M (2021). “Hyperparameter Optimization: Foundations, Algorithms, Best Practices and Open Challenges.” 2107.05847, <https://arxiv.org/abs/2107.05847>.

---

tuning\_spaces\_rbv2      *RandomBot Tuning Spaces*

---

**Description**

Tuning spaces from the Kuehn (2018) article.

**glmnet tuning space**

- alpha [0, 1]
- s [ $1e - 04$ , 1000]

**kknn tuning space**

- k [1, 30]

**ranger tuning space**

- num.trees [1, 2000]
- replace [TRUE,FALSE]
- sample.fraction [0.1, 1]
- mtry.ratio [0, 1]
- respect.unordered.factors [“ignore”, “order”, “partition”]
- min.node.size [1, 100]
- splitrule [“gini”, “extratrees”]
- num.random.splits [1, 100]

mtry.power is replaced by mtry.ratio.

**rpart tuning space**

- cp [ $1e - 04$ , 1]
- maxdepth [1, 30]
- minbucket [1, 100]
- minsplit [1, 100]

**svm tuning space**

- kernel [“linear”, “polynomial”, “radial”]
- cost [ $1e - 04$ , 1000]
- gamma [ $1e - 04$ , 1000]
- tolerance [ $1e - 04$ , 2]
- degree [2, 5]

**xgboost tuning space**

- booster [“gblinear”, “gbtree”, “dart”]
- nrounds [2, 8]
- eta [ $1e - 04$ , 1]
- gamma [ $1e - 05$ , 7]
- lambda [ $1e - 04$ , 1000]
- alpha [ $1e - 04$ , 1000]
- subsample [0.1, 1]
- max\_depth [1, 15]
- min\_child\_weight [1, 100]
- colsample\_bytree [0.01, 1]
- colsample\_bylevel [0.01, 1]
- rate\_drop [0, 1]
- skip\_drop [0, 1]

**Source**

Kuehn D, Probst P, Thomas J, Bischl B (2018). “Automatic Exploration of Machine Learning Experiments on OpenML.” 1806.10961, <https://arxiv.org/abs/1806.10961>.



# Index

- \* **Dictionary**
  - mlr\_tuning\_spaces, 3
- \* **TuningSpace**
  - mlr\_tuning\_spaces, 3
- \* **datasets**
  - mlr\_tuning\_spaces, 3
- classif.ranger.default
  - (tuning\_spaces\_default), 6
- classif.ranger.rbv2
  - (tuning\_spaces\_rbv2), 7
- classif.rpart.default
  - (tuning\_spaces\_default), 6
- classif.rpart.rbv2
  - (tuning\_spaces\_rbv2), 7
- classif.svm.default
  - (tuning\_spaces\_default), 6
- classif.svm.rbv2 (tuning\_spaces\_rbv2), 7
- classif.xgboost.default
  - (tuning\_spaces\_default), 6
- classif.xgboost.rbv2
  - (tuning\_spaces\_rbv2), 7
- data.table::data.table(), 4
- Learner, 5
- lts, 3
- ltss (lts), 3
- mlr3::Learner, 3, 5
- mlr3::mlr\_learners, 5
- mlr3::mlr\_sugar, 3
- mlr3learners::LearnerClassifRanger, 5
- mlr3misc::Dictionary, 3, 4
- mlr3tuningspaces
  - (mlr3tuningspaces-package), 2
- mlr3tuningspaces-package, 2
- mlr\_tuning\_spaces, 3, 3
- paradox::ParamSet, 4, 5
- paradox::TuneToken, 4, 5
- R6, 4
- R6::R6Class, 3
- regr.ranger.default
  - (tuning\_spaces\_default), 6
- regr.ranger.rbv2 (tuning\_spaces\_rbv2), 7
- regr.rpart.default
  - (tuning\_spaces\_default), 6
- regr.rpart.rbv2 (tuning\_spaces\_rbv2), 7
- regr.svm.default
  - (tuning\_spaces\_default), 6
- regr.svm.rbv2 (tuning\_spaces\_rbv2), 7
- regr.xgboost.default
  - (tuning\_spaces\_default), 6
- regr.xgboost.rbv2 (tuning\_spaces\_rbv2), 7
- TuneToken, 5
- tuning\_spaces\_default, 6
- tuning\_spaces\_rbv2, 7
- TuningSpace, 3, 4