Package 'OpenML'

March 2, 2018

Title Open Machine Learning and Open Data Platform

Description We provide an R interface to 'OpenML.org' which is an online machine learning platform where researchers can access open data, download and upload data sets, share their machine learning tasks and experiments and organize them online to work and collaborate with other researchers.
The R interface allows to query for data sets with specific properties, and allows the downloading and uploading of data sets, tasks, flows and runs.
See <https://www.openml.org/guide/api> for more information.

Author Giuseppe Casalicchio <giuseppe.casalicchio@stat.uni-muenchen.de>,
Bernd Bischl <bernd_bischl@gmx.net>,
Dominik Kirchhoff <dom.kirchhoff@gmail.com>,
Michel Lang <michellang@gmail.com>,
Benjamin Hofner <benjamin.hofner@fau.de>,
Jakob Bossek <j.bossek@gmail.com>,
Pascal Kerschke <kerschke@uni-muenster.de>,
Joaquin Vanschoren <joaquin.vanschoren@gmail.com>

Maintainer Giuseppe Casalicchio <giuseppe.casalicchio@stat.uni-muenchen.de>

License BSD_3_clause + file LICENSE

URL https://github.com/openml/openml-r

BugReports https://github.com/openml/openml-r/issues

Depends R (>= 3.0.2), mlr (>= 2.11)

Suggests testthat, xml2, randomForest, rpart, RWeka, farff, knitr, rmarkdown, R.rsp, lintr (>= 1.0.1)

Imports backports (>= 1.1.0), BBmisc (>= 1.11), checkmate (>= 1.8.2), ParamHelpers (>= 1.10), data.table, digest, httr, stringi, XML, jsonlite, memoise (>= 1.0.0), stats, curl (>= 3.1)

LazyData yes

ByteCompile yes

Version 1.8

RoxygenNote 6.0.1

VignetteBuilder R.rsp

NeedsCompilation no

Repository CRAN

Date/Publication 2018-03-02 06:27:46 UTC

R topics documented:

`chunkOMLlist` *Do chunked listings*

Description

Allows you to do multiple chunked requests with the listOML* functions. The request will be repeated until `total.limit` is reached or until there are no more results available on the server.

Usage

```
chunkOMLlist(listfun, ..., total.limit = 1e+05, chunk.limit = 1000)
```

Arguments

`listfun`	[`character(1)`] the listing function for which you want to do chunked requests.
`...`	[`ANY`] arguments are passed to the function specified in `listfun`.
`total.limit`	[`integer`] the total limit of results that should be listed. Set this to a high number to get all available results from the server.
`chunk.limit`	[`integer`] the limit for a single request. If you reduce this number, the number of server requests will increase.

See Also

Other listing functions: `listOMLDataSetQualities`, `listOMLDataSets`, `listOMLEstimationProcedures`, `listOMLEvaluationMeasures`, `listOMLFlows`, `listOMLRuns`, `listOMLSetup`, `listOMLStudies`, `listOMLTaskTypes`, `listOMLTasks`

clearOMLCache *Clear cache directories*

Description

Delete all cached objects and recreate cache directories.

Usage

```
clearOMLCache()
```

Examples

```
# \dontrun{
#   clearOMLCache()
# }
```

configuration *OpenML configuration.*

Description

After loading the package, it tries to find a configuration in your home directory. The R command `path.expand("~/.openml/config")` gives you the full path to the configuration file on your operating system.

For further information please read the vignette.

Note

By default the cache directory is located in a temporary directory and the cache will be deleted in between R sessions. We thus recommend to set the cache directory by hand.

See Also

Other config: `getOMLConfig`, `loadOMLConfig`, `saveOMLConfig`, `setOMLConfig`

convertMlrLearnerToOMLFlow

Converts an OMLFlow to an mlr learner.

Description

Creates an OMLFlow for an **mlr** Learner] Required if you want to upload an mlr learner to the OpenML server.

Usage

```
convertMlrLearnerToOMLFlow(lrn, name = paste0("mlr.", lrn$id),
  description = NULL, ...)
```

Arguments

lrn	[Learner] The mlr learner.
name	[character(1)] The name of the flow object. Default is the learner ID with the prefix "mlr" prepended.
description	[character(1)] An optional description of the learner. Default is a short specification of the learner and the associated package.
...	[any] Further optional parameters that are passed to makeOMLFlow.

Value

OMLFlow .

convertMlrTaskToOMLDataSet

Converts a mlr task to an OpenML data set.

Description

Converts a Task to an OMLDataSet.

Usage

```
convertMlrTaskToOMLDataSet(task, description = NULL)
```

Arguments

task	[Task] A mlr task.
description	[character(1)|OMLDataSetDescription] Either an OMLDataSetDescription or a character(1) that describes the data. For the latter, all other relevant information is autogenerated from the Task.

Value

OMLDataSet .

See Also

Other data set-related functions: OMLDataSetDescription, OMLDataSet, convertOMLDataSetToMlr, deleteOMLObject, getOMLDataSet, listOMLDataSets, tagOMLObject, uploadOMLDataSet

---

convertOMLDataSetToMlr

Convert an OpenML data set to mlr task.

Description

Converts an OMLDataSet to a Task.

Usage

```
convertOMLDataSetToMlr(obj, mlr.task.id = "<oml.data.name>",
  task.type = NULL, target = obj$desc$default.target.attribute,
  ignore.flagged.attributes = TRUE, drop.levels = TRUE,
  fix.colnames = TRUE, verbosity = NULL)
```

Arguments

obj	[OMLDataSet] The object that should be converted.
mlr.task.id	[character(1)] Id string for Task object. The strings <oml.data.name>, <oml.data.id> and <oml.data.version> will be replaced by their respective values contained in the OMLDataSet object. Default is <oml.data.name>.
task.type	[character(1)] As we only pass the data set, we need to define the task type manually. Possible are: "Supervised Classification", "Supervised Regression", "Survival Analysis". Default is NULL which means to guess it from the target column in the data set. If that is a factor or a logical, we choose classification. If it is numeric we choose regression. In all other cases an error is thrown.

target [character]
The target for the classification/regression task. Default is the default.target.attribute of the OMLDataSetDescription.

ignore.flagged.attributes
[logical(1)]
Should those features that are listed in the data set description slot "ignore.attribute" be removed? Default is TRUE.

drop.levels [logical(1)]
Should empty factor levels be dropped in the data? Default is TRUE.

fix.colnames [logical(1)]
Should colnames of the data be fixed using make.names? Default is TRUE.

verbosity [integer(1)]
Print verbose output on console? Possible values are:
0: normal output,
1: info output,
2: debug output.
Default is set via setOMLConfig.

Value

Task .

See Also

Other data set-related functions: OMLDataSetDescription, OMLDataSet, convertMlrTaskToOMLDataSet, deleteOMLObject, getOMLDataSet, listOMLDataSets, tagOMLObject, uploadOMLDataSet

Examples

```
# \dontrun{
#  library("mlr")
#  autosOML = getOMLDataSet(data.id = 9)
#  autosMlr = convertOMLDataSetToMlr(autosOML)
# }
```

---

convertOMLFlowToMlr *Converts a flow to a mlr learner.*

Description

Converts an OMLFlow that was originally created with the OpenML R-package to a Learner.

Usage

```
convertOMLFlowToMlr(flow)
```

Arguments

flow	[OMLFlow] The flow object.

Value

Learner .

See Also

Other flow-related functions: deleteOMLObject, getOMLFlow, listOMLFlows, makeOMLFlowParameter, makeOMLFlow, tagOMLObject

---

convertOMLMlrRunToBMR *Convert* OMLMlrRun*s to a* BenchmarkResult.

Description

Converts one or more OMLMlrRuns to a BenchmarkResult.

Usage

```
convertOMLMlrRunToBMR(...)
```

Arguments

...	[OMLMlrRun] One or more OMLMlrRuns

Value

BenchmarkResult .

See Also

Other run-related functions: convertOMLRunToBMR, deleteOMLObject, getOMLRun, listOMLRuns, makeOMLRunParameter, makeOMLRun, tagOMLObject, uploadOMLRun

convertOMLRunToBMR *Convert an OpenML run set to a benchmark result for mlr.*

Description

Converts an OMLRun to a BenchmarkResult.

Usage

```
convertOMLRunToBMR(run, measures = run$task.evaluation.measure,
  recompute = FALSE)
```

Arguments

run	[OMLRun] The run that should be converted.
measures	[character] Character describing the measures (see listOMLEvaluationMeasures) that will be converted into mlr measures and are then used in the BenchmarkResult. Currently, not all measures from OpenML can be converted into mlr measures.
recompute	[logical(1)] Should the measures be recomputed with mlr using the predictions? Currently recomputing is not supported.

Value

BenchmarkResult .

See Also

Other run-related functions: convertOMLMlrRunToBMR, deleteOMLObject, getOMLRun, listOMLRuns, makeOMLRunParameter, makeOMLRun, tagOMLObject, uploadOMLRun

convertOMLTaskToMlr *Convert an OpenML task to mlr.*

Description

Converts an OMLTask to a list of Task, ResampleInstance and Measure.

Usage

```
convertOMLTaskToMlr(obj, measures = NULL, mlr.task.id = "<oml.data.name>",
  ignore.flagged.attributes = TRUE, drop.levels = TRUE, verbosity = NULL)
```

Arguments

obj	[OMLTask] The OML task object that should be converted.
measures	[Measure] Additional measures that should be computed.
mlr.task.id	[character(1)] Id string for Task object. The strings <oml.data.name>, <oml.data.id>, <oml.data.version> and <oml.task.id> will be replaced by their respective values contained in the OMLTask object. Default is <oml.data.name>.
ignore.flagged.attributes	[logical(1)] Should those features that are listed in the data set description slot "ignore.attribute" be removed? Default is TRUE.
drop.levels	[logical(1)] Should empty factor levels be dropped in the data? Default is TRUE.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

list A list with the following objects:

mlr.task [Task]

mlr.rin [ResampleInstance]

mlr.measures [list of Measures to optimize for.

See Also

Other task-related functions: deleteOMLObject, getOMLTask, listOMLTaskTypes, listOMLTasks, makeOMLTask, tagOMLObject

Examples

```
# \dontrun{
#  library("mlr")
#  vinnieOML = getOMLTask(task.id = 4845)
#  vinnieMlr = convertOMLTaskToMlr(vinnieOML)
# }
```

`deleteOMLObject` *Delete an OpenML object.*

Description

This will delete one of your uploaded datasets, tasks, flows or runs. Note that you can only delete the objects you uploaded.

Usage

```
deleteOMLObject(id, object = c("data", "task", "flow", "run"),
  verbosity = NULL)
```

Arguments

`id`	`[integer(1)]` The ID of the respective object.
`object`	`[character(1)]` A character that specifies the object you want to delete from the server. Can be either `"data"`, `"task"`, `"flow"` or `"run"`.
`verbosity`	`[integer(1)]` Print verbose output on console? Possible values are: `0`: normal output, `1`: info output, `2`: debug output. Default is set via `setOMLConfig`.

See Also

Other data set-related functions: `OMLDataSetDescription`, `OMLDataSet`, `convertMlrTaskToOMLDataSet`, `convertOMLDataSetToMlr`, `getOMLDataSet`, `listOMLDataSets`, `tagOMLObject`, `uploadOMLDataSet`

Other task-related functions: `convertOMLTaskToMlr`, `getOMLTask`, `listOMLTaskTypes`, `listOMLTasks`, `makeOMLTask`, `tagOMLObject`

Other flow-related functions: `convertOMLFlowToMlr`, `getOMLFlow`, `listOMLFlows`, `makeOMLFlowParameter`, `makeOMLFlow`, `tagOMLObject`

Other run-related functions: `convertOMLMlrRunToBMR`, `convertOMLRunToBMR`, `getOMLRun`, `listOMLRuns`, `makeOMLRunParameter`, `makeOMLRun`, `tagOMLObject`, `uploadOMLRun`

extractOMLStudyIds *Extract IDs of a OMLStudy object*

Description

Extracts either all data.ids, task.ids, flow.ids or run.ids from an OMLStudy object.

Usage

```
extractOMLStudyIds(object, type)
```

Arguments

object	[OMLStudy] The OMLStudy object.
type	[character(1)] A character that specifies which ids should be extracted from the study. Can be either "data.id", "task.id", "flow.id" or "run.id".

Value

numeric .

getCachedOMLDataSetStatus *Check status of cached datasets.*

Description

The caching mechanism is fine, but sometimes you might want to work on a dataset, which is already cached and has been deactivated in the meanwhile. This function can be used to determine the status of all cached datasets.

Usage

```
getCachedOMLDataSetStatus(show.warnings = TRUE, ...)
```

Arguments

show.warnings	[logical(1)] Show warning if there are deactivated datasets in cache? Default is TRUE.
...	Arguments passed to listOMLDataSets

Value

data.frame

Examples

```
# \dontrun{
#  getCachedOMLDataSetStatus()
# }
```

`getOMLConfig` *Get OpenML configuration.*

Description

Returns a list of OpenML configuration settings.

Usage

```
getOMLConfig()
```

Value

`list` of current configuration variables with class "OMLConfig".

See Also

Other config: `configuration`, `loadOMLConfig`, `saveOMLConfig`, `setOMLConfig`

Examples

```
getOMLConfig()
```

`getOMLDataSet` *Get an OpenML data set.*

Description

Given a data set ID, the corresponding `OMLDataSet` will be downloaded (if not in cache) and returned.

Note that data splits and other task-related information are not included in an `OMLDataSet`. Tasks can be downloaded with `getOMLTask`.

Usage

```
getOMLDataSet(data.id = NULL, data.name = NULL, data.version = NULL,
  cache.only = FALSE, verbosity = NULL)
```

Arguments

data.id	[integer(1)] ID of the data set.
data.name	[character(1)] Data set name. This is an alternative to data.id. Default is NULL.
data.version	[integer(1)] Version number of the data set with name data.name. Default is NULL. Ignored if data.id is passed.
cache.only	[logical(1)] Only try to retrieve the object from cache. Will result in error if the object is not found. Default is FALSE.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

OMLDataSet .

Note

One of data.id or data.name must be passed.

See Also

Other downloading functions: getOMLDataSetQualities, getOMLFlow, getOMLRun, getOMLStudy, getOMLTask

Other data set-related functions: OMLDataSetDescription, OMLDataSet, convertMlrTaskToOMLDataSet, convertOMLDataSetToMlr, deleteOMLObject, listOMLDataSets, tagOMLObject, uploadOMLDataSet

Examples

```
# \dontrun{
#  dat = getOMLDataSet(data.id = 9)
#
#  # this object contains the data ($data)
#  # and meta information
#  str(dat, 1)
#  summary(dat$data)
# }
```

getOMLDataSetQualities *List available OpenML qualities with values for given data set.*

Description

The returned data.frame contains data set quality "name"s and values "value".

Usage

```
getOMLDataSetQualities(data.id, verbosity = NULL, name = NULL)
```

Arguments

data.id	[integer(1)] ID of the data set.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.
name	[character] Returns only the data qualities from "name" (see also listOMLDataSetQualities). Default is NULL and uses all available data qualities.

Value

data.frame .

See Also

Other downloading functions: getOMLDataSet, getOMLFlow, getOMLRun, getOMLStudy, getOMLTask

Examples

```
# \dontrun{
#   a = getOMLDataSetQualities(data.id = 9)
#   a[a$name == "number.of.missing.values", ]
#   getOMLDataSetQualities(data.id = 9, name = "number.of.missing.values")
# }
```

getOMLFlow *Download an OpenML flow.*

Description

Given an flow id, the corresponding OMLFlow is downloaded if not already available in cache.

Usage

```
getOMLFlow(flow.id, cache.only = FALSE, verbosity = NULL)
```

Arguments

flow.id	[integer(1)] ID of the implementation of an OpenML flow.
cache.only	[logical(1)] Only try to retrieve the object from cache. Will result in error if the object is not found. Default is FALSE.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

OMLFlow .

See Also

Other downloading functions: getOMLDataSetQualities, getOMLDataSet, getOMLRun, getOMLStudy, getOMLTask

Other flow-related functions: convertOMLFlowToMlr, deleteOMLObject, listOMLFlows, makeOMLFlowParameter, makeOMLFlow, tagOMLObject

Examples

```
# \dontrun{
#  r_ctree = getOMLFlow(flow.id = 2569)
#  weka_bagging = getOMLFlow(flow.id = 2286)
# }
```

getOMLRun *Get an OpenML run.*

Description

Given an run id, the corresponding OMLRun including all server and user computed metrics is downloaded if not already available in cache.

Usage

```
getOMLRun(run.id, cache.only = FALSE, only.xml = FALSE, verbosity = NULL)
```

Arguments

run.id	[integer(1)] The run ID.
cache.only	[logical(1)] Only try to retrieve the object from cache. Will result in error if the object is not found. Default is FALSE.
only.xml	[logical(1)] Should only the XML be downloaded?
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

OMLRun .

See Also

Other downloading functions: getOMLDataSetQualities, getOMLDataSet, getOMLFlow, getOMLStudy, getOMLTask

Other run-related functions: convertOMLMlrRunToBMR, convertOMLRunToBMR, deleteOMLObject, listOMLRuns, makeOMLRunParameter, makeOMLRun, tagOMLObject, uploadOMLRun

Examples

```
# \dontrun{
#  runs_ctree = listOMLRuns(flow.id = 2569)
#  run1 = getOMLRun(run.id = runs_ctree$run.id[1])
#  str(run1, 1)
# }
```

getOMLRunParList *Extract OMLRunParList from run*

Description

Extracts the seed information as OMLRunParList from a OMLRun.

Usage

```
getOMLRunParList(run)
```

Arguments

run	[OMLRun] A OMLRun

Value

OMLRunParList .

getOMLSeedParList *Extract OMLSeedParList from run*

Description

Extracts the seed information as OMLSeedParList from a OMLRun.

Usage

```
getOMLSeedParList(run)
```

Arguments

run	[OMLRun] A OMLRun

Value

OMLSeedParList .

getOMLStudy *Get OpenML Study information.*

Description

A OpenML study is a collection of OpenML objects with a specific tag defined by the user (i.e. "study_X"). If you create a study through the website https://www.openml.org/new/study, you can also specify an alias which can be used to access the study.

Usage

```
getOMLStudy(study = NULL, verbosity = NULL)
```

Arguments

study	[numeric(1)|character(1)] Either the id or the alias of a study.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

OMLStudy .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

See Also

Other downloading functions: getOMLDataSetQualities, getOMLDataSet, getOMLFlow, getOMLRun, getOMLTask

getOMLTask *Get an OpenML task.*

Description

Given a task ID, the corresponding OMLTask will be downloaded (if not in cache) and returned.

Usage

```
getOMLTask(task.id, cache.only = FALSE, verbosity = NULL)
```

Arguments

task.id	[integer(1)] Task ID.
cache.only	[logical(1)] Only try to retrieve the object from cache. Will result in error if the object is not found. Default is FALSE.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

OMLTask .

See Also

Other downloading functions: getOMLDataSetQualities, getOMLDataSet, getOMLFlow, getOMLRun, getOMLStudy

Other task-related functions: convertOMLTaskToMlr, deleteOMLObject, listOMLTaskTypes, listOMLTasks, makeOMLTask, tagOMLObject

Examples

```
# # Download task and access relevant information to start running experiments
# \dontrun{
#   task = getOMLTask(1)
#   task
#   task$task.type
#   task$input$data.set
#   head(task$input$data.set$data)
# }
```

listOMLDataSetQualities *List available OpenML qualities names.*

Description

The returned data.frame contains quality name "name".

Usage

```
listOMLDataSetQualities(verbosity = NULL)
```

Arguments

verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSets, listOMLEstimationProcedures, listOMLEvaluationMeasure
listOMLFlows, listOMLRuns, listOMLSetup, listOMLStudies, listOMLTaskTypes, listOMLTasks

Examples

```
# \dontrun{
#  listOMLDataSetQualities()
# }
```

`listOMLDataSets` *List the first 5000 OpenML data sets.*

Description

The returned `data.frame` contains the data set id "data.id", the "status" ("active", "deactivated", "in_preparation") and describing data qualities. Note that by default only the first 5000 data sets will be returned (due to the argument "limit = 5000").

Usage

```
listOMLDataSets(number.of.instances = NULL, number.of.features = NULL,
  number.of.classes = NULL, number.of.missing.values = NULL, tag = NULL,
  data.name = NULL, limit = 5000, offset = NULL, status = "active",
  verbosity = NULL)
```

Arguments

`number.of.instances`
: [numeric(1) | numeric(2)]
If not NULL, subsets the entries with respect to the given values or, if a vector of length 2 is passed, the given ranges.

`number.of.features`
: [numeric(1) | numeric(2)]
If not NULL, it subsets the entries with respect to the given values or, if a vector of length 2 is passed, the given range.

`number.of.classes`
: [numeric(1) | numeric(2)]
If not NULL, subsets the entries with respect to the given values or, if a vector of length 2 is passed, the given ranges.

`number.of.missing.values`
: [numeric(1) | numeric(2)]
If not NULL, subsets the entries with respect to the given values or, if a vector of length 2 is passed, the given ranges.

`tag`
: [character]
If not NULL only entries with the corresponding tags are listed.

`data.name`
: [character(1)]
Name of the data set.

`limit`
: [numeric(1)]
Optional. The maximum number of entries to return. Without specifying `offset`, it returns the first 'limit' entries. Setting `limit = NULL` returns all available entries.

`offset`
: [numeric(1)]
Optional. The offset to start from. Should be indices starting from 0, which do not refer to IDs. Is ignored when no `limit` is given.

status [character]
Subsets the results according to the status. Possible values are {"active", "deactivated", "in_prepa
Default is "active".

verbosity [integer(1)]
Print verbose output on console? Possible values are:
0: normal output,
1: info output,
2: debug output.
Default is set via setOMLConfig.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSetQualities, listOMLEstimationProcedures, listOMLEvaluationMeasures, listOMLFlows, listOMLRuns, listOMLSetup, listOMLStudies, listOMLTaskTypes, listOMLTasks

Other data set-related functions: OMLDataSetDescription, OMLDataSet, convertMlrTaskToOMLDataSet, convertOMLDataSetToMlr, deleteOMLObject, getOMLDataSet, tagOMLObject, uploadOMLDataSet

Examples

```
# \dontrun{
#  datasets = listOMLDataSets()
#  tail(datasets)
# }
```

---

listOMLEstimationProcedures

List available estimation procedures.

Description

The returned data.frame contains the est.id and the corresponding name of the estimation procedure.

Usage

```
listOMLEstimationProcedures(verbosity = NULL)
```

Arguments

verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSetQualities, listOMLDataSets, listOMLEvaluationMeasures, listOMLFlows, listOMLRuns, listOMLSetup, listOMLStudies, listOMLTaskTypes, listOMLTasks

Examples

```
# \dontrun{
#   listOMLEstimationProcedures()
# }
```

---

listOMLEvaluationMeasures

List available OpenML evaluation measures.

Description

The names of all evaluation measures which are used in at least one run are returned in a data.frame.

Usage

```
listOMLEvaluationMeasures(verbosity = NULL)
```

Arguments

verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSetQualities, listOMLDataSets, listOMLEstimationProcedures, listOMLFlows, listOMLRuns, listOMLSetup, listOMLStudies, listOMLTaskTypes, listOMLTasks

Examples

```
# \dontrun{
#  listOMLEvaluationMeasures()
# }
```

---

listOMLFlows *List all registered OpenML flows.*

Description

The returned data.frame contains the flow id "fid", the flow name ("full.name" and "name"), version information ("version" and "external.version") and the uploader ("uploader") of all registered OpenML flows.

Usage

```
listOMLFlows(tag = NULL, limit = NULL, offset = NULL, verbosity = NULL)
```

Arguments

tag	[character] If not NULL only entries with the corresponding tags are listed.
limit	[numeric(1)] Optional. The maximum number of entries to return. Without specifying offset, it returns the first 'limit' entries. Setting limit = NULL returns all available entries.
offset	[numeric(1)] Optional. The offset to start from. Should be indices starting from 0, which do not refer to IDs. Is ignored when no limit is given.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSetQualities, listOMLDataSets, listOMLEstimationProcedures, listOMLEvaluationMeasures, listOMLRuns, listOMLSetup, listOMLStudies, listOMLTaskTypes, listOMLTasks

Other flow-related functions: convertOMLFlowToMlr, deleteOMLObject, getOMLFlow, makeOMLFlowParameter, makeOMLFlow, tagOMLObject

Examples

```
# \dontrun{
#  flows = listOMLFlows()
#  tail(flows)
# }
```

`listOMLRunEvaluations` *List run results of a task.*

Description

Retrieves all run results for task(s) (`task.id`), flow(s) (`flow.id`) run(s) (`run.id`) or uploaders(s) (`uploader.id` and returns a `data.frame`. Each row contains, among others, the run id "rid". Alternatively the function can be passed a single `tag` to list only runs with the corresponding tag associated.

Usage

```
listOMLRunEvaluations(task.id = NULL, flow.id = NULL, run.id = NULL,
  uploader.id = NULL, tag = NULL, limit = NULL, offset = NULL,
  verbosity = NULL, evaluation.measure = NULL,
  show.array.measures = FALSE, extend.flow.name = TRUE)
```

Arguments

`task.id`	[`integer`] a single ID or a vector of IDs of the task(s).
`flow.id`	[`integer`] a single ID or a vector of IDs of the flow(s).
`run.id`	[`integer`] a single ID or a vector of IDs of the run(s).
`uploader.id`	[`integer`] a single ID or a vector of IDs of uploader profile(s).
`tag`	[`character`] If not `NULL` only entries with the corresponding `tag`s are listed.
`limit`	[`numeric(1)`] Optional. The maximum number of entries to return. Without specifying `offset`, it returns the first '`limit`' entries. Setting `limit = NULL` returns all available entries.
`offset`	[`numeric(1)`] Optional. The offset to start from. Should be indices starting from 0, which do not refer to IDs. Is ignored when no `limit` is given.
`verbosity`	[`integer(1)`] Print verbose output on console? Possible values are: `0`: normal output, `1`: info output, `2`: debug output. Default is set via `setOMLConfig`.

evaluation.measure
: [character(1)]
Use this to speedup your request. It restricts the results to only one evaluation measure (see listOMLEvaluationMeasures for possible values). Default is NULL, which means that no restriction is going to happen and all possible evaluation measures will be returned.

show.array.measures
: [logical(1)]
Should measures that return an array instead of a single skalar value be shown (e.g. confusion matrix, predictive accuracy within each class)? Default is FALSE.

extend.flow.name
: [logical(1)]
Adds a column flow.version that refers to the version number of the flow and a column flow.source containing the prefix of the flow that specifies the source of the flow (i.e. weka, R) and a column learner.name that refers to the learner. Default is TRUE.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

Examples

```
# \dontrun{
#  # get run results of task 6 (as many rows as runs for this task)
#  rev_tid6 = listOMLRunEvaluations(task.id = 6L)
#  str(rev_tid6)
#
#  # get run results of run 8 (one row)
#  rev_rid8 = listOMLRunEvaluations(run.id = 8)
#  str(rev_rid8)
# }
```

---

listOMLRuns *List OpenML runs.*

Description

This function returns information on all OpenML runs that match certain task.id(s), run.id(s), flow ID flow.id and/or uploader.id(s). Alternatively the function can be passed a single tag to list only runs with the corresponding tag associated.

Usage

```
listOMLRuns(task.id = NULL, flow.id = NULL, run.id = NULL,
  uploader.id = NULL, tag = NULL, limit = NULL, offset = NULL,
  verbosity = NULL)
```

Arguments

task.id	[integer] a single ID or a vector of IDs of the task(s).
flow.id	[integer] a single ID or a vector of IDs of the flow(s).
run.id	[integer] a single ID or a vector of IDs of the run(s).
uploader.id	[integer] a single ID or a vector of IDs of uploader profile(s).
tag	[character] If not NULL only entries with the corresponding tags are listed.
limit	[numeric(1)] Optional. The maximum number of entries to return. Without specifying offset, it returns the first 'limit' entries. Setting limit = NULL returns all available entries.
offset	[numeric(1)] Optional. The offset to start from. Should be indices starting from 0, which do not refer to IDs. Is ignored when no limit is given.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSetQualities, listOMLDataSets, listOMLEstimationProcedures, listOMLEvaluationMeasures, listOMLFlows, listOMLSetup, listOMLStudies, listOMLTaskTypes, listOMLTasks

Other run-related functions: convertOMLMlrRunToBMR, convertOMLRunToBMR, deleteOMLObject, getOMLRun, makeOMLRunParameter, makeOMLRun, tagOMLObject, uploadOMLRun

Examples

```
# \dontrun{
#   runs_ctree = listOMLRuns(flow.id = 2569)
#   head(runs_ctree)
# }
```

`listOMLSetup` *List hyperparameter settings*

Description

Each run has a `setup.id`, i.e. an ID for the hyperparameter settings of the flow that produced the run. This function allows the listing of hyperparameter settings.

Usage

```
listOMLSetup(setup.id = NULL, flow.id = NULL, limit = 1000,
  offset = NULL, verbosity = NULL)
```

Arguments

`setup.id`	[`integer(1)`] ID of the setup (which is basically an ID for the parameter configuration).
`flow.id`	[`integer(1)`] ID of the implementation of an OpenML flow.
`limit`	[`numeric(1)`] Optional. The maximum number of entries to return. Without specifying `offset`, it returns the first '`limit`' entries. Setting `limit = NULL` returns all available entries.
`offset`	[`numeric(1)`] Optional. The offset to start from. Should be indices starting from 0, which do not refer to IDs. Is ignored when no `limit` is given.
`verbosity`	[`integer(1)`] Print verbose output on console? Possible values are: `0`: normal output, `1`: info output, `2`: debug output. Default is set via `setOMLConfig`.

Value

`data.frame` .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling `forget` on the function manually.

See Also

Other listing functions: `chunkOMLlist`, `listOMLDataSetQualities`, `listOMLDataSets`, `listOMLEstimationProcedures`, `listOMLEvaluationMeasures`, `listOMLFlows`, `listOMLRuns`, `listOMLStudies`, `listOMLTaskTypes`, `listOMLTasks`

Examples

```
# \dontrun{
#   listOMLSetup(limit = 1)
# }
```

---

listOMLStudies *list OpenML Studies.*

Description

Retrives a list of available studies.

Usage

```
listOMLStudies(verbosity = NULL)
```

Arguments

verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via `setOMLConfig`.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling `forget` on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSetQualities, listOMLDataSets, listOMLEstimationProcedures, listOMLEvaluationMeasures, listOMLFlows, listOMLRuns, listOMLSetup, listOMLTaskTypes, listOMLTasks

---

listOMLTasks *List the first 5000 OpenML tasks.*

Description

The returned data.frame contains the task_id, the data set id data.id, the status and some describing data qualities. Note that by default only the first 5000 data sets will be returned (due to the argument "limit = 5000").

Usage

```
listOMLTasks(task.type = NULL, estimation.procedure = NULL,
  evaluation.measures = NULL, number.of.instances = NULL,
  number.of.features = NULL, number.of.classes = NULL,
  number.of.missing.values = NULL, tag = NULL, data.name = NULL,
  data.tag = NULL, limit = 5000, offset = NULL, status = "active",
  verbosity = NULL)
```

Arguments

task.type
: [character(1)]
If not NULL, only tasks belonging to the given task type are listed. Use listOMLTaskTypes()$name to see possible values for task.type. The default is NULL, which means that tasks with all available task types are listed.

estimation.procedure
: [character]
If not NULL, only tasks belonging the given estimation procedures are listed. Use listOMLEstimationProcedures()$name to see possible values for estimation.procedure. The default is NULL, which means that tasks with all available estimation procedures are listed.

evaluation.measures
: [character]
If not NULL, only tasks belonging the given evaluation measures are listed. Use listOMLEvaluationMeasures()$name to see possible values for evaluation.measures. The default is NULL, which means that tasks with all available evaluation measures are listed.

number.of.instances
: [numeric(1) | numeric(2)]
If not NULL, subsets the entries with respect to the given values or, if a vector of length 2 is passed, the given ranges.

number.of.features
: [numeric(1) | numeric(2)]
If not NULL, it subsets the entries with respect to the given values or, if a vector of length 2 is passed, the given range.

number.of.classes
: [numeric(1) | numeric(2)]
If not NULL, subsets the entries with respect to the given values or, if a vector of length 2 is passed, the given ranges.

number.of.missing.values
: [numeric(1) | numeric(2)]
If not NULL, subsets the entries with respect to the given values or, if a vector of length 2 is passed, the given ranges.

tag
: [character]
If not NULL only entries with the corresponding tags are listed.

data.name
: [character(1)]
Name of the data set.

data.tag
: [character(1)]
Refers to the tag of the dataset the task is based on. If not NULL only tasks with the corresponding data.tag are listed.

limit
: [numeric(1)]
Optional. The maximum number of entries to return. Without specifying offset, it returns the first 'limit' entries. Setting limit = NULL returns all available entries.

offset
: [numeric(1)]
Optional. The offset to start from. Should be indices starting from 0, which do not refer to IDs. Is ignored when no limit is given.

status
: [character]
Subsets the results according to the status. Possible values are {"active", "deactivated", "in_prepa
Default is "active".

verbosity
: [integer(1)]
Print verbose output on console? Possible values are:
0: normal output,
1: info output,
2: debug output.
Default is set via setOMLConfig.

Value

data.frame .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling forget on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSetQualities, listOMLDataSets, listOMLEstimationProcedures, listOMLEvaluationMeasures, listOMLFlows, listOMLRuns, listOMLSetup, listOMLStudies, listOMLTaskTypes

Other task-related functions: convertOMLTaskToMlr, deleteOMLObject, getOMLTask, listOMLTaskTypes, makeOMLTask, tagOMLObject

Examples

```
# \dontrun{
#  tasks = listOMLTasks()
#  head(tasks)
# }
```

`listOMLTaskTypes` *List available OpenML task types.*

Description

The returned `data.frame` contains the type `id` and the character name of the OpenML task type.

Usage

```
listOMLTaskTypes(verbosity = NULL)
```

Arguments

`verbosity`	`[integer(1)]` Print verbose output on console? Possible values are: `0`: normal output, `1`: info output, `2`: debug output. Default is set via `setOMLConfig`.

Value

`data.frame` .

Note

This function is memoised. I.e., if you call this function twice in a running R session, the first call will query the server and store the results in memory while the second and all subsequent calls will return the cached results from the first call. You can reset the cache by calling `forget` on the function manually.

See Also

Other listing functions: chunkOMLlist, listOMLDataSetQualities, listOMLDataSets, listOMLEstimationProcedures, listOMLEvaluationMeasures, listOMLFlows, listOMLRuns, listOMLSetup, listOMLStudies, listOMLTasks

Other task-related functions: convertOMLTaskToMlr, deleteOMLObject, getOMLTask, listOMLTasks, makeOMLTask, tagOMLObject

Examples

```
# \dontrun{
#   listOMLTaskTypes()
# }
```

---

loadOMLConfig *Load OpenML configuration.*

Description

Loads the OpenML config file from the disk and overwrites the current OpenML config. If there is no API key in the configuration file, the key is retrieved from the environment variable "OPENMLAPIKEY" (if defined).

Usage

```
loadOMLConfig(path = "~/.openml/config", assign = TRUE)
```

Arguments

path	[character(1)] Full path location of the config file to be loaded.
assign	[logical(1)] Use the loaded configuration as the current configuration? If set to FALSE, the configuration is just returned by the function. Default is TRUE.

Value

list of current configuration variables with class "OMLConfig".

See Also

Other config: configuration, getOMLConfig, saveOMLConfig, setOMLConfig

Examples

```
# # if assign = FALSE nothing is changed
# # usually one would want assign = TRUE
# \dontrun{
#   loadOMLConfig(assign = FALSE)
# }
```

`makeOMLFlow` *Construct OMLFlow.*

Description

More details about the elements of a `OMLFlow` can be found in the XSD scheme.

Usage

```
makeOMLFlow(flow.id = NA_integer_, uploader = NA_integer_, name,
  version = NA_character_, external.version = NA_character_, description,
  creator = NA_character_, contributor = NA_character_,
  upload.date = NA_character_, licence = NA_character_,
  language = "English", full.description = NA_character_,
  installation.notes = NA_character_, dependencies = NA_character_,
  bibliographical.reference = NULL, implements = NA_character_,
  parameters = NULL, components = NULL, qualities = NULL,
  tags = NA_character_, source.url = NA_character_,
  binary.url = NA_character_, source.format = NA_character_,
  binary.format = NA_character_, source.md5 = NA_character_,
  binary.md5 = NA_character_, source.path = NA_character_,
  binary.path = NA_character_, object = NULL)
```

Arguments

`flow.id`	[integer(1)] ID of the flow. Generated by the server, based on name and version of the flow. Ignored when uploaded manually.
`uploader`	[integer(1)] The user that uploaded the flow. Added by the server. Ignored when uploaded manually.
`name`	[character(1)] The name of the flow. Name-version combinations should be unique. Allowed characters: () [] a-z A-Z 0-9 . _ - +
`version`	[character(1)] The version of the flow. Default is 1.0. Ignored at upload time.
`external.version`	[character(1)] An external version, defined by the user. In combination with the name, it must be unique.
`description`	[character(1)] A user description of the flow.
`creator`	[character] Optional. The persons/institutions that created the flow.

`contributor` [character]
Optional. (Minor) contributors to the workflow

`upload.date` [character(1)]
The date on which the flow was uploaded.
Format YYYY-mm-ddThh:MM:SS. Added by the server. Ignored when uploaded manually.

`licence` [character(1)]
Optional. Default is none, meaning Public Domain or "don't know/care".

`language` [character(1)]
Optional. Starts with one upper case letter, rest is lower case. Default is English.

`full.description`
[character(1)]
Optional. Full description of the workflow, e.g, man pages filled in by tool. This is a much more elaborate description than given in the 'description field'. It may include information about all components of the workflow.

`installation.notes`
[character(1)]
Optional. Additional hints on how to run the flow.

`dependencies` [character(1)]
Optional. The dependencies of the flow.

`bibliographical.reference`
[list]
An optional list containing information on bibliographical references in form of `OMLBibRef`.

`implements` [character(1)]
Ontological reference.

`parameters` [list]
The parameters of the flow. A list containing `OMLFlowParameter`s.

`components` [list]
A list containing `OMLFlow`s. Typically components of a workflow or subfunctions of an algorithm (e.g. kernels). Components can have their own parameters.

`qualities` [list]
Qualities of the algorithm. Each member of the list is an `OMLFlowQuality`.

`tags` [character]
Tags describing the algorithm.

`source.url` [character(1)]
URL from which the source code can be downloaded. Added by the server. Ignored when uploaded manually.

`binary.url` [character(1)]
URL from which the binary can be downloaded. Added by the server. Ignored when uploaded manually.

`source.format` [character(1)]
Format of the source file.

`binary.format` [character(1)]
Format of the binary file.

source.md5 [character(1)]
MD5 checksum to check if the source code was uploaded correctly.

binary.md5 [character(1)]
MD5 checksum to check if the binary code was uploaded correctly.

source.path [character(1)]
The path to the cached source file, once getOMLFlow was run.

binary.path [character(1)]
The path to the cached binary file, once getOMLFlow was run.

object [any]
(optional) Any R object referring to the flow.

See Also

Other flow-related functions: convertOMLFlowToMlr, deleteOMLObject, getOMLFlow, listOMLFlows, makeOMLFlowParameter, tagOMLObject

---

makeOMLRun *Construct OMLRun.*

Description

More details about the elements of a OMLRun can be found in the XSD scheme.

Usage

```
makeOMLRun(run.id = NA_integer_, uploader = NA_integer_,
  uploader.name = NA_character_, task.id, task.type = NA_character_,
  task.evaluation.measure = NA_character_, flow.id = NA_integer_,
  flow.name = NA_character_, setup.id = NA_integer_,
  setup.string = NA_character_, error.message = NA_character_,
  parameter.setting = list(), tags = NA_character_, predictions = NULL,
  input.data = makeOMLIOData(), output.data = makeOMLIOData())
```

Arguments

run.id [numeric(1)]
ID of the run. Added by server. Ignored when uploading a run.

uploader [numeric(1)]
ID of the user that uploaded the run. Added by server. Ignored when uploading a run.

uploader.name [character(1)]
Name of the user that uploaded the run. Ignored when uploading a run.

task.id [numeric(1)]
ID of the task that is solved in this run. This ID is given in the task description.

task.type [character(1)]
Task type of the run. See listOMLTaskTypes for all possible types.

task.evaluation.measure
[character(1)]
Evaluation measure used in the run.

flow.id [character(1)]
ID of the flow used to solve the task. Returned by the API when you upload the flow, or given in the flow description when you download an existing flow.

flow.name [character(1)]
Name of the flow.

setup.id [numeric(1)]
Unique ID of the used setup. Ignored when uploading a run (i.e., it will be searched based on the parameter settings).

setup.string [character(1)]
The CLI string that can invoke the learner with the correct parameter settings. This argument is optional.

error.message [character(1)]
Whenever an error occurs during the run, this can be reported here.

parameter.setting
[list]
A list of OMLRunParameters containing information on the parameter settings.

tags [character]
Optional tags describing the run.

predictions [data.frame]
The predictions of the run.

input.data [OMLIOData]
All data that served as input for the run. Added by server. Ignored when uploading.

output.data [OMLIOData]
All data that was the output of this run, i.e., predictions, evaluation scores. Most of this will be added by the server, but users can also provide evaluation scores for their own evaluation measures.

See Also

Other run-related functions: convertOMLMlrRunToBMR, convertOMLRunToBMR, deleteOMLObject, getOMLRun, listOMLRuns, makeOMLRunParameter, tagOMLObject, uploadOMLRun

---

makeOMLRunParList *Construct OMLRunParList.*

Description

Generate a list of OpenML run parameter settings for a given mlr learner.

Usage

```
makeOMLRunParList(mlr.lrn, component = NA_character_)
```

Arguments

mlr.lrn	[Learner] The mlr learner.
component	[character] If the learner is a (sub-)component of a flow, this component's name.

Value

A OMLRunParList which is a list of OMLRunParameters.

---

makeOMLSeedParList *Construct OMLSeedParList*

Description

Generate a list of OpenML seed parameter settings for a given seed.

Usage

```
makeOMLSeedParList(seed, prefix = "openml")
```

Arguments

seed	[numeric(1)] The seed.
prefix	[character] prefix for seed parameter names.

Value

A OMLSeedParList which is a list of OMLRunParameters that provide only information about the seed.

makeOMLTask *Construct OMLTask.*

Description

More details about the elements of a OMLTask can be found in the XSD scheme.

Usage

```
makeOMLTask(task.id, task.type, input, parameters = list(), output,
  tags = NA_character_)
```

Arguments

task.id	[integer(1)] The ID of this task. Generated by the API.
task.type	[character(1)] The task type of this task. Task types can be browsed and created on the OpenML website. See also listOMLTaskTypes for a list of all available tasks.
input	[list] The inputs given for this task (i.e. data.set, estimation.procedure, evaluation.measures, cost.matrix).
parameters	[list] Parameter settings for this task (depends on the task type).
output	[list] Outputs expected after running this task.
tags	[character] Optional tags describing the (data of the) task.

See Also

Other task-related functions: convertOMLTaskToMlr, deleteOMLObject, getOMLTask, listOMLTaskTypes, listOMLTasks, tagOMLObject

OMLDataSet *OMLDataSet.*

Description

An OMLDataSet consists of an OMLDataSetDescription, a data.frame containing the data set, the old and new column names and, finally, the target features.

The OMLDataSetDescription provides information on the data set, like the ID, name, version, etc. To see a full list of all elements, please see the XSD.

The slot colnames.old contains the original names, i.e., the column names that were uploaded to the server, while colnames.new contains the names that you will see when working with the data in R. Most of the time, old and new column names are identical. Only if the original names are not valid, the new ones will differ.

The slot target.features contains the column name(s) from the data.frame of the OMLDataSet that refer to the target feature(s).

Usage

```
makeOMLDataSet(desc, data, colnames.old = colnames(data),
  colnames.new = colnames(data), target.features)
```

Arguments

desc	[OMLDataSetDescription] Data set description.
data	[data.frame] The data set.
colnames.old	[character] Names of the features that were uploaded to the server.
colnames.new	[character] Names of the features that are displayed.
target.features	[character] Name(s) of the target feature(s).

Value

OMLDataSet

See Also

Other data set-related functions: OMLDataSetDescription, convertMlrTaskToOMLDataSet, convertOMLDataSetToMlr, deleteOMLObject, getOMLDataSet, listOMLDataSets, tagOMLObject, uploadOMLDataSet

Examples

```
data("airquality")
dsc = "Daily air quality measurements in New York, May to September 1973.
This data is taken from R."
cit = "Chambers, J. M., Cleveland, W. S., Kleiner, B. and Tukey, P. A. (1983) Graphical
Methods for Data Analysis. Belmont, CA: Wadsworth."
```

```
desc_airquality = makeOMLDataSetDescription(name = "airquality",
  description = dsc,
  creator = "New York State Department of Conservation (ozone data) and the National
    Weather Service (meteorological data)",
  collection.date = "May 1, 1973 to September 30, 1973",
  language = "English",
  licence = "GPL-2",
  url = "https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html",
  default.target.attribute = "Ozone",
  citation = cit,
  tags = "R")

airquality_oml = makeOMLDataSet(desc = desc_airquality,
  data = airquality,
  colnames.old = colnames(airquality),
  colnames.new = colnames(airquality),
  target.features = "Ozone")
```

`OMLDataSetDescription` *Construct OMLDataSetDescription.*

Description

Creates a description for an OMLDataSet. To see a full list of all elements, please see the XSD.

Usage

```
makeOMLDataSetDescription(id = 0L, name, version = "0", description,
  format = "ARFF", creator = NA_character_, contributor = NA_character_,
  collection.date = NA_character_, upload.date = as.POSIXct(Sys.time()),
  language = NA_character_, licence = NA_character_, url = NA_character_,
  default.target.attribute = NA_character_,
  row.id.attribute = NA_character_, ignore.attribute = NA_character_,
  version.label = NA_character_, citation = NA_character_,
  visibility = NA_character_, original.data.url = NA_character_,
  paper.url = NA_character_, update.comment = NA_character_,
  md5.checksum = NA_character_, status = NA_character_,
  tags = NA_character_)
```

Arguments

`id`	[integer(1)] Data set ID, autogenerated by the server. Ignored when set manually.
`name`	[character(1)] The name of the data set.
`version`	[character(1)] Version of the data set, autogenerated by the server. Ignored when set manually.

`description` [character(1)]
Description of the data set, given by the uploader.

`format` [character(1)]
Format of the data set. At the moment this is always "ARFF".

`creator` [character]
The person(s), that created this data set. Optional.

`contributor` [character]
People, that contibuted to this version of the data set (e.g., by reformatting). Optional.

`collection.date`
[character(1)]
The date the data was originally collected. Given by the uploader. Optional.

`upload.date` [POSIXt]
The date the data was uploaded. Added by the server. Ignored when set manually.

`language` [character(1)]
Language in which the data is represented. Starts with 1 upper case letter, rest lower case, e.g. 'English'

`licence` [character(1)]
Licence of the data. NA means: Public Domain or "don't know/care".

`url` [character(1)]
Valid URL that points to the data file.

`default.target.attribute`
[character]
The default target attribute, if it exists. Of course, tasks can be defined that use another attribute as target.

`row.id.attribute`
[character(1)]
The attribute that represents the row-id column, if present in the data set. Else NA.

`ignore.attribute`
[character(1)]
Attributes that should be excluded in modelling, such as identifiers and indexes. Optional.

`version.label` [character(1)]
Version label provided by user, something relevant to the user. Can also be a date, hash, or some other type of id.

`citation` [character(1)]
Reference(s) that should be cited when building on this data.

`visibility` [character(1)]
Who can see the data set. Typical values: 'Everyone', 'All my friends', 'Only me'. Can also be any of the user's circles.

`original.data.url`
[character(1)]
For derived data, the url to the original data set. This can be an OpenML data set, e.g. 'http://openml.org/d/1'.

paper.url [character(1)]
Link to a paper describing the data set.

update.comment [character(1)]
When the data set is updated, add an explanation here.

md5.checksum [character(1)]
MD5 checksum to check if the data set is downloaded without corruption. Can be ignored by user.

status [character(1)]
The status of the data set, autogenerated by the server. Ignored when set manually.

tags [character]
Optional tags for the data set.

See Also

Other data set-related functions: OMLDataSet, convertMlrTaskToOMLDataSet, convertOMLDataSetToMlr, deleteOMLObject, getOMLDataSet, listOMLDataSets, tagOMLObject, uploadOMLDataSet

Examples

```
data("airquality")
dsc = "Daily air quality measurements in New York, May to September 1973.
This data is taken from R."
cit = "Chambers, J. M., Cleveland, W. S., Kleiner, B. and Tukey, P. A. (1983) Graphical
Methods for Data Analysis. Belmont, CA: Wadsworth."
desc_airquality = makeOMLDataSetDescription(name = "airquality",
  description = dsc,
  creator = "New York State Department of Conservation (ozone data) and the National
    Weather Service (meteorological data)",
  collection.date = "May 1, 1973 to September 30, 1973",
  language = "English",
  licence = "GPL-2",
  url = "https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html",
  default.target.attribute = "Ozone",
  citation = cit,
  tags = "R")

airquality_oml = makeOMLDataSet(desc = desc_airquality,
  data = airquality,
  colnames.old = colnames(airquality),
  colnames.new = colnames(airquality),
  target.features = "Ozone")
```

---

populateOMLCache *Download a bunch of OpenML objects to cache.*

Description

Given a set of OML object ids, the function populates the cache directory by downloading the corresponding objects. This can avoid network access in later experiments, as you can retrieve all objects from the cache on disk. This is of particular interest in highly parallel computations on a cluster with a shared file system.

Usage

```
populateOMLCache(data.ids = integer(0L), task.ids = integer(0L),
  flow.ids = integer(0L), run.ids = integer(0L), verbosity = NULL,
  overwrite = FALSE)
```

Arguments

`data.ids`	[integer] Dataset IDs. Default is none.
`task.ids`	[integer] Task IDs. Default is none.
`flow.ids`	[integer] Flow IDs. Default is none.
`run.ids`	[integer] Run IDs. Default is none.
`verbosity`	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via `setOMLConfig`.
`overwrite`	[integer(1)] Should files that are already in cache be overwritten?

Value

`invisible(NULL)`

`runTaskFlow` *Reproduce the Run*

Description

Uses the ID of the run and tries to reproduce its results by downloading the flow and applying it to the respective task.

Usage

```
runTaskFlow(task, flow, par.list, seed = 1, predict.type = NULL,
  verbosity = NULL, models = TRUE)
```

Arguments

task	[OMLTask] An OpenML task.
flow	[OMLFlow] Flow that is applied to the Task.
par.list	[list|OMLRunParList] Can be either a named list containing the hyperparameter values or a OMLRunParList.
seed	[numeric(1)|OMLSeedParList] Set a seed to make the run reproducible. Default is 1 and sets the seed using set.seed(1).
predict.type	[character(1)] Optional. See setPredictType. Default is "response".
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.
models	[logical(1)] This argument is passed to benchmark. Should all fitted models be stored in the ResampleResult? Default is TRUE.

Value

OMLMlrRun , an OMLRun.

---

runTaskMlr *Run mlr learner on OpenML task.*

Description

Run task with a specified learner from **mlr** and produce predictions.

Usage

```
runTaskMlr(task, learner, measures = NULL, verbosity = NULL, seed = 1,
  scimark.vector = NULL, models = TRUE, ...)
```

Arguments

task	[OMLTask] An OpenML task.
learner	[Learner] Learner from package mlr to run the task.
measures	[Measure] Additional measures that should be computed.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.
seed	[numeric(1)|OMLSeedParList] Set a seed to make the run reproducible. Default is 1 and sets the seed using set.seed(1).
scimark.vector	[numeric(6)] Optional vector of performance measurements computed by the scientific SciMark benchmark. May be computed using the **rscimark** R package. Default is NULL, which means no performance measurements.
models	[logical(1)] This argument is passed to benchmark. Should all fitted models be stored in the ResampleResult? Default is TRUE.
...	[any] Further arguments that are passed to convertOMLTaskToMlr.

Value

list Named list with the following components:

run The OMLRun object.

bmr Benchmark result returned by benchmark.

flow The generated OMLFlow object.

See Also

getOMLTask, makeLearner

Examples

```
# \dontrun{
#   library(mlr)
#   ## run a single flow (learner) on a single task
#   task = getOMLTask(57)
#   lrn = makeLearner("classif.rpart")
#   res = runTaskMlr(task, lrn)
#   ## the result "res" is a list, storing information on the actual "run", the
```

```
#   ## corresponding benchmark result "bmr" and the applied "flow"
# }
```

---

saveOMLConfig *Saves a list of OpenML configuration settings to file.*

Description

The new configuration is automatically assigned via setOMLConfig if all checks pass. If you don't set a certain option, package defaults will be inserted into the file.

Usage

```
saveOMLConfig(server = NULL, verbosity = NULL, apikey = NULL,
  cachedir = NULL, arff.reader = NULL, confirm.upload = NULL,
  overwrite = FALSE)
```

Arguments

server	[character(1)] URL of the XML API endpoint.
verbosity	[integer(1)] Verbosity level. Possible values are 0 (normal output), 1 (info output), 2 (debug output).
apikey	[character(1)] Your OpenML API key. Log in to OpenML, move to your profile to get it.
cachedir	[character(1)] Path to the cache directory.
arff.reader	[character(1)] Name of the package which should be used to parse arff files. Possible are "RWeka", which is the default and "farff".
confirm.upload	[logical(1)] Should the user be asked for confirmation before upload of OML objects?
overwrite	[logical(1)] Should an existing file be overwritten? Default is FALSE.

See Also

Other config: configuration, getOMLConfig, loadOMLConfig, setOMLConfig

`setOMLConfig` *Settter for configuration settings.*

Description

Set and overwrite configuration settings.

Usage

```
setOMLConfig(server = NULL, verbosity = NULL, apikey = NULL,
  cachedir = NULL, arff.reader = NULL, confirm.upload = NULL)
```

Arguments

`server`	[`character(1)`] URL of the XML API endpoint.
`verbosity`	[`integer(1)`] Verbosity level. Possible values are 0 (normal output), 1 (info output), 2 (debug output).
`apikey`	[`character(1)`] Your OpenML API key. Log in to OpenML, move to your profile to get it.
`cachedir`	[`character(1)`] Path to the cache directory.
`arff.reader`	[`character(1)`] Name of the package which should be used to parse arff files. Possible are "RWeka", which is the default and "farff".
`confirm.upload`	[`logical(1)`] Should the user be asked for confirmation before upload of OML objects?

Value

Invisibly returns a list of configuration settings.

See Also

Other config: `configuration`, `getOMLConfig`, `loadOMLConfig`, `saveOMLConfig`

tagOMLObject *Tagging of OpenML objects*

Description

Add or remove a specific tag to a OpenML data, task, flow or run.

Usage

```
tagOMLObject(ids, object = c("data", "task", "flow", "run"), tags,
  verbosity = NULL)

untagOMLObject(ids, object = c("data", "task", "flow", "run"), tags,
  verbosity = NULL)
```

Arguments

ids	[integer] The IDs of the respective objects.
object	[character(1)] A character that specifies the object you want to delete from the server. Can be either "data", "task", "flow" or "run".
tags	[character] The tags that should be added/removed.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

See Also

Other data set-related functions: OMLDataSetDescription, OMLDataSet, convertMlrTaskToOMLDataSet, convertOMLDataSetToMlr, deleteOMLObject, getOMLDataSet, listOMLDataSets, uploadOMLDataSet

Other task-related functions: convertOMLTaskToMlr, deleteOMLObject, getOMLTask, listOMLTaskTypes, listOMLTasks, makeOMLTask

Other flow-related functions: convertOMLFlowToMlr, deleteOMLObject, getOMLFlow, listOMLFlows, makeOMLFlowParameter, makeOMLFlow

Other run-related functions: convertOMLMlrRunToBMR, convertOMLRunToBMR, deleteOMLObject, getOMLRun, listOMLRuns, makeOMLRunParameter, makeOMLRun, uploadOMLRun

`uploadOMLDataSet` *Upload a data set to the OpenML server.*

Description

Share a data set by uploading it to the OpenML server.

Usage

```
uploadOMLDataSet(x, tags = NULL, description = NULL,
  confirm.upload = NULL, verbosity = NULL)
```

Arguments

`x`	[Task|[OMLDataSet] Contains the data set that should be uploaded.
`tags`	[character] The tags that should be added after uploading.
`description`	[character(1)|OMLDataSetDescription] Either an OMLDataSetDescription or a character(1) that describes the data. For the latter, all other relevant information is autogenerated from the Task.
`confirm.upload`	[logical(1)] Should the user be asked to confirm the upload? Default is the setting from your config.
`verbosity`	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.

Value

`invisible(numeric(1))` . The ID of the data (`data.id`).

Note

This function will reset the cache of `listOMLDataSets` on success.

See Also

Other uploading functions: `uploadOMLFlow`, `uploadOMLRun`

Other data set-related functions: `OMLDataSetDescription`, `OMLDataSet`, `convertMlrTaskToOMLDataSet`, `convertOMLDataSetToMlr`, `deleteOMLObject`, `getOMLDataSet`, `listOMLDataSets`, `tagOMLObject`

uploadOMLFlow *Upload an OpenML.*

Description

Share a flow by uploading it to the OpenML server.

Usage

```
uploadOMLFlow(x, tags = NULL, verbosity = NULL, confirm.upload = NULL,
  sourcefile = NULL, binaryfile = NULL)
```

Arguments

x	[OMLFlow|Learner] The flow that should be uploaded.
tags	[character] The tags that should be added after uploading.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.
confirm.upload	[logical(1)] Should the user be asked to confirm the upload? Default is the setting from your config.
sourcefile	[character(1)] The file path to the flow (not needed for Learner).
binaryfile	[character(1)] The file path to the flow (not needed for Learner).

Value

invisible(numeric) . The ID of the flow (flow.id).

Note

This function will reset the cache of listOMLFlows on success.

See Also

Other uploading functions: uploadOMLDataSet, uploadOMLRun

uploadOMLRun *Upload an OpenML run.*

Description

Share a run of a flow on a given OpenML task by uploading it to the OpenML server.

Usage

```
uploadOMLRun(run, upload.bmr = FALSE, tags = NULL, confirm.upload = NULL,
  verbosity = NULL, ...)
```

Arguments

run	[OMLRun|OMLMlrRun] The run that should be uploaded. Either a OMLRun or a run created with OMLMlrRun.
upload.bmr	[logical(1)] Should the Benchmark result created by benchmark function be uploaded? If set to TRUE and the flow is created via makeTuneWrapper, an arff file that contains the hyperparameter optimization trace is also uploaded.
tags	[character] The tags that should be added after uploading.
confirm.upload	[logical(1)] Should the user be asked to confirm the upload? Default is the setting from your config.
verbosity	[integer(1)] Print verbose output on console? Possible values are: 0: normal output, 1: info output, 2: debug output. Default is set via setOMLConfig.
...	Not used.

Value

invisible(numeric(1)) . The run ID.

Note

This function will reset the cache of listOMLRuns and listOMLRunEvaluations on success.

By default you will be asked to confirm the upload. You can deactivate the need for confirmation by setting "confirm.upload = TRUE" via setOMLConfig or set the corresponding argument each time you call the function.

See Also

Other uploading functions: uploadOMLDataSet, uploadOMLFlow

Other run-related functions: convertOMLMlrRunToBMR, convertOMLRunToBMR, deleteOMLObject, getOMLRun, listOMLRuns, makeOMLRunParameter, makeOMLRun, tagOMLObject

Index