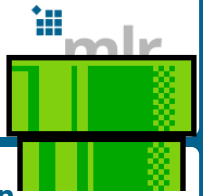
mlr3pipelines CHEAT SHEET

Organize machine learning operations into streams



Constructing PipeOps

`po(.obj)`
Shortcut to extracting existing PipeOp (it's preferred to creating your own).

`.obj`
If string, then returns PipeOp from mlr_pipeops$get(.obj); otherwise, PipeOpLearner or PipeOpFilter, if .obj is Learner or Filter respectably

Operators

`` `%>>%`(g1, g2) ``
Pipe operator that makes connection from a Graph or PipeOp to another.

`g1`
Graph or PipeOp with one output channel (or as many as g2 inputs)

`g2`
Graph or PipeOp with one input channel (or as many as g1 outputs)

class Graph

`$add_pipeop(op)`
Adds disconnected PipeOp to graph.

`op`
A PipeOp object to append

`$train(input)`
Traverses PipeOps within graph and calls $train() on each.

`input`
Task object, but it depends on PipeOps within graph

`$predict(input)`
Creates prediction on trained graph.

`input`
Task object, but it depends on PipeOps within graph

`$plot(html)`
Illustrates pipe flow in graph form.

`html`
Boolean indicating whether to use igraph or produce htmlWidget

Graph manipulation

`gunion(graphs)`
Creates Graph with disconnected graphs from list.

`graphs`
List of Graphs or PipeOps to join in one Graph

`greplicate(graph, n)`
Creates n disconnected copies of Graph or PipeOp.

`graph`
Graph or PipeOp to replicate

`n`
Number of copies to create

`branch(...)`
Creates Graph with multiple branches.

`...`
Graphs or PipeOps with exactly one output, each for its own branch

More at github.com/mlr-org/mlr3pipelines