

Package: dagml (via r-universe)

October 8, 2026

Title R Binding for DAG-ML Local Implementations

Version 0.3.41

Description Provides R bindings for native DAG-ML phase execution, host hyperparameter search, cross-validation, refit, prediction and bundle replay through process adapters. R loss and metric functions are retained in process-local registries keyed by validated implementation descriptors; executable functions are never serialized into graph contracts.

License CeCILL-2.1 | AGPL (>= 3)

URL <https://github.com/GBeurier/dag-ml>

BugReports <https://github.com/GBeurier/dag-ml/issues>

Depends R (>= 4.1)

Imports jsonlite

SystemRequirements DAG-ML C ABI shared library for task-bound loss invocation

Encoding UTF-8

NeedsCompilation yes

Repository <https://gbeurier.r-universe.dev>

Date/Publication 2026-10-08 01:22:38 UTC

RemoteUrl <https://github.com/GBeurier/dag-ml>

RemoteRef v0.3.41

RemoteSha 6f4044b45028a90a92d3f29287e67779bb5fd0b9

RemoteSubdir bindings/r

Contents

dagml_cv_refit_predict	2
dagml_execute_execution_plan_phase	3
dagml_host_hpo_search	4
dagml_initial_full_refit	5

dagml_initial_full_refit_predict	6
dagml_local_implementation_registry	7
dagml_replay_bundle	7
Index	9

dagml_cv_refit_predict

Run CV, REFIT and PREDICT in one native DAG-ML session

Description

The CLI retains host model handles across phases and returns a native bundle, OOF averages, replay prediction blocks and scores. This is a one-session replay; the bundle JSON does not include host model sidecars.

Usage

```
dagml_cv_refit_predict(dsl, controllers, envelope, adapter,  
cli = "dag-ml-cli", output = NULL, process_workers = 1L,  
process_timeout_ms = 30000L, process_retries = 0L,  
bundle_id = "bundle:cli.process.dsl.cv.refit.replay", variant_id = NULL,  
selection_metric = "rmse", selections = NULL,  
plan_id = "plan:cli.process.dsl.cv.refit.replay",  
run_id = "run:cli.process.dsl.cv.refit.replay", root_seed = 12345L,  
scheduler = "sequential", scheduler_workers = 1L, cpu_threads = 1L,  
gpu_devices = character())
```

Arguments

- dsl, controllers, envelope
JSON input file paths.
- adapter
Executable operator process adapter.
- cli
Path to dag-ml-cli or a command on PATH.
- output
Optional outcome JSON path; temporary by default.
- process_workers, process_timeout_ms
Positive process settings.
- process_retries
Non-negative retry count.
- bundle_id, plan_id, run_id
Native identifiers.
- variant_id
Optional concrete variant ID.
- selection_metric
One of rmse, accuracy, balanced_accuracy.
- selections
Optional selection-decision JSON path.

root_seed	Non-negative native seed.
scheduler	Either sequential or parallel.
scheduler_workers, cpu_threads	Positive resource counts.
gpu_devices	Optional GPU device identifiers.

Value

Native CV, REFIT and PREDICT outcome decoded as an R list.

dagml_execute_execution_plan_phase

Execute One ExecutionPlan Phase Through Native DAG-ML

Description

Runs one phase with DAG-ML's native sequential scheduler while keeping operator state and executable R callbacks local to the process. A controller callback can close over `dagml_local_implementation_registry()` and call `invoke_training_loss()` with the exact `task_json` it receives; the loss/metric registries remain separate and only loss callbacks can emit training-loss attestations.

Usage

```
dagml_execute_execution_plan_phase(
  execution_plan,
  trusted_controller_manifests,
  run_id,
  root_seed,
  phase,
  controllers,
  native_library = Sys.getenv("DAGML_NATIVE_LIBRARY", unset = "")
)
```

Arguments

execution_plan	ExecutionPlan JSON text, or a JSON-serializable R value.
trusted_controller_manifests	ControllerManifest list JSON text, or a JSON-serializable R value. The native core refuses execution if these manifests do not exactly match the manifests embedded in <code>execution_plan</code> .
run_id	DAG-ML run id for the phase execution.
root_seed	Non-negative safe integer root seed.
phase	ABI phase name such as FIT_CV or REFIT.
controllers	Named list or environment of callbacks keyed by <code>controller_id</code> . Each callback receives (<code>controller_id</code> , <code>task_json</code>) and returns <code>NodeResult</code> JSON text or a JSON-serializable list.

`native_library` Path to the DAG-ML C ABI shared library. Defaults to the DAGML_NATIVE_LIBRARY environment variable.

Value

A list of decoded NodeResult objects produced by the native scheduler.

`dagml_host_hpo_search` *Run scheduler-owned host hyperparameter search*

Description

Calls the DAG-ML CLI HPO scheduler. The native core owns trial execution, fold scoring, pruning, selection, and checkpoints; the host adapters own R operators and optimizer proposals.

Usage

```
dagml_host_hpo_search(plan, envelope, request, operator_adapter,
  optimizer_adapter, cli = "dag-ml-cli", checkpoint = NULL,
  output = NULL, operator_persistent = FALSE, parallel_trials = 1L,
  adapter_timeout_ms = 30000L,
  operator_timeout_ms = 0L)
```

Arguments

`plan, envelope, request`
Paths to DAG-ML JSON contracts.

`operator_adapter, optimizer_adapter`
Executable JSONL process adapters.

`cli`
Path to dag-ml-cli or a command on PATH.

`checkpoint`
Optional native checkpoint file for durable resume.

`output`
Optional result JSON file. Defaults to a temporary file.

`operator_persistent`
Keep a candidate-local operator process across folds.

`parallel_trials`
Positive number of concurrent candidate trials. DAG-ML starts isolated operator-adapter processes.

`adapter_timeout_ms`
Positive adapter response timeout in milliseconds.

`operator_timeout_ms`
Operator task timeout in milliseconds. Zero disables it. The optimizer timeout remains separate.

Value

Native HPO outcome decoded as an R list without simplifying JSON arrays.

dagml_initial_full_refit

Capture a no-CV full REFIT package through DAG-ML

Description

Runs the native no-splitter REFIT coordinator and captures its signed package. Host model bytes remain in host-managed sidecars.

Usage

```
dagml_initial_full_refit(dsl, controllers, envelope, training_sample_ids,
    adapter, package_output, cli = "dag-ml-cli", output = NULL,
    persistent = FALSE, process_workers = 1L, process_timeout_ms = 30000L,
    process_retries = 0L,
    package_id = "package:cli.process.dsl.initial.refit",
    plan_id = "plan:cli.process.dsl.refit.phase",
    run_id = "run:cli.process.dsl.refit.phase", root_seed = 12345L,
    scheduler = "sequential", scheduler_workers = 1L, cpu_threads = 1L,
    gpu_devices = character())
```

Arguments

dsl, controllers, envelope, training_sample_ids	JSON input file paths.
adapter	Executable operator process adapter.
package_output	Required destination for the signed package JSON.
cli	Path to dag-ml-cli or a command on PATH.
output	Optional outcome JSON path; temporary by default.
persistent	Keep the operator adapter alive across tasks.
process_workers	Positive number of adapter workers.
process_timeout_ms	Positive adapter timeout in milliseconds.
process_retries	Non-negative retry count.
package_id, plan_id, run_id	Native identifiers for capture lineage.
root_seed	Non-negative native seed.
scheduler	Either sequential or parallel.
scheduler_workers, cpu_threads	Positive resource counts.
gpu_devices	Optional GPU device identifiers.

Value

Native REFIT outcome decoded as an R list.

dagml_initial_full_refit_predict
<i>Replay PREDICT from a no-CV full REFIT package</i>

Description

Validates the signed package and V2 prediction cohort, then resolves exact host-sidecar artifact handles through the host adapter.

Usage

```
dagml_initial_full_refit_predict(package, envelope, adapter,
  artifact_handles, output_ids, cli = "dag-ml-cli", output = NULL,
  persistent = FALSE, process_workers = 1L, process_timeout_ms = 30000L,
  process_retries = 0L, run_id = "run:cli.initial.refit.predict")
```

Arguments

package, envelope, adapter, artifact_handles, output_ids	Package, JSON contract, or executable adapter paths required by native replay.
cli	Path to dag-ml-cli or a command on PATH.
output	Optional outcome JSON path; temporary by default.
persistent	Keep the operator adapter alive across tasks.
process_workers	Positive number of adapter workers.
process_timeout_ms	Positive adapter timeout in milliseconds.
process_retries	Non-negative retry count.
run_id	Native replay run identifier.

Value

Native PREDICT replay outcome decoded as an R list.

dagml_local_implementation_registry

Create a Process-Local R Implementation Registry

Description

Retains R loss and metric functions under exact DAG-ML implementation descriptors. The `invoke_training_loss()` method executes an active FIT_CV or REFIT loss from the exact native NodeTask JSON and returns the task attestation only after the function succeeds. Role selection and attestation validation are delegated to the native DAG-ML core; tensors and executable R functions remain in the R process and are never written to DAG-ML JSON contracts.

Usage

```
dagml_local_implementation_registry(  
  native_library = Sys.getenv("DAGML_NATIVE_LIBRARY", unset = "")  
)
```

Arguments

`native_library` Path to the DAG-ML C ABI shared library. Defaults to the DAGML_NATIVE_LIBRARY environment variable and is required only for task-bound training-loss invocation.

Value

An environment whose methods register, resolve, invoke, unregister, inspect, and clear process-local R functions.

dagml_replay_bundle *Replay a persisted CV bundle with host-owned model sidecars*

Description

DAG-ML validates the persisted CV bundle, exact host artifact handles and separate replay envelopes. The host owns model sidecar loading into its adapter's invocation-local handle namespace.

Usage

```
dagml_replay_bundle(graph, campaign, controllers, bundle, replay_request,  
  adapter, artifact_handles, envelopes, cli = "dag-ml-cli", output = NULL,  
  persistent = FALSE, process_workers = 1L,  
  process_timeout_ms = 30000L, process_retries = 0L,  
  prediction_cache_payload = NULL, prediction_cache_store = NULL,  
  score_output = NULL, plan_id = "plan:cli.bundle",  
  run_id = "run:cli.process.replay", root_seed = 12345L,  
  scheduler = "sequential", scheduler_workers = 1L)
```

Arguments

<code>graph, campaign, controllers, bundle, replay_request, adapter, artifact_handles</code>	Required JSON or executable adapter paths.
<code>envelopes</code>	Named character vector mapping replay data keys to V2 envelope JSON paths.
<code>cli</code>	Path to <code>dag-ml-cli</code> or a command on PATH.
<code>output</code>	Optional outcome JSON path; temporary by default.
<code>persistent</code>	Keep the adapter alive across tasks.
<code>process_workers, process_timeout_ms</code>	Positive process settings.
<code>process_retries</code>	Non-negative retry count.
<code>prediction_cache_payload, prediction_cache_store</code>	Optional native prediction cache sources for OOF-dependent replay.
<code>score_output</code>	Optional score JSON destination.
<code>plan_id, run_id</code>	Native identifiers.
<code>root_seed</code>	Non-negative native seed.
<code>scheduler</code>	Either sequential or parallel.
<code>scheduler_workers</code>	Positive scheduler worker count.

Value

Native replay outcome decoded as an R list.

Index

`dagml_cv_refit_predict`, [2](#)
`dagml_execute_execution_plan_phase`, [3](#)
`dagml_host_hpo_search`, [4](#)
`dagml_initial_full_refit`, [5](#)
`dagml_initial_full_refit_predict`, [6](#)
`dagml_local_implementation_registry`, [7](#)
`dagml_replay_bundle`, [7](#)