Package 'agua'

June 2, 2022

Title 'tidymodels' Integration with 'h2o'

Version 0.0.1

Description Create and evaluate models using 'tidymodels' and 'h2o' <https://h2o.ai/>. The package enables users to specify 'h2o' as an engine for several modeling methods.

License MIT + file LICENSE

URL https://agua.tidymodels.org/, https://github.com/tidymodels/agua

Depends parsnip

Imports dplyr, glue, h2o (>= 3.36.0.4), hardhat, purrr, rlang, stats, tibble, utils

Suggests covr, knitr, testthat (>= 3.0.0)

Config/Needs/website tidyverse/tidytemplate

Config/testthat/edition 3

Config/testthat/parallel false

Encoding UTF-8

RoxygenNote 7.2.0.9000

NeedsCompilation no

Author Max Kuhn [aut] (<https://orcid.org/0000-0003-2402-136X>), Qiushi Yan [aut, cre], Steven Pawley [aut], RStudio [cph]

Maintainer Qiushi Yan <qiushi.yann@gmail.com>

Repository CRAN

Date/Publication 2022-06-02 09:40:02 UTC

R topics documented:

as_h2o *Data conversion tools*

Description

Data conversion tools

Usage

```
as_h2o(df, destination_frame_prefix = "object")

## S3 method for class 'H2OFrame'
as_tibble(
  x,
  ...,
  .rows = NULL,
  .name_repair = c("check_unique", "unique", "universal", "minimal"),
  rownames = pkgconfig::get_config("tibble::rownames", NULL)
)
```

Arguments

df	A R data frame.
destination_frame_prefix	A character string to use as the base name.
x	An H2OFrame.
...	Unused, for extensibility.
.rows	The number of rows, useful to create a 0-column tibble or just as an additional check.
.name_repair	Treatment of problematic column names: • "minimal": No name repair or checks, beyond basic existence, • "unique": Make sure names are unique and not empty, • "check_unique": (default value), no name repair, but check they are unique, • "universal": Make the names unique and syntactic • a function: apply custom name repair (e.g., .name_repair = make.names for names in the style of base R). • A purrr-style anonymous function, see rlang::as_function() This argument is passed on as repair to vctrs::vec_as_names(). See there for more details on these terms and the strategies used to enforce them.
rownames	How to treat existing row names of a data frame or matrix: • NULL: remove row names. This is the default. • NA: keep row names. • A string: the name of a new column. Existing rownames are transferred into this column and the row.names attribute is deleted. Read more in rownames.

Value

A tibble or, for `as_h2o()`, a list with `data` (an H2OFrame) and `id` (the id on the h2o server).

Examples

```
# start with h2o::h2o.init()
if (h2o_running()) {
 cars2 <- as_h2o(mtcars)
 cars2
 class(cars2$data)

 cars0 <- as_tibble(cars2$data)
 cars0
}
```

`h2o_running` *Check if h2o cluster is initialized*

Description

Check if h2o cluster is initialized

Usage

```
h2o_running(verbose = FALSE)
```

Arguments

`verbose`	Print out the message if no cluster is available.

Value

A logical.

Examples

```
h2o_running()
h2o_running(verbose = TRUE)
```

h2o_train *Model wrappers for h2o*

Description

Basic model wrappers for h2o model functions that include data conversion, seed configuration, and so on.

Usage

```
h2o_train(x, y, model, ...)

h2o_train_rf(x, y, ntrees = 50, mtries = -1, min_rows = 1, ...)

h2o_train_xgboost(
  x,
  y,
  ntrees = 50,
  max_depth = 6,
  min_rows = 1,
  learn_rate = 0.3,
  sample_rate = 1,
  col_sample_rate = 1,
  min_split_improvement = 0,
  stopping_rounds = 0,
  ...
)

h2o_train_glm(x, y, lambda = NULL, alpha = NULL, ...)
```

Arguments

x	A data frame of predictors
y	A vector of outcomes.
model	A character string for the model. Current selections are "randomForest", "xgboost", and "glm". Use h2o::h2o.xgboost.available() to see if that model can be used on your OS/h2o server.
...	Other options to pass to the h2o model functions (e.g., h2o::h2o.randomForest()).
ntrees	Number of trees. Defaults to 50.
mtries	Number of variables randomly sampled as candidates at each split. If set to -1, defaults to sqrtp for classification and p/3 for regression (where p is the # of predictors Defaults to -1.
min_rows	Fewest allowed (weighted) observations in a leaf. Defaults to 1.
max_depth	Maximum tree depth (0 for unlimited). Defaults to 20.
learn_rate	(same as eta) Learning rate (from 0.0 to 1.0) Defaults to 0.3.

`sample_rate` Row sample rate per tree (from 0.0 to 1.0) Defaults to 0.632.

`col_sample_rate`
(same as colsample_bylevel) Column sample rate (from 0.0 to 1.0) Defaults to 1.

`min_split_improvement`
Minimum relative improvement in squared error reduction for a split to happen Defaults to 1e-05.

`stopping_rounds`
Early stopping based on convergence of stopping_metric. Stop if simple moving average of length k of the stopping_metric does not improve for k:=stopping_rounds scoring events (0 to disable) Defaults to 0.

`lambda` Regularization strength

`alpha` Distribution of regularization between the L1 (Lasso) and L2 (Ridge) penalties. A value of 1 for alpha represents Lasso regression, a value of 0 produces Ridge regression, and anything in between specifies the amount of mixing between the two. Default value of alpha is 0 when SOLVER = 'L-BFGS'; 0.5 otherwise.

Value

An h2o model object.

Examples

```
# start with h2o::h2o.init()

if (h2o_running()) {
  # ----------------------------------------------------------------------
  # Using the model wrappers:
  h2o_train_glm(mtcars[, -1], mtcars$mpg)

  # ----------------------------------------------------------------------
  # using parsnip:

  spec <-
    rand_forest(mtry = 3, trees = 1000) %>%
    set_engine("h2o") %>%
    set_mode("regression")

  set.seed(1)
  mod <- fit(spec, mpg ~ ., data = mtcars)
  mod

  predict(mod, head(mtcars))
}
```

Index