Package: orbital (via r-universe)

September 27, 2024

Title Predict with 'tidymodels' Workflows in Databases

Version 0.2.0

Description Turn 'tidymodels' workflows into objects containing the sufficient sequential equations to perform predictions. These smaller objects allow for low dependency prediction locally or directly in databases.

License MIT + file LICENSE

URL https://github.com/tidymodels/orbital

BugReports https://github.com/tidymodels/orbital/issues

Imports cli, dplyr, rlang

Suggests arrow, DBI, dbplyr, dtplyr, duckdb, embed, glue, hardhat, jsonlite, kknn, knitr, modeldata, parsnip, R6, recipes, rmarkdown, RSQLite, rstanarm, sparklyr, testthat (>= 3.0.0), themis, tidypredict, workflows

VignetteBuilder knitr

Config/Needs/website tidyverse/tidytemplate, rmarkdown

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Emil Hvitfeldt [aut, cre], Posit Software, PBC [cph, fnd]

Maintainer Emil Hvitfeldt <emil.hvitfeldt@posit.co>

Repository CRAN

Date/Publication 2024-07-28 04:20:02 UTC

Contents

`orbital` *Turn tidymodels objects into orbital objects*

Description

Fitted workflows, parsnip objects, and recipes objects can be turned into an orbital object that contain all the information needed to perform predictions.

Usage

```
orbital(x, ...)
```

Arguments

`x`	A fitted workflow, parsnip, or recipes object.
`...`	Not currently used.

Details

An orbital object contains all the information that is needed to perform predictions. This makes the objects substantially smaller than the original objects. The main downside with this object is that all the input checking has been removed, and it is thus up to the user to make sure the data is correct.

The printing of orbital objects reduce the number of significant digits for easy viewing, the can be changes by using the `digits` argument of `print()` like so `print(orbital_object, digits = 10)`. The printing likewise truncates each equation to fit on one line. This can be turned off using the `truncate` argument like so `print(orbital_object, truncate = FALSE)`.

Full list of supported models and recipes steps can be found here: `vignette("supported-models")`.

These objects will not be useful by themselves. They can be used to predict() with, or to generate code using functions such as `orbital_sql()` or `orbital_dt()`.

Value

An orbital object.

Examples

```
library(workflows)
library(recipes)
library(parsnip)

rec_spec <- recipe(mpg ~ ., data = mtcars) %>%
  step_normalize(all_numeric_predictors())

lm_spec <- linear_reg()

wf_spec <- workflow(rec_spec, lm_spec)

wf_fit <- fit(wf_spec, mtcars)

orbital(wf_fit)

# Also works on parsnip object by itself
fit(lm_spec, mpg ~ disp, data = mtcars) %>%
  orbital()

# And prepped recipes
prep(rec_spec) %>%
  orbital()
```

---

orbital_dt *Convert to data.table code*

Description

Returns data.table code that is equivilant to prediction code.

Usage

```
orbital_dt(x)
```

Arguments

x	An orbital object. This function requires dtplyr to be installed to run. The resulting code will likely need to be adopted to your use-case. Most likely by removing the initial `copy(data-name)` at the start.

Value

data.table code.

Examples

```
library(workflows)
library(recipes)
library(parsnip)

rec_spec <- recipe(mpg ~ ., data = mtcars) %>%
  step_normalize(all_numeric_predictors())

lm_spec <- linear_reg()

wf_spec <- workflow(rec_spec, lm_spec)

wf_fit <- fit(wf_spec, mtcars)

orbital_obj <- orbital(wf_fit)

orbital_dt(orbital_obj)
```

---

orbital_inline *Convert orbital objects to quosures*

Description

Use orbital object splicing function to apply orbital prediction in a quosure aware function such as `dplyr::mutate()`.

Usage

```
orbital_inline(x)
```

Arguments

x	An orbital object.

Details

This function is mostly going to be used for Dots Injection. This function is used internally in predict(), but is also exported for user flexibility. Should be used with `!!!` as seen in the examples.

Note should be taken that using this function modifies existing variables and creates new variables, unlike predict() which only returns predictions.

Value

a list of quosures.

Examples

```
library(workflows)
library(recipes)
library(parsnip)

rec_spec <- recipe(mpg ~ ., data = mtcars) %>%
  step_normalize(all_numeric_predictors())

lm_spec <- linear_reg()

wf_spec <- workflow(rec_spec, lm_spec)

wf_fit <- fit(wf_spec, mtcars)

orbital_obj <- orbital(wf_fit)

orbital_inline(orbital_obj)

library(dplyr)

mtcars %>%
  mutate(!!!orbital_inline(orbital_obj))
```

---

orbital_json_read *Read orbital json file*

Description

Reading an orbital object from disk

Usage

```
orbital_json_read(path)
```

Arguments

path	file on disk.

Details

This function is aware of the `version` field of the orbital object, and will read it in correctly, according to its specification.

Value

An orbital object.

See Also

orbital_json_write()

Examples

```
library(workflows)
library(recipes)
library(parsnip)

rec_spec <- recipe(mpg ~ ., data = mtcars) %>%
  step_normalize(all_numeric_predictors())

lm_spec <- linear_reg()

wf_spec <- workflow(rec_spec, lm_spec)

wf_fit <- fit(wf_spec, mtcars)

orbital_obj <- orbital(wf_fit)

tmp_file <- tempfile()

orbital_json_write(orbital_obj, tmp_file)

orbital_json_read(tmp_file)
```

---

orbital_json_write *Save orbital object as json file*

Description

Saving an orbital object to disk in a human and machine readable way.

Usage

```
orbital_json_write(x, path)
```

Arguments

x	An orbital object.
path	file on disk.

Details

The structure of the resulting JSON file allows for easy reading, both by orbital itself with orbital_json_read(), but potentially by other packages and langauges. The file is versioned by the version field to allow for changes why being backwards combatible with older file versions.

Value

Nothing.

See Also

orbital_json_read()

Examples

```
library(workflows)
library(recipes)
library(parsnip)

rec_spec <- recipe(mpg ~ ., data = mtcars) %>%
  step_normalize(all_numeric_predictors())

lm_spec <- linear_reg()

wf_spec <- workflow(rec_spec, lm_spec)

wf_fit <- fit(wf_spec, mtcars)

orbital_obj <- orbital(wf_fit)

tmp_file <- tempfile()

orbital_json_write(orbital_obj, tmp_file)

readLines(tmp_file)
```

---

orbital_r_fun *Turn orbital object into a R function*

Description

Returns a R file that contains a function that output predictions when applied to data frames.

Usage

```
orbital_r_fun(x, name = "orbital_predict", file)
```

Arguments

x	An orbital object.
name	Name of created function. Defaults to ‘"orbital_predict"‘.
file	A file name.

Details

The generated function is only expected to work on data frame objects. The generated function doesn't require the orbital package to be loaded. Depending on what models and steps are used, other packages such as dplyr will need to be loaded as well.

Value

Nothing.

Examples

```
library(workflows)
library(recipes)
library(parsnip)

rec_spec <- recipe(mpg ~ ., data = mtcars) %>%
  step_normalize(all_numeric_predictors())

lm_spec <- linear_reg()

wf_spec <- workflow(rec_spec, lm_spec)

wf_fit <- fit(wf_spec, mtcars)

orbital_obj <- orbital(wf_fit)

file_name <- tempfile()

orbital_r_fun(orbital_obj, file = file_name)

readLines(file_name)
```

---

orbital_sql *Convert to SQL code*

Description

Returns SQL code that is equivilant to prediction code.

Usage

```
orbital_sql(x, con)
```

Arguments

x	An orbital object.
con	A connection object.

Details

This function requires a database connection object, as the resulting code SQL code can differ depending on the type of database.

Value

SQL code.

Examples

```
library(workflows)
library(recipes)
library(parsnip)

rec_spec <- recipe(mpg ~ ., data = mtcars) %>%
  step_normalize(all_numeric_predictors())

lm_spec <- linear_reg()

wf_spec <- workflow(rec_spec, lm_spec)

wf_fit <- fit(wf_spec, mtcars)

orbital_obj <- orbital(wf_fit)

library(dbplyr)
con <- simulate_dbi()

orbital_sql(orbital_obj, con)
```

`predict.orbital_class` *Prediction using orbital objects*

Description

Running prediction on data frame of remote database table, without needing to load original packages used to fit model.

Usage

```
## S3 method for class 'orbital_class'
predict(object, new_data, ...)
```

Arguments

`object`	An orbital object.
`new_data`	A data frame or remote database table.
`...`	Not currently used.

Details

Using this function should give identical results to running `predict()` or `bake()` on the orginal object.

The prediction done will only return prediction colunms, a opposed to returning all modified functions as done with `orbital_inline()`.

Value

A modified data frame or remote database table.

Examples

```
library(workflows)
library(recipes)
library(parsnip)

rec_spec <- recipe(mpg ~ ., data = mtcars) %>%
  step_normalize(all_numeric_predictors())

lm_spec <- linear_reg()

wf_spec <- workflow(rec_spec, lm_spec)

wf_fit <- fit(wf_spec, mtcars)

orbital_obj <- orbital(wf_fit)

predict(orbital_obj, mtcars)
```

Index