Package 'tidyAML'

August 24, 2026

Title Automatic Machine Learning with 'tidymodels'

Version 0.0.8

Description The goal of this package will be to provide a simple interface for automatic machine learning that fits the 'tidymodels' framework. The intention is to work for regression and classification problems with a simple verb framework.

License MIT + file LICENSE

Encoding UTF-8

URL https://www.spsanderson.com/tidyAML/,
https://github.com/spsanderson/tidyAML

BugReports https://github.com/spsanderson/tidyAML/issues

Depends parsnip, R (>= 4.1.0)

Suggests knitr, rmarkdown, stats, tibble, stringr, utils, recipes, multilevelmod, rules, poissonreg, censored, baguette, bonsai, brulee, rstanarm, dbarts, kknn, ranger, randomForest, LiblineaR, flexsurv, gee, glmnet, discrim, kernlab, klaR, mda, sda, sparsediscrim

VignetteBuilder knitr

Imports rlang (>= 0.4.11), purrr (>= 0.3.5), dplyr (>= 1.0.10), rsample (>= 1.1.0), workflows (>= 1.1.2), forcats, workflowsets, tidyr, broom, ggplot2, magrittr, tune (>= 1.3.0)

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Steven Sanderson [aut, cre, cph] (ORCID: <https://orcid.org/0009-0006-7661-8247>)

Maintainer Steven Sanderson <spsanderson@gmail.com>

Repository CRAN

Date/Publication 2026-08-24 18:00:02 UTC

Contents

check_duplicate_rows . . . 2
core_packages . . . 3
create_model_spec . . . 4
create_splits . . . 5
create_workflow_set . . . 6
extract_model_spec . . . 8
extract_regression_residuals . . . 9
extract_tunable_params . . . 10
extract_wflw . . . 11
extract_wflw_fit . . . 12
extract_wflw_pred . . . 13
fast_classification . . . 14
fast_classification_parsnip_spec_tbl . . . 15
fast_regression . . . 16
fast_regression_parsnip_spec_tbl . . . 18
full_internal_make_wflw . . . 19
get_model . . . 20
install_deps . . . 21
internal_make_fitted_wflw . . . 22
internal_make_spec_tbl . . . 23
internal_make_wflw . . . 24
internal_make_wflw_gee_lin_reg . . . 25
internal_make_wflw_predictions . . . 26
internal_model_builders_classification . . . 28
internal_model_builders_regression . . . 30
internal_set_args_to_tune . . . 32
load_deps . . . 33
make_classification_base_tbl . . . 34
make_regression_base_tbl . . . 34
match_args . . . 35
plot_regression_predictions . . . 36
plot_regression_residuals . . . 37
quantile_normalize . . . 38

Index **40**

`check_duplicate_rows` *Check for Duplicate Rows in a Data Frame*

Description

This function checks for duplicate rows in a data frame.

Usage

```
check_duplicate_rows(.data)
```

Arguments

.data	A data frame.

Details

This function checks for duplicate rows by comparing each row in the data frame to every other row. If a row is identical to another row, it is considered a duplicate.

Value

A logical vector indicating whether each row is a duplicate or not.

Author(s)

Steven P. Sanderson II, MPH

See Also

duplicated, anyDuplicated

Other Utility: core_packages(), create_splits(), create_workflow_set(), fast_classification_parsnip_spec_t
fast_regression_parsnip_spec_tbl(), full_internal_make_wflw(), install_deps(), load_deps(),
match_args(), quantile_normalize()

Examples

```
data <- data.frame(
  x = c(1, 2, 3, 1),
  y = c(2, 3, 4, 2),
  z = c(3, 2, 5, 3)
)

check_duplicate_rows(data)
```

---

core_packages *Functions to Install all Core Libraries*

Description

Lists the core packages necessary to run all potential modeling algorithms.

Usage

```
core_packages()
```

Details

Lists the core packages necessary to run all potential modeling algorithms.

Value

A character vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), create_splits(), create_workflow_set(), fast_classification_parsnip, fast_regression_parsnip_spec_tbl(), full_internal_make_wflw(), install_deps(), load_deps(), match_args(), quantile_normalize()

Examples

```
core_packages()
```

---

create_model_spec *Generate Model Specification calls to* parsnip

---

Description

Creates a list/tibble of parsnip model specifications.

Usage

```
create_model_spec(
  .parsnip_eng = list("lm"),
  .mode = list("regression"),
  .parsnip_fns = list("linear_reg"),
  .return_tibble = TRUE
)
```

Arguments

.parsnip_eng	The input must be a list. The default for this is set to all. This means that all of the parsnip **linear regression engines** will be used, for example lm, or glm.
.mode	The input must be a list. The default is 'regression'
.parsnip_fns	The input must be a list. The default for this is set to all. This means that all of the parsnip **linear regression** functions will be used, for example linear_reg(), or cubist_rules.
.return_tibble	The default is TRUE. FALSE will return a list object.

Details

Creates a list/tibble of parsnip model specifications. With this function you can generate a list/tibble output of any model specification and engine you choose that is supported by the `parsnip` ecosystem.

Value

A list or a tibble.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Model_Generator: `fast_classification()`, `fast_regression()`

Examples

```
create_model_spec(
 .parsnip_eng = list("lm","glm","glmnet","cubist"),
 .parsnip_fns = list(
      "linear_reg","linear_reg","linear_reg",
      "cubist_rules"
     )
 )

create_model_spec(
 .parsnip_eng = list("lm","glm","glmnet","cubist"),
 .parsnip_fns = list(
      "linear_reg","linear_reg","linear_reg",
      "cubist_rules"
     ),
 .return_tibble = FALSE
 )
```

---

create_splits *Utility Create Splits Object*

Description

Create a splits object.

Usage

```
create_splits(.data, .split_type = "initial_split", .split_args = NULL)
```

Arguments

.data	The data being passed to make a split on
.split_type	The default is "initial_split", you can pass any other split type from the rsample library.
.split_args	The default is NULL in order to use the default split arguments. If you want to pass other arguments then must pass a list with the parameter name and the argument.

Details

Create a splits object that returns a list object of both the splits object itself and the splits type. This function supports all splits types from the rsample package.

Value

A list object

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), core_packages(), create_workflow_set(), fast_classification_parsnip
fast_regression_parsnip_spec_tbl(), full_internal_make_wflw(), install_deps(), load_deps(),
match_args(), quantile_normalize()

Examples

```
create_splits(mtcars, .split_type = "vfold_cv")
```

---

create_workflow_set *Create a Workflow Set Object*

Description

Create a workflow set object tibble from a model spec tibble.

Usage

```
create_workflow_set(.model_tbl = NULL, .recipe_list = list(), .cross = TRUE)
```

Arguments

.model_tbl	The model table that is generated from a function like fast_regression_parsnip_spec_tbl(). The model spec column will be grabbed automatically as the class of the object must be tidyaml_base_tbl
.recipe_list	Provide a list of recipes here that will get added to the workflow set object.
.cross	The default is TRUE, can be set to FALSE. This is passed to the cross parameter as an argument to the workflow_set() function.

Details

Create a workflow set object/tibble from a model spec tibble where the object class type is tidyaml_base_tbl. This function will take in a list of recipes and will grab the model specifications from the base tibble to create the workflow sets object. You can also supply the logical of TRUE/FALSe the .cross parameter which gets passed to the corresponding parameter as an argumnt to the workflowsets::workflow_set() function.

Value

A list object of workflows.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://workflowsets.tidymodels.org/

Other Utility: check_duplicate_rows(), core_packages(), create_splits(), fast_classification_parsnip_spec_
fast_regression_parsnip_spec_tbl(), full_internal_make_wflw(), install_deps(), load_deps(),
match_args(), quantile_normalize()

Examples

```
library(recipes)

rec_obj <- recipe(mpg ~ ., data = mtcars)
spec_tbl <- fast_regression_parsnip_spec_tbl(
  .parsnip_fns = "linear_reg",
  .parsnip_eng = c("lm","glm")
)

create_workflow_set(
  spec_tbl,
  list(rec_obj)
)
```

`extract_model_spec` *Extract A Model Specification*

Description

Extract a model specification from a tidyAML model tibble.

Usage

```
extract_model_spec(.data, .model_id = NULL)
```

Arguments

`.data`	The model table that must have the class `tidyaml_mod_spec_tbl`.
`.model_id`	The model number that you want to select, Must be an integer or sequence of integers, ie. `1` or `c(1,3,5)` or `1:2`

Details

This function allows you to get a model specification or more from a tibble with a class of "tidyaml_mod_spec_tbl". It allows you to select the model by the `.model_id` column. You can call the model id's by an integer or a sequence of integers.

Value

A tibble with the chosen model specification(s).

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Extractor: `extract_regression_residuals()`, `extract_tunable_params()`, `extract_wflw()`, `extract_wflw_fit()`, `extract_wflw_pred()`, `get_model()`

Examples

```
spec_tbl <- fast_regression_parsnip_spec_tbl(
  .parsnip_fns = "linear_reg",
  .parsnip_eng = c("lm","glm")
)

extract_model_spec(spec_tbl, 1)
extract_model_spec(spec_tbl, 1:2)
```

extract_regression_residuals *Extract Residuals from Fast Regression Models*

Description

This function extracts residuals from a fast regression model table (`fast_regression()`).

Usage

```
extract_regression_residuals(.model_tbl, .pivot_long = FALSE)
```

Arguments

`.model_tbl`	A fast regression model specification table (`fst_reg_spec_tbl`).
`.pivot_long`	A logical value indicating if the output should be pivoted. The default is `FALSE`.

Details

The function checks if the input model specification table inherits the class 'fst_reg_spec_tbl' and if it contains the column 'pred_wflw'. It then manipulates the data, grouping it by model, and extracts residuals for each model. The result is a list of data frames, each containing residuals, actual values, and predicted values for a specific model.

Value

The function returns a list of data frames, each containing residuals, actual values, and predicted values for a specific model.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Extractor: `extract_model_spec()`, `extract_tunable_params()`, `extract_wflw()`, `extract_wflw_fit()`, `extract_wflw_pred()`, `get_model()`

Examples

```
library(recipes, quietly = TRUE)

rec_obj <- recipe(mpg ~ ., data = mtcars)

fr_tbl <- fast_regression(mtcars, rec_obj, .parsnip_eng = c("lm","glm"),
.parsnip_fns = "linear_reg")

extract_regression_residuals(fr_tbl)
```

```
extract_regression_residuals(fr_tbl, .pivot_long = TRUE)
```

`extract_tunable_params` *Extract Tunable Parameters from Model Specifications*

Description

Extract a list of tunable parameters from the `.model_spec` column of a `tidyaml_mod_spec_tbl`.

Usage

```
extract_tunable_params(.model_tbl)
```

Arguments

`.model_tbl`	A model table with a class of `tidyaml_mod_spec_tbl`.

Details

This function iterates over the `.model_spec` column of a model table and extracts tunable parameters for each model using `tunable()`. The result is a list that can be further processed into a tibble if needed.

Value

A list of tibbles, each containing the tunable parameters for a model.

See Also

Other Extractor: `extract_model_spec()`, `extract_regression_residuals()`, `extract_wflw()`, `extract_wflw_fit()`, `extract_wflw_pred()`, `get_model()`

Examples

```
library(dplyr)
mods <- fast_regression_parsnip_spec_tbl(
  .parsnip_fns = "linear_reg",
  .parsnip_eng = c("lm","glmnet")
  )
extract_tunable_params(mods)
```

`extract_wflw` *Extract A Model Workflow*

Description

Extract a model workflow from a tidyAML model tibble.

Usage

```
extract_wflw(.data, .model_id = NULL)
```

Arguments

`.data`	The model table that must have the class `tidyaml_mod_spec_tbl`.
`.model_id`	The model number that you want to select, Must be an integer or sequence of integers, ie. `1` or `c(1,3,5)` or `1:2`

Details

This function allows you to get a model workflow or more from a tibble with a class of "tidyaml_mod_spec_tbl". It allows you to select the model by the `.model_id` column. You can call the model id's by an integer or a sequence of integers.

Value

A tibble with the chosen model workflow(s).

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Extractor: `extract_model_spec()`, `extract_regression_residuals()`, `extract_tunable_params()`, `extract_wflw_fit()`, `extract_wflw_pred()`, `get_model()`

Examples

```
library(recipes)

rec_obj <- recipe(mpg ~ ., data = mtcars)
frt_tbl <- fast_regression(mtcars, rec_obj, .parsnip_eng = c("lm","glm"),
                                            .parsnip_fns = "linear_reg")

extract_wflw(frt_tbl, 1)
extract_wflw(frt_tbl, 1:2)
```

`extract_wflw_fit` *Extract A Model Fitted Workflow*

Description

Extract a model fitted workflow from a tidyAML model tibble.

Usage

```
extract_wflw_fit(.data, .model_id = NULL)
```

Arguments

`.data`	The model table that must have the class `tidyaml_mod_spec_tbl`.
`.model_id`	The model number that you want to select, Must be an integer or sequence of integers, ie. `1` or `c(1,3,5)` or `1:2`

Details

This function allows you to get a model fitted workflow or more from a tibble with a class of "tidyaml_mod_spec_tbl". It allows you to select the model by the `.model_id` column. You can call the model id's by an integer or a sequence of integers.

Value

A tibble with the chosen model workflow(s).

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Extractor: `extract_model_spec()`, `extract_regression_residuals()`, `extract_tunable_params()`, `extract_wflw()`, `extract_wflw_pred()`, `get_model()`

Examples

```
library(recipes)

rec_obj <- recipe(mpg ~ ., data = mtcars)
frt_tbl <- fast_regression(mtcars, rec_obj, .parsnip_eng = c("lm","glm"),
                                            .parsnip_fns = "linear_reg")

extract_wflw_fit(frt_tbl, 1)
extract_wflw_fit(frt_tbl, 1:2)
```

`extract_wflw_pred` *Extract A Model Workflow Predictions*

Description

Extract a model workflow predictions from a tidyAML model tibble.

Usage

```
extract_wflw_pred(.data, .model_id = NULL)
```

Arguments

`.data`	The model table that must have the class `tidyaml_mod_spec_tbl`.
`.model_id`	The model number that you want to select, Must be an integer or sequence of integers, ie. `1` or `c(1,3,5)` or `1:2`

Details

This function allows you to get a model workflow predictions or more from a tibble with a class of "tidyaml_mod_spec_tbl". It allows you to select the model by the `.model_id` column. You can call the model id's by an integer or a sequence of integers.

Value

A tibble with the chosen model workflow(s).

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Extractor: `extract_model_spec()`, `extract_regression_residuals()`, `extract_tunable_params()`, `extract_wflw()`, `extract_wflw_fit()`, `get_model()`

Examples

```
library(recipes)

rec_obj <- recipe(mpg ~ ., data = mtcars)
frt_tbl <- fast_regression(mtcars, rec_obj, .parsnip_eng = c("lm","glm"),
                                            .parsnip_fns = "linear_reg")

extract_wflw_pred(frt_tbl, 1)
extract_wflw_pred(frt_tbl, 1:2)
```

`fast_classification` *Generate Model Specification calls to* `parsnip`

Description

Creates a list/tibble of parsnip model specifications.

Usage

```
fast_classification(
  .data,
  .rec_obj,
  .parsnip_fns = "all",
  .parsnip_eng = "all",
  .split_type = "initial_split",
  .split_args = NULL,
  .drop_na = TRUE
)
```

Arguments

`.data`	The data being passed to the function for the classification problem
`.rec_obj`	The recipe object being passed.
`.parsnip_fns`	The default is 'all' which will create all possible classification model specifications supported.
`.parsnip_eng`	the default is 'all' which will create all possible classification model specifications supported.
`.split_type`	The default is 'initial_split', you can pass any type of split supported by `rsample`
`.split_args`	The default is NULL, when NULL then the default parameters of the split type will be executed for the rsample split type.
`.drop_na`	The default is TRUE, which will drop all NA's from the data.

Details

With this function you can generate a tibble output of any classification model specification and it's fitted `workflow` object. Per recipes documentation explicitly with `step_string2factor()` it is encouraged to mutate your predictor into a factor before you create your recipe.

Value

A list or a tibble.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Model_Generator: `create_model_spec()`, `fast_regression()`

Examples

```
library(recipes)
library(dplyr)
library(tidyr)

df <- Titanic |>
 as_tibble() |>
 uncount(n) |>
 mutate(across(everything(), as.factor))

rec_obj <- recipe(Survived ~ ., data = df)

fct_tbl <- fast_classification(
  .data = df,
  .rec_obj = rec_obj,
  .parsnip_eng = c("glm","earth")
  )

fct_tbl
```

---

fast_classification_parsnip_spec_tbl

Utility Classification call to `parsnip`

Description

Creates a tibble of parsnip classification model specifications.

Usage

```
fast_classification_parsnip_spec_tbl(
  .parsnip_fns = "all",
  .parsnip_eng = "all"
)
```

Arguments

`.parsnip_fns`	The default for this is set to `all`. This means that all of the parsnip **classification** functions will be used, for example `bag_mars()`, or `bart()`. You can also choose to pass a c() vector like `c("barg_mars","bart")`
`.parsnip_eng`	The default for this is set to `all`. This means that all of the parsnip **classification engines** will be used, for example `earth`, or `dbarts`. You can also choose to pass a c() vector like `c('earth', 'dbarts')`

Details

Creates a tibble of parsnip classification model specifications. This will create a tibble of 32 different classification model specifications which can be filtered. The model specs are created first and then filtered out. This will only create models for **classification** problems. To find all of the supported models in this package you can visit https://www.tidymodels.org/find/parsnip/

Value

A tibble with an added class of 'fst_class_spec_tbl'

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), core_packages(), create_splits(), create_workflow_set(), fast_regression_parsnip_spec_tbl(), full_internal_make_wflw(), install_deps(), load_deps(), match_args(), quantile_normalize()

Examples

```
fast_classification_parsnip_spec_tbl(.parsnip_fns = "logistic_reg")
fast_classification_parsnip_spec_tbl(.parsnip_eng = c("earth","dbarts"))
```

---

fast_regression *Generate Model Specification calls to* parsnip

---

Description

Creates a list/tibble of parsnip model specifications.

Usage

```
fast_regression(
  .data,
  .rec_obj,
  .parsnip_fns = "all",
  .parsnip_eng = "all",
  .split_type = "initial_split",
  .split_args = NULL,
  .drop_na = TRUE
)
```

Arguments

.data	The data being passed to the function for the regression problem
.rec_obj	The recipe object being passed.
.parsnip_fns	The default is 'all' which will create all possible regression model specifications supported.
.parsnip_eng	the default is 'all' which will create all possible regression model specifications supported.
.split_type	The default is 'initial_split', you can pass any type of split supported by rsample
.split_args	The default is NULL, when NULL then the default parameters of the split type will be executed for the rsample split type.
.drop_na	The default is TRUE, which will drop all NA's from the data.

Details

With this function you can generate a tibble output of any regression model specification and it's fitted workflow object.

Value

A list or a tibble.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Model_Generator: create_model_spec(), fast_classification()

Examples

```
library(recipes, quietly = TRUE)

rec_obj <- recipe(mpg ~ ., data = mtcars)
frt_tbl <- fast_regression(
  mtcars,
  rec_obj,
  .parsnip_eng = c("lm","glm","gee"),
  .parsnip_fns = "linear_reg"
  )

frt_tbl
```

`fast_regression_parsnip_spec_tbl` *Utility Regression call to* `parsnip`

Description

Creates a tibble of parsnip regression model specifications.

Usage

```
fast_regression_parsnip_spec_tbl(.parsnip_fns = "all", .parsnip_eng = "all")
```

Arguments

`.parsnip_fns`	The default for this is set to `all`. This means that all of the parsnip **linear regression** functions will be used, for example `linear_reg()`, or `cubist_rules`. You can also choose to pass a c() vector like `c("linear_reg","cubist_rules")`
`.parsnip_eng`	The default for this is set to `all`. This means that all of the parsnip **linear regression engines** will be used, for example `lm`, or `glm`. You can also choose to pass a c() vector like `c('lm', 'glm')`

Details

Creates a tibble of parsnip regression model specifications. This will create a tibble of 46 different regression model specifications which can be filtered. The model specs are created first and then filtered out. This will only create models for **regression** problems. To find all of the supported models in this package you can visit https://www.tidymodels.org/find/parsnip/

Value

A tibble with an added class of 'fst_reg_spec_tbl'

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `core_packages()`, `create_splits()`, `create_workflow_set()`, `fast_classification_parsnip_spec_tbl()`, `full_internal_make_wflw()`, `install_deps()`, `load_deps()`, `match_args()`, `quantile_normalize()`

Examples

```
fast_regression_parsnip_spec_tbl(.parsnip_fns = "linear_reg")
fast_regression_parsnip_spec_tbl(.parsnip_eng = c("lm","glm"))
```

full_internal_make_wflw *Full Internal Workflow for Model and Recipe*

Description

This function creates a full internal workflow for a model and recipe combination.

Usage

```
full_internal_make_wflw(.model_tbl, .rec_obj)
```

Arguments

.model_tbl	A model specification table (tidyaml_mod_spec_tbl).
.rec_obj	A recipe object.

Details

The function checks if the input model specification table inherits the class 'tidyaml_mod_spec_tbl'. It then manipulates the input table, making adjustments for factors and creating a list of grouped models. For each model-recipe pair, it uses the appropriate internal function based on the model type to create a workflow object. The specific internal function is selected using a switch statement based on the class of the model.

Value

The function returns a workflow object for the first model-recipe pair based on the internal function selected.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), core_packages(), create_splits(), create_workflow_set(), fast_classification_parsnip_spec_tbl(), fast_regression_parsnip_spec_tbl(), install_deps(), load_deps(), match_args(), quantile_normalize()

Examples

```
library(dplyr)
library(recipes)

rec_obj <- recipe(mpg ~ ., data = mtcars)

mod_tbl <- make_regression_base_tbl()
```

```
mod_tbl <- mod_tbl |>
  filter(
    .parsnip_engine %in% c("lm", "glm") &
    .parsnip_fns == "linear_reg"
    )
class(mod_tbl) <- c("tidyaml_mod_spec_tbl", class(mod_tbl))
mod_spec_tbl <- internal_make_spec_tbl(mod_tbl)
result <- full_internal_make_wflw(mod_spec_tbl, rec_obj)
result
```

`get_model` *Get a Model*

Description

Get a model from a tidyAML model tibble.

Usage

```
get_model(.data, .model_id = NULL)
```

Arguments

`.data`	The model table that must have the class `tidyaml_mod_spec_tbl`.
`.model_id`	The model number that you want to select, Must be an integer or sequence of integers, ie. `1` or `c(1,3,5)` or `1:2`

Details

This function allows you to get a model or models from a tibble with a class of "tidyaml_mod_spec_tbl". It allows you to select the model by the `.model_id` column. You can call the model id's by an integer or a sequence of integers.

Value

A tibble with the chosen models.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Extractor: `extract_model_spec()`, `extract_regression_residuals()`, `extract_tunable_params()`, `extract_wflw()`, `extract_wflw_fit()`, `extract_wflw_pred()`

Examples

```
library(recipes)

rec_obj <- recipe(mpg ~ ., data = mtcars)
spec_tbl <- fast_regression_parsnip_spec_tbl(
  .parsnip_fns = "linear_reg",
  .parsnip_eng = c("lm","glm")
)

get_model(spec_tbl, 1)
get_model(spec_tbl, 1:2)
```

---

install_deps *Functions to Install all Core Libraries*

Description

Installs all dependencies in the `core_packages()` function.

Usage

```
install_deps()
```

Details

Installs all dependencies in the `core_packages()` function.

Value

No return value, called for side effects

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `core_packages()`, `create_splits()`, `create_workflow_set()`, `fast_classification_parsnip_spec_tbl()`, `fast_regression_parsnip_spec_tbl()`, `full_internal_make_wflw()`, `load_deps()`, `match_args()`, `quantile_normalize()`

Examples

```
## Not run:
  install_deps()

## End(Not run)
```

`internal_make_fitted_wflw` *Internals Safely Make a Fitted Workflow from Model Spec tibble*

Description

Safely Make a fitted workflow from a model spec tibble.

Usage

```
internal_make_fitted_wflw(.model_tbl, .splits_obj)
```

Arguments

`.model_tbl`	The model table that is generated from a function like `fast_regression_parsnip_spec_tbl()`, must have a class of "tidyaml_mod_spec_tbl". This is meant to be used after the function `internal_make_wflw()` has been run and the tibble has been saved.
`.splits_obj`	The splits object from the auto_ml function. It is internal to the `auto_ml_` function.

Details

Create a fitted `parnsip` model from a `workflow` object.

Value

A list object of workflows.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: `internal_make_spec_tbl()`, `internal_make_wflw()`, `internal_make_wflw_gee_lin_reg()`, `internal_make_wflw_predictions()`, `internal_model_builders_classification`, `internal_model_builders_regr`
`internal_set_args_to_tune()`, `make_classification_base_tbl()`, `make_regression_base_tbl()`

Examples

```
library(recipes, quietly = TRUE)

mod_spec_tbl <- fast_regression_parsnip_spec_tbl(
  .parsnip_eng = c("lm","glm"),
  .parsnip_fns = "linear_reg"
)

rec_obj <- recipe(mpg ~ ., data = mtcars)
```

```
splits_obj <- create_splits(mtcars, "initial_split")

mod_tbl <- mod_spec_tbl |>
  mutate(wflw = full_internal_make_wflw(mod_spec_tbl, rec_obj))

internal_make_fitted_wflw(mod_tbl, splits_obj)
```

`internal_make_spec_tbl` *Internals Make a Model Spec tibble*

Description

Make a Model Spec tibble.

Usage

```
internal_make_spec_tbl(.model_tbl)
```

Arguments

`.model_tbl`	This is the data that should be coming from inside of the regression/classification to parsnip spec functions.

Details

Make a Model Spec tibble.

Value

A model spec tbl.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: `internal_make_fitted_wflw()`, `internal_make_wflw()`, `internal_make_wflw_gee_lin_reg()`, `internal_make_wflw_predictions()`, `internal_model_builders_classification`, `internal_model_builders_regr`
`internal_set_args_to_tune()`, `make_classification_base_tbl()`, `make_regression_base_tbl()`

Examples

```
make_regression_base_tbl() |>
  internal_make_spec_tbl()

make_classification_base_tbl() |>
  internal_make_spec_tbl()
```

`internal_make_wflw` *Internals Safely Make Workflow from Model Spec tibble*

Description

Safely Make a workflow from a model spec tibble.

Usage

```
internal_make_wflw(.model_tbl, .rec_obj)
```

Arguments

`.model_tbl`	The model table that is generated from a function like `fast_regression_parsnip_spec_tbl()`, must have a class of "tidyaml_mod_spec_tbl".
`.rec_obj`	The recipe object that is going to be used to make the workflow object.

Details

Create a model specification tibble that has a `workflows::workflow()` list column.

Value

A list object of workflows.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: `internal_make_fitted_wflw()`, `internal_make_spec_tbl()`, `internal_make_wflw_gee_lin_reg()`, `internal_make_wflw_predictions()`, `internal_model_builders_classification`, `internal_model_builders_regr`, `internal_set_args_to_tune()`, `make_classification_base_tbl()`, `make_regression_base_tbl()`

Examples

```
library(recipes, quietly = TRUE)

mod_spec_tbl <- fast_regression_parsnip_spec_tbl(
  .parsnip_eng = c("lm","glm","gee"),
  .parsnip_fns = "linear_reg"
)

rec_obj <- recipe(mpg ~ ., data = mtcars)

internal_make_wflw(mod_spec_tbl, rec_obj)
```

---

internal_make_wflw_gee_lin_reg

Internals Safely Make Workflow for GEE Linear Regression

Description

Safely Make a workflow from a model spec tibble.

Usage

```
internal_make_wflw_gee_lin_reg(.model_tbl, .rec_obj)
```

Arguments

.model_tbl	The model table that is generated from a function like fast_regression_parsnip_spec_tbl(), must have a class of "tidyaml_mod_spec_tbl".
.rec_obj	The recipe object that is going to be used to make the workflow object.

Details

Create a model specification tibble that has a workflows::workflow() list column.

Value

A list object of workflows.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: internal_make_fitted_wflw(), internal_make_spec_tbl(), internal_make_wflw(), internal_make_wflw_predictions(), internal_model_builders_classification, internal_model_builders_regr internal_set_args_to_tune(), make_classification_base_tbl(), make_regression_base_tbl()

Examples

```
library(dplyr)
library(recipes)
library(multilevelmod)

mod_tbl <- make_regression_base_tbl()
mod_tbl <- mod_tbl |>
  filter(
  .parsnip_engine %in% c("gee") &
  .parsnip_fns == "linear_reg"
  )

class(mod_tbl) <- c("tidyaml_mod_spec_tbl", class(mod_tbl))
mod_spec_tbl <- internal_make_spec_tbl(mod_tbl)
rec_obj <- recipe(mpg ~ ., data = mtcars)

internal_make_wflw_gee_lin_reg(mod_spec_tbl, rec_obj)
```

---

internal_make_wflw_predictions

Internals Safely Make Predictions on a Fitted Workflow from Model Spec tibble

---

Description

Safely Make predictions on a fitted workflow from a model spec tibble.

Usage

```
internal_make_wflw_predictions(.model_tbl, .splits_obj)
```

Arguments

.model_tbl	The model table that is generated from a function like fast_regression_parsnip_spec_tbl(), must have a class of "tidyaml_mod_spec_tbl". This is meant to be used after the function internal_make_fitted_wflw() has been run and the tibble has been saved.
.splits_obj	The splits object from the auto_ml function. It is internal to the auto_ml_ function.

Details

Create predictions on a fitted parnsip model from a workflow object.

Value

A list object tibble of the outcome variable and it's values along with the testing and training predictions in a single tibble.

.data_category	.data_type	.value
actual	actual	21.0
actual	actual	21.0
actual	actual	22.8
...	...	...
predicted	training	21.0
...	...	...
predicted	training	21.0

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: internal_make_fitted_wflw(), internal_make_spec_tbl(), internal_make_wflw(), internal_make_wflw_gee_lin_reg(), internal_model_builders_classification, internal_model_builders_regr internal_set_args_to_tune(), make_classification_base_tbl(), make_regression_base_tbl()

Examples

```
library(recipes, quietly = TRUE)

mod_spec_tbl <- fast_regression_parsnip_spec_tbl(
  .parsnip_eng = c("lm","glm"),
  .parsnip_fns = "linear_reg"
)

rec_obj <- recipe(mpg ~ ., data = mtcars)
splits_obj <- create_splits(mtcars, "initial_split")

mod_tbl <- mod_spec_tbl |>
  mutate(wflw = full_internal_make_wflw(mod_spec_tbl, rec_obj))

mod_fitted_tbl <- mod_tbl |>
  mutate(fitted_wflw = internal_make_fitted_wflw(mod_tbl, splits_obj))

internal_make_wflw_predictions(mod_fitted_tbl, splits_obj)
```

`internal_model_builders_classification` *Internal Model Builders for Classification*

Description

Internal functions that build pre-configured parsnip model specifications for classification models. Each function returns a parsnip model specification with the mode and engine pre-set.

Usage

```
earth_classification_bag_mars()

earth_classification_mars()

earth_classification_discrim_flexible()

dbarts_classification_bart()

mass_classification_discrim_linear()

mda_classification_discrim_linear()

sda_classification_discrim_linear()

sparsediscrim_classification_discrim_linear()

mass_classification_discrim_quad()

sparsediscrim_classification_discrim_quad()

klar_classification_discrim_regularized()

mgcv_classification_gen_additive_mod()

brulee_classification_logistic_reg()

gee_classification_logistic_reg()

glm_classification_logistic_reg()

glmer_classification_logistic_reg()

glmnet_classification_logistic_reg()

liblinear_classification_logistic_reg()
```

`brulee_classification_mlp()`

`nnet_classification_mlp()`

`brulee_classification_multinom_reg()`

`glmnet_classification_multinom_reg()`

`nnet_classification_multinom_reg()`

`klar_classification_naive_bayes()`

`kknn_classification_nearest_neighbor()`

`xrf_classification_rule_fit()`

`kernlab_classification_svm_linear()`

`liblinear_classification_svm_linear()`

`kernlab_classification_svm_poly()`

`kernlab_classification_svm_rbf()`

`liquidsvm_classification_svm_rbf()`

Details

These functions are used internally by `internal_make_spec_tbl()` to create model specifications. Each function follows the naming pattern: `{engine}_{mode}_{parsnip_function}()`

Value

A parsnip model specification object

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: `internal_make_fitted_wflw()`, `internal_make_spec_tbl()`, `internal_make_wflw()`, `internal_make_wflw_gee_lin_reg()`, `internal_make_wflw_predictions()`, `internal_model_builders_regressio`
`internal_set_args_to_tune()`, `make_classification_base_tbl()`, `make_regression_base_tbl()`

`internal_model_builders_regression` *Internal Model Builders for Regression*

Description

Internal functions that build pre-configured parsnip model specifications for regression models. Each function returns a parsnip model specification with the mode and engine pre-set.

Usage

```
lm_regression_linear_reg()

brulee_regression_linear_reg()

gee_regression_linear_reg()

glm_regression_linear_reg()

glmer_regression_linear_reg()

glmnet_regression_linear_reg()

gls_regression_linear_reg()

lme_regression_linear_reg()

lmer_regression_linear_reg()

stan_regression_linear_reg()

stan_glmer_regression_linear_reg()

cubist_regression_cubist_rules()

glm_regression_poisson_reg()

gee_regression_poisson_reg()

glmer_regression_poisson_reg()

glmnet_regression_poisson_reg()

hurdle_regression_poisson_reg()

stan_regression_poisson_reg()
```

`stan_glmer_regression_poisson_reg()`

`zeroinfl_regression_poisson_reg()`

`earth_regression_bag_mars()`

`earth_regression_mars()`

`rpart_regression_bag_tree()`

`rpart_regression_decision_tree()`

`partykit_regression_decision_tree()`

`dbarts_regression_bart()`

`xgboost_regression_boost_tree()`

`lightgbm_regression_boost_tree()`

`mgcv_regression_gen_additive_mod()`

`nnet_regression_mlp()`

`brulee_regression_mlp()`

`kknn_regression_nearest_neighbor()`

`ranger_regression_rand_forest()`

`randomforest_regression_rand_forest()`

`xrf_regression_rule_fit()`

`liblinear_regression_svm_linear()`

`kernlab_regression_svm_linear()`

`kernlab_regression_svm_poly()`

`kernlab_regression_svm_rbf()`

Details

These functions are used internally by `internal_make_spec_tbl()` to create model specifications. Each function follows the naming pattern: `{engine}_{mode}_{parsnip_function}()`

Value

A parsnip model specification object

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: internal_make_fitted_wflw(), internal_make_spec_tbl(), internal_make_wflw(), internal_make_wflw_gee_lin_reg(), internal_make_wflw_predictions(), internal_model_builders_classifica internal_set_args_to_tune(), make_classification_base_tbl(), make_regression_base_tbl()

`internal_set_args_to_tune`

Internals Make a Tunable Model Specification

Description

Make a tuned model specification object.

Usage

```
internal_set_args_to_tune(.model_tbl)
```

Arguments

`.model_tbl`	The model table that is generated from a function like `fast_regression_parsnip_spec_tbl()`, must have a class of "tidyaml_mod_spec_tbl".

Details

This will take a model specification that is created from a function like `fast_regression_parsnip_spec_tbl()` and update the **model_spec** `args` to `tune::tune()`. This is done dynamically, meaning you do not need to know the names of the parameters inside of the model specification.

Value

A list object of workflows.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: internal_make_fitted_wflw(), internal_make_spec_tbl(), internal_make_wflw(), internal_make_wflw_gee_lin_reg(), internal_make_wflw_predictions(), internal_model_builders_classifica internal_model_builders_regression, make_classification_base_tbl(), make_regression_base_tbl()

Examples

```
library(dplyr)

mod_tbl <- fast_regression_parsnip_spec_tbl()
mod_tbl$model_spec[[1]]

updated_mod_tbl <- mod_tbl |>
  mutate(model_spec = internal_set_args_to_tune(mod_tbl))
updated_mod_tbl$model_spec[[1]]
```

---

load_deps *Functions to Install all Core Libraries*

Description

Load all the core packages necessary to run all potential modeling algorithms.

Usage

```
load_deps()
```

Details

Load all the core packages necessary to run all potential modeling algorithms.

Value

No return value, called for side effects

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), core_packages(), create_splits(), create_workflow_set(), fast_classification_parsnip_spec_tbl(), fast_regression_parsnip_spec_tbl(), full_internal_make_wflw(), install_deps(), match_args(), quantile_normalize()

Examples

```
## Not run:
load_deps()

## End(Not run)
```

make_classification_base_tbl

Internals Make Base Classification Tibble

Description

Creates a base tibble to create parsnip classification model specifications.

Usage

```
make_classification_base_tbl()
```

Details

Creates a base tibble to create parsnip classification model specifications.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: `internal_make_fitted_wflw()`, `internal_make_spec_tbl()`, `internal_make_wflw()`, `internal_make_wflw_gee_lin_reg()`, `internal_make_wflw_predictions()`, `internal_model_builders_classifica`
`internal_model_builders_regression`, `internal_set_args_to_tune()`, `make_regression_base_tbl()`

Examples

```
make_classification_base_tbl()
```

make_regression_base_tbl

Internals Make Base Regression Tibble

Description

Creates a base tibble to create parsnip regression model specifications.

Usage

```
make_regression_base_tbl()
```

Details

Creates a base tibble to create parsnip regression model specifications.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Internals: `internal_make_fitted_wflw()`, `internal_make_spec_tbl()`, `internal_make_wflw()`, `internal_make_wflw_gee_lin_reg()`, `internal_make_wflw_predictions()`, `internal_model_builders_classifica` `internal_model_builders_regression`, `internal_set_args_to_tune()`, `make_classification_base_tbl()`

Examples

```
make_regression_base_tbl()
```

---

match_args *Match function arguments*

Description

Match a functions arguments.

Usage

```
match_args(f, args)
```

Arguments

`f`	The parsnip function such as `"linear_reg"` as a string and without the parentheses.
`args`	The arguments you want to supply to `f`

Details

Match a functions arguments, the bad ones passed will be rejected but the remaining passing ones will be returned.

Value

A list of matched arguments.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), core_packages(), create_splits(), create_workflow_set(), fast_classification_parsnip_spec_tbl(), fast_regression_parsnip_spec_tbl(), full_internal_make_wflw(), install_deps(), load_deps(), quantile_normalize()

Examples

```
match_args(
  f = "linear_reg",
  args = list(
    mode = "regression",
    engine = "lm",
    trees = 1,
    mtry = 1
   )
 )
```

---

plot_regression_predictions

Create ggplot2 plot of regression predictions

Description

Create a ggplot2 plot of regression predictions.

Usage

```
plot_regression_predictions(.data, .output = "list")
```

Arguments

.data	The data from the output of the extract_wflw_pred() function.
.output	The default is "list" which will return a list of plots. The other option is "facet" which will return a single faceted plot.

Details

Create a ggplot2 plot of regression predictions, the actual, training, and testing values. The output of this function can either be a list of plots or a single faceted plot. This function takes the output of the function extract_wflw_pred() function.

Value

A list of ggplot2 plots or a faceted plot.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Plotting: `plot_regression_residuals()`

Examples

```
library(recipes)

rec_obj <- recipe(mpg ~ ., data = mtcars)
frt_tbl <- fast_regression(
  mtcars,
  rec_obj,
  .parsnip_eng = c("lm","glm"),
  .parsnip_fns = "linear_reg"
  )

extract_wflw_pred(frt_tbl,1) |> plot_regression_predictions()
extract_wflw_pred(frt_tbl,1:nrow(frt_tbl)) |>
  plot_regression_predictions(.output = "facet")
```

---

plot_regression_residuals

Create ggplot2 plot of regression residuals

Description

Create a ggplot2 plot of regression residuals.

Usage

```
plot_regression_residuals(.data)
```

Arguments

`.data`	The data from the output of the `extract_regression_residuals()` function.

Details

Create a ggplot2 plot of regression residuals. The output of this function can either be a list of plots or a single faceted plot. This function takes the output of the `extract_regression_residuals()` function.

Value

A list of ggplot2 plots or a faceted plot.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Plotting: plot_regression_predictions()

Examples

```
library(recipes)

rec_obj <- recipe(mpg ~ ., data = mtcars)
frt_tbl <- fast_regression(
  mtcars,
  rec_obj,
  .parsnip_eng = c("lm","glm"),
  .parsnip_fns = "linear_reg"
  )

extract_regression_residuals(frt_tbl, FALSE)[1] |> plot_regression_residuals()
extract_regression_residuals(frt_tbl, TRUE)[1] |> plot_regression_residuals()
```

---

quantile_normalize *Perform quantile normalization on a numeric matrix/data.frame*

Description

This function will perform quantile normalization on two or more distributions of equal length. Quantile normalization is a technique used to make the distribution of values across different samples more similar. It ensures that the distributions of values for each sample have the same quantiles. This function takes a numeric matrix as input and returns a quantile-normalized matrix.

Usage

```
quantile_normalize(.data, .return_tibble = FALSE)
```

Arguments

.data	A numeric matrix where each column represents a sample.
.return_tibble	A logical value that determines if the output should be a tibble. Default is 'FALSE'.

Details

This function performs quantile normalization on a numeric matrix by following these steps:

1. Sort each column of the input matrix.
2. Calculate the mean of each row across the sorted columns.
3. Replace each column's sorted values with the row means.
4. Unsort the columns to their original order.

Value

A list object that has the following:

1. A numeric matrix that has been quantile normalized.
2. The row means of the quantile normalized matrix.
3. The sorted data
4. The ranked indices

Author(s)

Steven P. Sanderson II, MPH

See Also

`rowMeans`: Calculate row means.

`apply`: Apply a function over the margins of an array.

`order`: Order the elements of a vector.

Other Utility: `check_duplicate_rows()`, `core_packages()`, `create_splits()`, `create_workflow_set()`, `fast_classification_parsnip_spec_tbl()`, `fast_regression_parsnip_spec_tbl()`, `full_internal_make_wflw()`, `install_deps()`, `load_deps()`, `match_args()`

Examples

```
# Create a sample numeric matrix
data <- matrix(rnorm(20), ncol = 4)

# Perform quantile normalization
normalized_data <- quantile_normalize(data)
normalized_data

as.data.frame(normalized_data$normalized_data) |>
  sapply(function(x) quantile(x, probs = seq(0, 1, 1 / 4)))

quantile_normalize(data, .return_tibble = TRUE)
```

Index

∗ **Extractor**
extract_model_spec, 8
extract_regression_residuals, 9
extract_tunable_params, 10
extract_wflw, 11
extract_wflw_fit, 12
extract_wflw_pred, 13
get_model, 20
∗ **Internals**
internal_make_fitted_wflw, 22
internal_make_spec_tbl, 23
internal_make_wflw, 24
internal_make_wflw_gee_lin_reg, 25
internal_make_wflw_predictions, 26
internal_model_builders_classification, 28
internal_model_builders_regression, 30
internal_set_args_to_tune, 32
make_classification_base_tbl, 34
make_regression_base_tbl, 34
∗ **Model_Generator**
create_model_spec, 4
fast_classification, 14
fast_regression, 16
∗ **Plotting**
plot_regression_predictions, 36
plot_regression_residuals, 37
∗ **Utility**
check_duplicate_rows, 2
core_packages, 3
create_splits, 5
create_workflow_set, 6
fast_classification_parsnip_spec_tbl, 15
fast_regression_parsnip_spec_tbl, 18
full_internal_make_wflw, 19
install_deps, 21
load_deps, 33
match_args, 35
quantile_normalize, 38

anyDuplicated, *3*
apply, *39*

brulee_classification_logistic_reg (internal_model_builders_classification), 28
brulee_classification_mlp (internal_model_builders_classification), 28
brulee_classification_multinom_reg (internal_model_builders_classification), 28
brulee_regression_linear_reg (internal_model_builders_regression), 30
brulee_regression_mlp (internal_model_builders_regression), 30

check_duplicate_rows, 2
check_duplicate_rows(), *4*, *6*, *7*, *16*, *18*, *19*, *21*, *33*, *36*, *39*
core_packages, 3
core_packages(), *3*, *6*, *7*, *16*, *18*, *19*, *21*, *33*, *36*, *39*
create_model_spec, 4
create_model_spec(), *15*, *17*
create_splits, 5
create_splits(), *3*, *4*, *7*, *16*, *18*, *19*, *21*, *33*, *36*, *39*
create_workflow_set, 6
create_workflow_set(), *3*, *4*, *6*, *16*, *18*, *19*, *21*, *33*, *36*, *39*
cubist_regression_cubist_rules (internal_model_builders_regression), 30

dbarts_classification_bart
(internal_model_builders_classification), 28
dbarts_regression_bart
(internal_model_builders_regression), 30
duplicated, 3

earth_classification_bag_mars
(internal_model_builders_classification), 28
earth_classification_discrim_flexible
(internal_model_builders_classification), 28
earth_classification_mars
(internal_model_builders_classification), 28
earth_regression_bag_mars
(internal_model_builders_regression), 30
earth_regression_mars
(internal_model_builders_regression), 30
extract_model_spec, 8
extract_model_spec(), 9–13, 20
extract_regression_residuals, 9
extract_regression_residuals(), 8, 10–13, 20
extract_tunable_params, 10
extract_tunable_params(), 8, 9, 11–13, 20
extract_wflw, 11
extract_wflw(), 8–10, 12, 13, 20
extract_wflw_fit, 12
extract_wflw_fit(), 8–11, 13, 20
extract_wflw_pred, 13
extract_wflw_pred(), 8–12, 20

fast_classification, 14
fast_classification(), 5, 17
fast_classification_parsnip_spec_tbl, 15
fast_classification_parsnip_spec_tbl(), 3, 4, 6, 7, 18, 19, 21, 33, 36, 39
fast_regression, 16
fast_regression(), 5, 15
fast_regression_parsnip_spec_tbl, 18
fast_regression_parsnip_spec_tbl(), 3, 4, 6, 7, 16, 19, 21, 32, 33, 36, 39
full_internal_make_wflw, 19
full_internal_make_wflw(), 3, 4, 6, 7, 16, 18, 21, 33, 36, 39

gee_classification_logistic_reg
(internal_model_builders_classification), 28
gee_regression_linear_reg
(internal_model_builders_regression), 30
gee_regression_poisson_reg
(internal_model_builders_regression), 30
get_model, 20
get_model(), 8–13
glm_classification_logistic_reg
(internal_model_builders_classification), 28
glm_regression_linear_reg
(internal_model_builders_regression), 30
glm_regression_poisson_reg
(internal_model_builders_regression), 30
glmer_classification_logistic_reg
(internal_model_builders_classification), 28
glmer_regression_linear_reg
(internal_model_builders_regression), 30
glmer_regression_poisson_reg
(internal_model_builders_regression), 30
glmnet_classification_logistic_reg
(internal_model_builders_classification), 28
glmnet_classification_multinom_reg
(internal_model_builders_classification), 28
glmnet_regression_linear_reg
(internal_model_builders_regression), 30
glmnet_regression_poisson_reg
(internal_model_builders_regression), 30
gls_regression_linear_reg
(internal_model_builders_regression), 30

hurdle_regression_poisson_reg

(internal_model_builders_regression), 30

install_deps, 21
install_deps(), 3, 4, 6, 7, 16, 18, 19, 33, 36, 39
internal_make_fitted_wflw, 22
internal_make_fitted_wflw(), 23–25, 27, 29, 32, 34, 35
internal_make_spec_tbl, 23
internal_make_spec_tbl(), 22, 24, 25, 27, 29, 32, 34, 35
internal_make_wflw, 24
internal_make_wflw(), 22, 23, 25, 27, 29, 32, 34, 35
internal_make_wflw_gee_lin_reg, 25
internal_make_wflw_gee_lin_reg(), 22–24, 27, 29, 32, 34, 35
internal_make_wflw_predictions, 26
internal_make_wflw_predictions(), 22–25, 29, 32, 34, 35
internal_model_builders_classification, 22–25, 27, 28, 32, 34, 35
internal_model_builders_regression, 22–25, 27, 29, 30, 32, 34, 35
internal_set_args_to_tune, 32
internal_set_args_to_tune(), 22–25, 27, 29, 32, 34, 35

kernlab_classification_svm_linear (internal_model_builders_classification), 28
kernlab_classification_svm_poly (internal_model_builders_classification), 28
kernlab_classification_svm_rbf (internal_model_builders_classification), 28
kernlab_regression_svm_linear (internal_model_builders_regression), 30
kernlab_regression_svm_poly (internal_model_builders_regression), 30
kernlab_regression_svm_rbf (internal_model_builders_regression), 30
kknn_classification_nearest_neighbor (internal_model_builders_classification), 28
kknn_regression_nearest_neighbor (internal_model_builders_regression), 30
klar_classification_discrim_regularized (internal_model_builders_classification), 28
klar_classification_naive_bayes (internal_model_builders_classification), 28

liblinear_classification_logistic_reg (internal_model_builders_classification), 28
liblinear_classification_svm_linear (internal_model_builders_classification), 28
liblinear_regression_svm_linear (internal_model_builders_regression), 30
lightgbm_regression_boost_tree (internal_model_builders_regression), 30
liquidsvm_classification_svm_rbf (internal_model_builders_classification), 28
lm_regression_linear_reg (internal_model_builders_regression), 30
lme_regression_linear_reg (internal_model_builders_regression), 30
lmer_regression_linear_reg (internal_model_builders_regression), 30
load_deps, 33
load_deps(), 3, 4, 6, 7, 16, 18, 19, 21, 36, 39

make_classification_base_tbl, 34
make_classification_base_tbl(), 22–25, 27, 29, 32, 35
make_regression_base_tbl, 34
make_regression_base_tbl(), 22–25, 27, 29, 32, 34
mass_classification_discrim_linear (internal_model_builders_classification), 28
mass_classification_discrim_quad (internal_model_builders_classification),

28
match_args, 35
match_args(), *3, 4*, *6, 7*, *16*, *18, 19*, *21*, *33*, *39*
mda_classification_discrim_linear
(internal_model_builders_classification), 28
mgcv_classification_gen_additive_mod
(internal_model_builders_classification), 28
mgcv_regression_gen_additive_mod
(internal_model_builders_regression), 30

nnet_classification_mlp
(internal_model_builders_classification), 28
nnet_classification_multinom_reg
(internal_model_builders_classification), 28
nnet_regression_mlp
(internal_model_builders_regression), 30

order, *39*

partykit_regression_decision_tree
(internal_model_builders_regression), 30
plot_regression_predictions, 36
plot_regression_predictions(), *38*
plot_regression_residuals, 37
plot_regression_residuals(), *37*

quantile_normalize, 38
quantile_normalize(), *3, 4*, *6, 7*, *16*, *18, 19*, *21*, *33*, *36*

randomforest_regression_rand_forest
(internal_model_builders_regression), 30
ranger_regression_rand_forest
(internal_model_builders_regression), 30
rowMeans, *39*
rpart_regression_bag_tree
(internal_model_builders_regression), 30
rpart_regression_decision_tree
(internal_model_builders_regression), 30

sda_classification_discrim_linear
(internal_model_builders_classification), 28
sparsediscrim_classification_discrim_linear
(internal_model_builders_classification), 28
sparsediscrim_classification_discrim_quad
(internal_model_builders_classification), 28
stan_glmer_regression_linear_reg
(internal_model_builders_regression), 30
stan_glmer_regression_poisson_reg
(internal_model_builders_regression), 30
stan_regression_linear_reg
(internal_model_builders_regression), 30
stan_regression_poisson_reg
(internal_model_builders_regression), 30

workflows::workflow(), *24, 25*
workflowsets::workflow_set(), *7*

xgboost_regression_boost_tree
(internal_model_builders_regression), 30
xrf_classification_rule_fit
(internal_model_builders_classification), 28
xrf_regression_rule_fit
(internal_model_builders_regression), 30

zeroinfl_regression_poisson_reg
(internal_model_builders_regression), 30