

# Package ‘tidyAML’

November 29, 2023

**Title** Automatic Machine Learning with 'tidymodels'

**Version** 0.0.3

**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

**RoxygenNote** 7.2.3

**URL** <https://github.com/spsanderson/tidyAML>

**BugReports** <https://github.com/spsanderson/tidyAML/issues>

**Depends** parsnip, R (>= 4.1.0)

**Suggests** knitr, rmarkdown, roxygen2, stats, tibble, stringr, utils, recipes, multilevelmod, rules, poissonreg, censored, baguette, bonsai, brulee, rstanarm, dbarts, kknn, ranger, randomForest, LiblineaR, flexsurv, gee, glmnet

**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

**NeedsCompilation** no

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

**Maintainer** Steven Sanderson <spsanderson@gmail.com>

**Repository** CRAN

**Date/Publication** 2023-11-29 15:50:07 UTC

## R topics documented:

core_packages . . . . .	2
create_model_spec . . . . .	3
create_splits . . . . .	4

create_workflow_set . . . . .	5
extract_model_spec . . . . .	6
extract_wflw . . . . .	7
extract_wflw_fit . . . . .	8
extract_wflw_pred . . . . .	9
fast_classification . . . . .	10
fast_classification_parsnip_spec_tbl . . . . .	12
fast_regression . . . . .	13
fast_regression_parsnip_spec_tbl . . . . .	14
full_internal_make_wflw . . . . .	15
get_model . . . . .	16
install_deps . . . . .	17
internal_make_fitted_wflw . . . . .	18
internal_make_spec_tbl . . . . .	19
internal_make_wflw . . . . .	20
internal_make_wflw_gee_lin_reg . . . . .	21
internal_make_wflw_predictions . . . . .	22
internal_set_args_to_tune . . . . .	23
load_deps . . . . .	24
make_classification_base_tbl . . . . .	25
make_regression_base_tbl . . . . .	26
match_args . . . . .	26
<b>Index</b>	<b>28</b>

---

core_packages	<i>Functions to Install all Core Libraries</i>
---------------	--

---

**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: [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

```
core_packages()
```

---

create_model_spec	<i>Generate Model Specification calls to parsnip</i>
-------------------	--

---

### 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**

<code>.data</code>	The data being passed to make a split on
<code>.split_type</code>	The default is "initial_split", you can pass any other split type from the <code>rsample</code> library.
<code>.split_args</code>	The default is <code>NULL</code> 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: [core\\_packages\(\)](#), [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_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**

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

## Details

Create a workflow set object/tibble from a model spec tibble where the object class type is `tidyml_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 argument 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: `core_packages()`, `create_splits()`, `fast_classification_parsnip_spec_tbl()`, `fast_regression_parsnip_spec_tbl()`, `full_internal_make_wflw()`, `install_deps()`, `load_deps()`, `match_args()`

## 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 `tidyml_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 `"tidyml_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\\_wflw\\_fit\(\)](#), [extract\\_wflw\\_pred\(\)](#), [extract\\_wflw\(\)](#), [get\\_model\(\)](#)

**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")
)

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

---

`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\\_wflw\\_fit\(\)](#), [extract\\_wflw\\_pred\(\)](#), [get\\_model\(\)](#)

**Examples**

```
library(recipes)

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

extract_wflw(firt_tbl, 1)
extract_wflw(firt_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\\_wflw\\_pred\(\)](#), [extract\\_wflw\(\)](#), [get\\_model\(\)](#)

## Examples

```
library(recipes)

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

extract_wflw_fit(firt_tbl, 1)
extract_wflw_fit(firt_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**

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

**Details**

This function allows you to get a model workflow predictions or more from a tibble with a class of "tidyml\_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\\_wflw\\_fit\(\)](#), [extract\\_wflw\(\)](#), [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  
)
```

**Arguments**

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

**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, quietly = TRUE)  
library(dplyr, quietly = TRUE)  
  
df <- mtcars |> mutate(cyl = as.factor(cyl))  
rec_obj <- recipe(cyl ~ ., data = df)
```

```
fct_tbl <- fast_classification(  
  .data = df,  
  .rec_obj = rec_obj,  
  .parsnip_eng = c("glm", "LiblineaR"))  
  
glimpse(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: [core\\_packages\(\)](#), [create\\_splits\(\)](#), [create\\_workflow\\_set\(\)](#), [fast\\_regression\\_parsnip\\_spec\\_tbl\(\)](#), [full\\_internal\\_make\\_wflw\(\)](#), [install\\_deps\(\)](#), [load\\_deps\(\)](#), [match\\_args\(\)](#)

**Examples**

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

---

fast_regression	<i>Generate Model Specification calls to parsnip</i>
-----------------	--

---

**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
)
```

**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 <code>rsample</code>
.split_args	The default is NULL, when NULL then the default parameters of the split type will be executed for the <code>rsample</code> split type.

**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)
library(dplyr, quietly = TRUE)

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

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

**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: [core\\_packages\(\)](#), [create\\_splits\(\)](#), [create\\_workflow\\_set\(\)](#), [fast\\_classification\\_parsnip\\_spec\\_tbl\(\)](#), [full\\_internal\\_make\\_wflw\(\)](#), [install\\_deps\(\)](#), [load\\_deps\(\)](#), [match\\_args\(\)](#)

**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 (tidyml_mod_spec_tbl).
.rec_obj	A recipe object.

**Details**

The function checks if the input model specification table inherits the class 'tidyml\_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: [core\\_packages\(\)](#), [create\\_splits\(\)](#), [create\\_workflow\\_set\(\)](#), [fast\\_classification\\_parsnip\\_spec\\_tbl\(\)](#), [fast\\_regression\\_parsnip\\_spec\\_tbl\(\)](#), [install\\_deps\(\)](#), [load\\_deps\(\)](#), [match\\_args\(\)](#)

**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", "gee") &
    .parsnip_fns == "linear_reg"
  )
class(mod_tbl) <- c("tidyml_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**

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

**Details**

This function allows you to get a model or models from a tibble with a class of `"tidyml_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\\_wflw\\_fit\(\)](#), [extract\\_wflw\\_pred\(\)](#), [extract\\_wflw\(\)](#)

**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: [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\(\)](#)

**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**

<code>.model_tbl</code>	The model table that is generated from a function like <code>fast_regression_parsnip_spec_tbl()</code> , must have a class of "tidyml_mod_spec_tbl". This is meant to be used after the function <code>internal_make_wflw()</code> has been run and the tibble has been saved.
<code>.splits_obj</code>	The splits object from the <code>auto_ml</code> function. It is internal to the <code>auto_ml</code> 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\\_gee\\_lin\\_reg\(\)](#), [internal\\_make\\_wflw\\_prediction\(\)](#), [internal\\_make\\_wflw\(\)](#), [internal\\_set\\_args\\_to\\_tune\(\)](#), [make\\_classification\\_base\\_tbl\(\)](#), [make\\_regression\\_base\\_tbl\(\)](#)

**Examples**

```
library(recipes, quietly = TRUE)
library(dplyr, 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)
splits_obj <- create_splits(mtcars, "initial_split")

mod_tbl <- mod_spec_tbl |>
  mutate(wflw = 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\\_gee\\_lin\\_reg\(\)](#), [internal\\_make\\_wflw\\_predictions\(\)](#), [internal\\_make\\_wflw\(\)](#), [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 "tidyml\_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\\_set\\_args\\_to\\_tune\(\)](#), [make\\_classification\\_base\\_tbl\(\)](#), [make\\_regression\\_base\\_tbl\(\)](#)

## Examples

```
library(recipes, quietly = TRUE)
library(dplyr, 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

<code>.model_tbl</code>	The model table that is generated from a function like <code>fast_regression_parsnip_spec_tbl()</code> , must have a class of <code>"tidyml_mod_spec_tbl"</code> .
<code>.rec_obj</code>	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\\_predictions\(\)](#), [internal\\_make\\_wflw\(\)](#), [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("tidyml_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**

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

**Details**

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

**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\(\)](#), [internal\\_set\\_args\\_to\\_tune\(\)](#), [make\\_classification\\_base\\_tbl\(\)](#), [make\\_regression\\_base\\_tbl\(\)](#)

**Examples**

```
library(recipes, quietly = TRUE)
library(dplyr, 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)
splits_obj <- create_splits(mtcars, "initial_split")

mod_tbl <- mod_spec_tbl |>
  mutate(wflw = 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\_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 `"tidyml_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_gee_lin_reg()`, `internal_make_wflw_predictions()`, `internal_make_wflw()`, `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: [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\(\)](#)

**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\\_gee\\_lin\\_reg\(\)](#), [internal\\_make\\_wflw\\_predictions\(\)](#), [internal\\_make\\_wflw\(\)](#), [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\\_gee\\_lin\\_reg\(\)](#), [internal\\_make\\_wflw\\_predictions\(\)](#), [internal\\_make\\_wflw\(\)](#), [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: [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\(\)](#)

**Examples**

```
library(parsnip)

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

# Index

- \* **Extractor**
    - extract\_model\_spec, 6
    - extract\_wflw, 7
    - extract\_wflw\_fit, 8
    - extract\_wflw\_pred, 9
    - get\_model, 16
  - \* **Internals**
    - internal\_make\_fitted\_wflw, 18
    - internal\_make\_spec\_tbl, 19
    - internal\_make\_wflw, 20
    - internal\_make\_wflw\_gee\_lin\_reg, 21
    - internal\_make\_wflw\_predictions, 22
    - internal\_set\_args\_to\_tune, 23
    - make\_classification\_base\_tbl, 25
    - make\_regression\_base\_tbl, 26
  - \* **Model Generator**
    - create\_model\_spec, 3
    - fast\_classification, 10
    - fast\_regression, 13
  - \* **Utility**
    - core\_packages, 2
    - create\_splits, 4
    - create\_workflow\_set, 5
    - fast\_classification\_parsnip\_spec\_tbl, 12
    - fast\_regression\_parsnip\_spec\_tbl, 14
    - full\_internal\_make\_wflw, 15
    - install\_deps, 17
    - load\_deps, 24
    - match\_args, 26
- core\_packages, 2, 5, 6, 13, 15, 16, 18, 25, 27
- create\_model\_spec, 3, 11, 14
- create\_splits, 3, 4, 6, 13, 15, 16, 18, 25, 27
- create\_workflow\_set, 3, 5, 5, 13, 15, 16, 18, 25, 27
- extract\_model\_spec, 6, 8–10, 17
- extract\_wflw, 7, 7, 9, 10, 17
- extract\_wflw\_fit, 7, 8, 8, 10, 17
- extract\_wflw\_pred, 7–9, 9, 17
- fast\_classification, 4, 10, 14
- fast\_classification\_parsnip\_spec\_tbl, 3, 5, 6, 12, 15, 16, 18, 25, 27
- fast\_regression, 4, 11, 13
- fast\_regression\_parsnip\_spec\_tbl, 3, 5, 6, 13, 14, 16, 18, 25, 27
- fast\_regression\_parsnip\_spec\_tbl(), 24
- full\_internal\_make\_wflw, 3, 5, 6, 13, 15, 15, 18, 25, 27
- get\_model, 7–10, 16
- install\_deps, 3, 5, 6, 13, 15, 16, 17, 25, 27
- internal\_make\_fitted\_wflw, 18, 20, 21, 23–26
- internal\_make\_spec\_tbl, 18, 19, 20, 21, 23–26
- internal\_make\_wflw, 18, 20, 20, 21, 23–26
- internal\_make\_wflw\_gee\_lin\_reg, 18, 20, 21, 23–26
- internal\_make\_wflw\_predictions, 18, 20, 21, 22, 24–26
- internal\_set\_args\_to\_tune, 18, 20, 21, 23, 23, 25, 26
- load\_deps, 3, 5, 6, 13, 15, 16, 18, 24, 27
- make\_classification\_base\_tbl, 18, 20, 21, 23, 24, 25, 26
- make\_regression\_base\_tbl, 18, 20, 21, 23–25, 26
- match\_args, 3, 5, 6, 13, 15, 16, 18, 25, 26
- workflows::workflow(), 20, 21
- workflowsets::workflow\_set(), 6