

Package ‘hubEnsembles’

July 22, 2025

Title Ensemble Methods for Combining Hub Model Outputs

Version 1.0.0

Description Functions for combining model outputs (e.g. predictions or estimates) from multiple models into an aggregated ensemble model output.

License MIT + file LICENSE

URL <https://github.com/hubverse-org/hubEnsembles>,
<https://hubverse-org.github.io/hubEnsembles/>

BugReports <https://github.com/hubverse-org/hubEnsembles/issues>

Depends R (>= 4.1)

Imports cli, distfromq (>= 1.0.2), dplyr, hubUtils (>= 0.0.1),
lifecycle, matrixStats, purrr, rlang, stats, tidyr, tidyselect

Suggests cowplot, ggplot2, hubExamples, hubVis (>= 0.0.1), knitr,
rmarkdown, testthat (>= 3.0.0)

Additional_repositories <https://hubverse-org.r-universe.dev/>

Config/Needs/website hubverse-org/hubStyle

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author Li Shandross [aut, cre] (ORCID:

<<https://orcid.org/0009-0008-1348-1954>>),

Emily Howerton [aut] (ORCID: <<https://orcid.org/0000-0002-0639-3728>>),

Evan L Ray [aut],

Anna Krystalli [ctb] (ORCID: <<https://orcid.org/0000-0002-2378-4915>>),

Zhian N. Kamvar [ctb] (ORCID: <<https://orcid.org/0000-0003-1458-7108>>),

Nicholas G. Reich [ctb] (ORCID:

<<https://orcid.org/0000-0003-3503-9899>>),

Consortium of Infectious Disease Modeling Hubs [cph]

Maintainer Li Shandross <lshandross@umass.edu>
Repository CRAN
Date/Publication 2025-05-23 17:52:10 UTC

Contents

component_outputs	2
fweights	3
linear_pool	3
make_sample_indices_unique	6
model_outputs	6
simple_ensemble	7
validate_compound_taskid_set	9
weights	10
Index	11

component_outputs	<i>Example model output data for linear_pool()</i>
-------------------	--

Description

Toy model output data formatted according to hubverse standards to be used in the examples for linear_pool(). The predictions included are taken from three normal distributions with means -3, 0, 3 and all standard deviations 1.

Usage

component_outputs

Format

component_outputs:
A data frame with 123 rows and 5 columns:
model_id model ID
target forecast target
output_type type of forecast
output_type_id output type ID
value forecast value

fweights	<i>Example weights data for simple_ensemble()</i>
----------	---

Description

Toy weights data formatted according to hubverse standards to be used in the examples for `simple_ensemble()`

Usage

```
fweights
```

Format

fweights:

A data frame with 8 rows and 3 columns:

model_id model ID

location FIPS codes

weight weight

linear_pool	<i>Compute ensemble model outputs as a linear pool, otherwise known as a distributional mixture, of component model outputs for each combination of model task, output type, and output type id. Supported output types include mean, quantile, cdf, pmf, and sample.</i>
-------------	---

Description

Compute ensemble model outputs as a linear pool, otherwise known as a distributional mixture, of component model outputs for each combination of model task, output type, and output type id. Supported output types include mean, quantile, cdf, pmf, and sample.

Usage

```
linear_pool(
  model_out_tbl,
  weights = NULL,
  weights_col_name = "weight",
  model_id = "hub-ensemble",
  task_id_cols = NULL,
  compound_taskid_set = NA,
  derived_task_ids = NULL,
  n_samples = 10000,
  n_output_samples = NULL,
  ...,
  derived_tasks = lifecycle::deprecated()
)
```

Arguments

<code>model_out_tbl</code>	an object of class <code>model_out_tbl</code> with component model outputs (e.g., predictions).
<code>weights</code>	an optional <code>data.frame</code> with component model weights. If provided, it should have a column named <code>model_id</code> and a column containing model weights. Optionally, it may contain additional columns corresponding to task id variables, <code>output_type</code> , or <code>output_type_id</code> , if weights are specific to values of those variables. The default is <code>NULL</code> , in which case an equally-weighted ensemble is calculated. Should be prevalidated.
<code>weights_col_name</code>	character string naming the column in <code>weights</code> with model weights. Defaults to <code>"weight"</code>
<code>model_id</code>	character string with the identifier to use for the ensemble model.
<code>task_id_cols</code>	character vector with names of columns in <code>model_out_tbl</code> that specify modeling tasks. Defaults to <code>NULL</code> , in which case all columns in <code>model_out_tbl</code> other than <code>"model_id"</code> , <code>"output_type"</code> , <code>"output_type_id"</code> , and <code>"value"</code> are used as task ids.
<code>compound_taskid_set</code>	<p>character vector of the compound task ID variable set. This argument is only relevant for <code>output_type "sample"</code>. Can be one of three possible values, with the following meanings:</p> <ul style="list-style-type: none"> • <code>NA</code>: the <code>compound_taskid_set</code> is not relevant for the current modeling task • <code>NULL</code>: samples are from a multivariate joint distribution across all levels of all task id variables • Equality to <code>task_id_cols</code>: samples are from separate univariate distributions for each individual prediction task <p>Defaults to <code>NA</code>. Derived task ids must be included if all of the task ids their values depend on are part of the <code>compound_taskid_set</code>.</p>
<code>derived_task_ids</code>	character vector of derived task IDs (variables whose values depend on that of other task ID variables). Defaults to <code>NULL</code> , meaning there are no derived task IDs.
<code>n_samples</code>	numeric that specifies the number of samples to use when calculating quantiles from an estimated quantile function. Defaults to <code>1e4</code> .
<code>n_output_samples</code>	numeric that specifies how many sample forecasts to return per unique combination of task IDs. Currently the only supported value is <code>NULL</code> , in which case all provided component model samples are collected and returned.
<code>...</code>	parameters that are passed to <code>distfromq::make_q_fn</code> , specifying details of how to estimate a quantile function from provided quantile levels and quantile values for <code>output_type "quantile"</code> .
<code>derived_tasks</code>	[Deprecated] Use <code>derived_task_ids</code> instead. A character vector of derived task IDs.

Details

The underlying mechanism for the computations varies for different `output_types`. When the `output_type` is `cdf`, `pmf`, or `mean`, this function simply calls `simple_ensemble` to calculate a (weighted) mean of the component model outputs. This is the definitional calculation for the CDF or PMF of a linear pool. For the mean output type, this is justified by the fact that the (weighted) mean of the linear pool is the (weighted) mean of the means of the component distributions.

When the `output_type` is `quantile`, we obtain the quantiles of a linear pool in three steps:

1. Interpolate and extrapolate from the provided quantiles for each component model to obtain an estimate of the CDF of that distribution.
2. Draw samples from the distribution for each component model. To reduce Monte Carlo variability, we use quasi-random samples corresponding to quantiles of the estimated distribution.
3. Collect the samples from all component models and extract the desired quantiles.

Steps 1 and 2 in this process are performed by `distfromq::make_q_fn`.

When the `output_type` is `sample`, we obtain the resulting linear pool by collecting samples from each model, updating the `output_type_id` values to be unique for predictions that are not joint across, and pooling them together. If there is a restriction on the number of samples to output per compound unit, this number is divided evenly among the models for that compound unit (with any remainder distributed randomly).

Value

a `model_out_tbl` object of ensemble predictions. Note that any additional columns in the input `model_out_tbl` are dropped.

Examples

```
# We illustrate the calculation of a linear pool when we have quantiles from the
# component models. We take the components to be normal distributions with
# means -3, 0, and 3, all standard deviations 1, and weights 0.25, 0.5, and 0.25.
data(component_outputs)
data(weights)

expected_quantiles <- seq(from = -5, to = 5, by = 0.25)
lp_from_component_qs <- linear_pool(component_outputs, weights)

head(lp_from_component_qs)
all.equal(lp_from_component_qs$value, expected_quantiles, tolerance = 1e-2,
          check.attributes = FALSE)
```

make_sample_indices_unique	<i>Make the output type ID values of sample forecasts distinct for different models</i>
----------------------------	---

Description

Make the output type ID values of sample forecasts distinct for different models

Usage

```
make_sample_indices_unique(model_out_tbl)
```

Arguments

model_out_tbl an object of class model_out_tbl with component model outputs (e.g., predictions).

Details

The new output_type_id column values will follow one of two patterns, depending on whether the column is detected to be numeric:

1. If the output type ID is not numeric (may be a character): A concatenation of the prediction’s model ID and original output type ID
2. If the output type ID is numeric: A numeric representation of the above pattern rendered as a factor.

Value

a model_out_tbl object with unique output type ID values for different models but otherwise identical to the input model_out_tbl.

model_outputs	<i>Example model output data for simple_ensemble()</i>
---------------	--

Description

Toy model output data formatted according to hubverse standards to be used in the examples for simple_ensemble()

Usage

```
model_outputs
```

Format

model_outputs:

A data frame with 24 rows and 8 columns:

model_id model ID

location FIPS codes

horizon forecast horizon

target forecast target

target_date date that the forecast is for

output_type type of forecast

output_type_id output type ID

value forecast value

simple_ensemble	<i>Compute ensemble model outputs by summarizing component model outputs for each combination of model task, output type, and output type id. Supported output types include mean, median, quantile, cdf, and pmf.</i>
-----------------	--

Description

Compute ensemble model outputs by summarizing component model outputs for each combination of model task, output type, and output type id. Supported output types include mean, median, quantile, cdf, and pmf.

Usage

```
simple_ensemble(
  model_out_tbl,
  weights = NULL,
  weights_col_name = "weight",
  agg_fun = mean,
  agg_args = list(),
  model_id = "hub-ensemble",
  task_id_cols = NULL
)
```

Arguments

model_out_tbl	an object of class <code>model_out_tbl</code> with component model outputs (e.g., predictions).
weights	an optional <code>data.frame</code> with component model weights. If provided, it should have a column named <code>model_id</code> and a column containing model weights. Optionally, it may contain additional columns corresponding to task id variables, <code>output_type</code> , or <code>output_type_id</code> , if weights are specific to values of those variables. The default is <code>NULL</code> , in which case an equally-weighted ensemble is calculated. Should be prevalidated.

<code>weights_col_name</code>	character string naming the column in <code>weights</code> with model weights. Defaults to "weight"
<code>agg_fun</code>	a function or character string name of a function to use for aggregating component model outputs into the ensemble outputs. See the details for more information.
<code>agg_args</code>	a named list of any additional arguments that will be passed to <code>agg_fun</code> .
<code>model_id</code>	character string with the identifier to use for the ensemble model.
<code>task_id_cols</code>	character vector with names of columns in <code>model_out_tbl</code> that specify modeling tasks. Defaults to <code>NULL</code> , in which case all columns in <code>model_out_tbl</code> other than "model_id", "output_type", "output_type_id", and "value" are used as task ids.

Details

The default for `agg_fun` is "mean", in which case the ensemble's output is the average of the component model outputs within each group defined by a combination of values in the task id columns, output type, and output type id. The provided `agg_fun` should have an argument `x` for the vector of numeric values to summarize, and for weighted methods, an argument `w` with a numeric vector of weights. If it desired to use an aggregation function that does not accept these arguments, a wrapper would need to be written. For weighted methods, `agg_fun = "mean"` and `agg_fun = "median"` are translated to use `matrixStats::weightedMean` and `matrixStats::weightedMedian` respectively. For `matrixStats::weightedMedian`, the argument `interpolate` is automatically set to `FALSE` to circumvent a calculation issue that results in invalid distributions.

Value

a `model_out_tbl` object of ensemble predictions. Note that any additional columns in the input `model_out_tbl` are dropped.

Examples

```
# Calculate a weighted median in two ways
data(model_outputs)
data(fweights)

weighted_median1 <- simple_ensemble(model_outputs, weights = fweights,
                                   agg_fun = stats::median)
weighted_median2 <- simple_ensemble(model_outputs, weights = fweights,
                                   agg_fun = matrixStats::weightedMedian)
all.equal(weighted_median1, weighted_median2)
```

 validate_compound_taskid_set

Perform validations on the compound task ID set used to calculate an ensemble of component model outputs for the sample output type, including checks that (1) compound_taskid_set is a subset of task_id_cols, (2) the provided model_out_tbl is compatible with the specified compound_taskid_set, and (3) all models submit predictions for the same set of non compound_taskid_set variables.

Description

Perform validations on the compound task ID set used to calculate an ensemble of component model outputs for the sample output type, including checks that (1) compound_taskid_set is a subset of task_id_cols, (2) the provided model_out_tbl is compatible with the specified compound_taskid_set, and (3) all models submit predictions for the same set of non compound_taskid_set variables.

Usage

```
validate_compound_taskid_set(
  model_out_tbl,
  task_id_cols,
  compound_taskid_set,
  derived_task_ids = NULL,
  return_missing_combos = FALSE
)
```

Arguments

model_out_tbl an object of class model_out_tbl with component model outputs (e.g., predictions).

task_id_cols character vector with names of columns in model_out_tbl that specify modeling tasks. Defaults to NULL, in which case all columns in model_out_tbl other than "model_id", "output_type", "output_type_id", and "value" are used as task ids.

compound_taskid_set character vector of the compound task ID variable set. This argument is only relevant for output_type "sample". Can be one of three possible values, with the following meanings:

- NA: the compound_taskid_set is not relevant for the current modeling task
- NULL: samples are from a multivariate joint distribution across all levels of all task id variables
- Equality to task_id_cols: samples are from separate univariate distributions for each individual prediction task

Defaults to NA. Derived task ids must be included if all of the task ids their values depend on are part of the compound_taskid_set.

`derived_task_ids`
character vector of derived task IDs (variables whose values depend on that of other task ID variables). Defaults to `NULL`, meaning there are no derived task IDs.

`return_missing_combos`
boolean specifying whether to return a `data.frame` summarizing the missing combinations of dependent tasks for each model. If `TRUE`, the columns of the `data.frame` will be `"model_id"` and one for each of the dependent tasks (complement of the `compound_taskid_set`). Defaults to `FALSE`.

Value

If `model_out_tbl` passes the validations, there will be no return value. Otherwise, the function will either throw an error if `return_missing_combos` is `FALSE`, or a `data.frame` of the missing combinations of dependent tasks will be returned. See above for more details.

<code>weights</code>	<i>Example weights data for <code>linear_pool()</code></i>
----------------------	--

Description

Toy weights data formatted according to hubverse standards to be used in the examples for `linear_pool()`. Weights are 0.25, 0.5, 0.25.

Usage

`weights`

Format

`weights`:
A data frame with 3 rows and 2 columns:
model_id model ID
location FIPS codes
weight weight

Index

* **datasets**

component_outputs, [2](#)

fweights, [3](#)

model_outputs, [6](#)

weights, [10](#)

component_outputs, [2](#)

fweights, [3](#)

linear_pool, [3](#)

make_sample_indices_unique, [6](#)

model_outputs, [6](#)

simple_ensemble, [7](#)

validate_compound_taskid_set, [9](#)

weights, [10](#)