Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

modelimportance: An R package for evaluating model importance within an ensemble

Minsu Kim
University of Massachusetts Amherst

Evan Rays
CVS

Nicholas G. Reich
University of Massachusetts Amherst

Abstract

Ensemble forecasts are commonly used to support decision-making and policy planning across various fields because they often offer improved accuracy and stability compared to individual models. As each model has its own unique characteristics, understanding and measuring the value each constituent model adds to the overall accuracy of the ensemble is of great interest to building effective ensembles. The R package **modelimportance** provides tools to quantify how each component model contributes to the acccuracy of ensemble performance for different forecast types. It supports multiple functionalities; it allows users to specify which ensemble approach to implement and which model importance metric to use. Additionally, the software offers customizable options for handling missing values. These features enable the package to serve as a versatile tool for researchers and practitioners aiming to construct an effective ensemble model across a wide range of forecasting tasks.

Keywords: ensemble, forecast, prediction, model importance, Shapley value, R.

1. Introduction

Ensemble forecasting is a method to produce a single, consolidated prediction by combining forecasts generated from different models. This technique it utilizes each model's strengths to counterbalance their individual weaknesses, leading to more robust and accurate predictions (Gneiting and Raftery 2005; Hastie *et al.* 2001; Lutz *et al.* 2019; Viboud *et al.* 2018). Specifically, ensembles aggregate diverse insights from various models and effectively handle individual models models biases and variances by averaging them out, which can reduce prediction errors and improve overall model performance. These enhanced prediction accuracy and robustness enable ensemble forecasting to be widely used across various domains such as weather forecasting (Jordan A. Guerra *et al.* 2020; Gneiting and Raftery 2005), financial modeling (Sun *et al.* 2020; He *et al.* 2023), and infectious disease outbreak forecasting (Ray and Reich 2018; Reich *et al.* 2019) to improve decision-making and achieve more reliable predictions. For example, throughout the COVID-19 pandemic, the US COVID-19 Forecast Hub collected individual models developed by over 90 different research groups and built a probabilistic ensemble forecasting model for COVID-19 cases, hospitalizations, and deaths in the US based on those models' predictions, which served as the official short-term forecasts for the US Centers for Disease Control and Prevention (CDC) (Kim *et al.* 2024).

The quality of forecasts is assessed by evaluating their error, bias, sharpness, and/or calibration using different scoring metrics. The selection of the scoring metrics depends on the type of forecast, such as point forecasts and probabilistic forecasts, and their corresponding formats, such as median/mean, quantiles, and predictive cumulative distribution functions. In the case of point forecasts, accuracy is commonly measured by the mean absolute error (MAE) and the mean squared error (MSE), which are direct assessment tools that calculate the average magnitude of forecast errors. Probabilistic forecasts are typically assessed using proper scoring rules, which consider the uncertainty and variability in predictions, providing concise evaluations through numerical scores (Gneiting and Raftery 2007). Some examples include the weighted interval score (WIS) for the quantile-based forecasts and the continuous ranked probability score (CRPS) for the forecasts taking the form of predictive cumulative distribution functions (Bracher *et al.* 2021).

Several R packages have been developed for this purpose. To name a few, the **forecast** package (Hyndman and Khandakar 2008) is widely used for univariate time series forecasting and includes functions for accuracy measurement. The **Metrics** (Hamner and Frasco 2018) and **MLmetrics** (Yan 2024) provide a wide range of performance metrics specifically designed for evaluating machine learning models. The **scoringRules** (Jordan *et al.* 2019) package offers a comprehensive set of proper scoring rules for evaluating probabilistic forecasts and supports both univariate and multivariate settings. The **scoringutils** (Bosse *et al.* 2022) package offers additional features to the functionality provided by **scoringRules**, which makes it more useful for certain tasks, such as summarizing, comparing, and visualizing forecast performance. Additionally, the **scoringutils** supports forecasts represented by predictive samples or quantiles of predictive distributions, allowing for the assessment of any forecast type, even when a closed-form expression for a parametric distribution is not available. These packages have been valuable to evaluate individual models as independent entities, using performance metrics selected for each specific situation or problem type. However, they do not measure the individual models' contributions to the enhanced predictive accuracy when used as part of an ensemble. Kim et al. (Kim *et al.* 2024) demonstrate that a model's individual performance does not necessarily correspond to its contribution as a component within an ensemble. Our developed package introduces this capability. The **modelimportance** package provides tools to evaluate the role of each model as an ensemble member within an ensemble model, rather than focusing on the individual predictive performance per se.

In ensemble forecasting, certain models contribute more significantly to the overall predictions than others. Assessing the impact of each component model on ensemble predictions is methodologically similar to determining variable importance in traditional regression and machine learning models, where variable importance measures evaluate how much individual variables enhance the model's predictive performance. R packages that implement these functions include **randomForest** (Liaw and Wiener 2002), **caret** (Kuhn 2008), **xgboost** (Chen *et al.* 2024), and **gbm** (Ridgeway and Developers 2024), each providing variable importance measures for different types of models: random forest models, general machine learning models, extreme gradient boosting models, and generalized boosted regression models, respectively. Similarly, the tools in the **modelimportance** package quantify the contribution of each component model within an ensemble to the ensemble model's predictive performance. They assign numerical scores to each model based on a selected metric that measures forecast accuracy, depending on the forecast type.

These capabilities provide unique support for hub organizers, such as the CDC in the US and the European Centre for Disease Prevention and Control in the EU. These organizations can create more effective ensemble forecasts by gathering and utilizing forecasts from various models based on the precise evaluation of each model's contribution via the **modelimportance** package. This, in turn, facilitates better communication with the public and decision-makers and enhances the overall decision-making process. Specifically, **modelimportance** is incorporated into the 'hubverse', which is a collection of open-source software and data tools developed to promote collaborative modeling hub efforts and reduce the effort required to set up and operate them (Consortium of Infectious Disease Modeling Hubs 2024). This package adheres to the model output formats specified by the hubverse convention, which enables seamless integration and interoperability with other forecasting tools and systems.

The paper proceeds as follows. In Section 2, we address the model output formats proposed within the hubverse framework and the structure of forecasts, followed by a motivating example. Section 3 presents two algorithms implemented in **modelimportance** for calculating the model importance metric: leave-one-model-out and leave-all-subsets-of-models-out. We demonstrate the various functionalities **modelimportance** supports in Section 4 and give some examples in Section 5. We close this paper with some concluding remarks and a discussion of possible extensions.

2. Data

2.1. Model output format

Model outputs are structured in a tabular format designed specifically for predictions. In the hubverse standard, each row represents an individual prediction for a single task, and its details are described in multiple columns through which one can identify the model IDs, task characteristics, prediction representation type, and predicted values (Shandross *et al.* 2024). In Table 1, for example, the `model_id` column contains the uniquely identified name of the model that produced the prediction in each row. Task characteristics are represented by `reference_date`, `target`, `horizon`, `location`, and `target_end_date` columns, collectively referred to as the task ID columns. The prediction representation type is specified in the `output_type` and `output_type_id` columns. This example illustrates short-term forecasts of incident influenza hospitalizations in the US for Massachusetts (FIPS code 25), generated by the model 'Flusight-baseline' on November 19, 2022. The forecasts are provided in seven

model_id	reference_date	target	horizon	location	target_end_date	output_type	output_type_id	value
Flusight-baseline	2022-11-19	wk inc flu hosp	0	25	2022-11-19	quantile	0.05	22
Flusight-baseline	2022-11-19	wk inc flu hosp	0	25	2022-11-19	quantile	0.1	31
Flusight-baseline	2022-11-19	wk inc flu hosp	0	25	2022-11-19	quantile	0.25	45
Flusight-baseline	2022-11-19	wk inc flu hosp	0	25	2022-11-19	quantile	0.5	51
Flusight-baseline	2022-11-19	wk inc flu hosp	0	25	2022-11-19	quantile	0.75	57
Flusight-baseline	2022-11-19	wk inc flu hosp	0	25	2022-11-19	quantile	0.9	71
Flusight-baseline	2022-11-19	wk inc flu hosp	0	25	2022-11-19	quantile	0.95	80
Flusight-baseline	2022-11-19	wk inc flu hosp	1	25	2022-11-26	quantile	0.05	5
Flusight-baseline	2022-11-19	wk inc flu hosp	1	25	2022-11-26	quantile	0.1	21
Flusight-baseline	2022-11-19	wk inc flu hosp	1	25	2022-11-26	quantile	0.25	38

Table 1: Example of the model output for incident influenza hospitalizations extracted from `forecast_outputs` data in the **hubExamples** package.

Output Type	Scoring Rule	Description
mean	MSE	Evaluate using the mean squared error (MSE)
median	MAE	Evaluate using the mean absolute error (MAE)
quantile	WIS	Evaluate using the weighted interval score (WIS)
pmf	Log Score	Evaluate using the logarithm of the probability assigned to the true outcome (LogScore)

Table 2: Pairs of output types and their associated scoring rules for evaluating prediction performance.

quantiles (0.05, 0.1, 0.25, 0.5, 0.75, 0.9, 0.95) for each target end date.

2.2. Structure of forecasts

The forecasting model supports multiple output types. Generally, quantitative forecasts can be categorized into either point forecasts or probabilistic forecasts. For a specific task, point forecasts, represented by a single predicted value, provide a clear and concise prediction, making them easy to interpret and communicate. Probabilistic forecasts, on the other hand, describe the likelihood of different outcomes, conveying the uncertainty inherent in the prediction. These forecasts are represented by a probability distribution over possible future values in various ways, such as probability mass functions (pmf), cumulative distribution functions (cdf), or probability quantiles (or intervals).

The `output_type` and `output_type_id` columns in the model output format, as defined by the hubverse convention, specify the forecast structure. The `output_type` column takes the character value 'mean' or 'median' for point forecasts, and 'pmf', 'cdf', or 'quantile' for probabilistic forecasts. The `output_type_id` column provides additional details for probabilistic forecasts; for instance, when the `output_type` is 'quantile', the `output_type_id` identifies specific quantile levels, as illustrated in Table 1. In the case of point forecasts, `output_type_id` column has 'NA'. Different output types correspond to different scoring rules for evaluating a model's prediction performance. Table 2 presents the output types and their associated scoring rules supported by the **modelimportance** package.

3. Algorithms

This section provides a brief description of the leave one model out (LOMO) and leave all subsets of models out (LASOMO) algorithms. (Details can be found in Kim *et al.* (2024))

LOMO involves creating an ensemble by excluding one component model from the entire set of models. Let A be a set of n models and F^i be a forecast produced by model i, where $i = 1, 2, \ldots, n$. Each ensemble excludes exactly one model while including all the others. Denoting by $F^{A^{-i}}$ an ensemble forecast constructed without F^i and by F^A the ensemble forecast built from the entire set of models, the importance score using LOMO is calculated by the difference of measures of these two ensemble performances, $F^{A^{-i}}$ and F^A. For example, when evaluating model 1 within an ensemble of three models ($n = 3$), LOMO creates an ensemble forecast $F^{\{2,3\}}$ using only F^2 and F^3. The performance of this reduced ensemble is then compared to the full ensemble forecast $F^{\{1,2,3\}}$, which incorporates all three models.

On the other hand, LASOMO involves ensemble constructions from all possible subsets of models. For each subset S that does not contain the model i, $S \cup \{i\}$ plays a role of A in the LOMO; the score associated with the subset S is the difference of measures between F^S and $F^{S\cup\{i\}}$. Then, all scores are aggregated across all possible subsets that the model i does not belong to. For example, using the earlier setup of three forecast models, LASOMO considers three subsets, $\{2\}$, $\{3\}$, and $\{2, 3\}$, to calculate the importance score of model 1 (excluding all subsets that include model 1). The ensemble forecasts $F^{\{2\}}, F^{\{3\}}$, and $F^{\{2,3\}}$ are then compared to $F^{\{1,2\}}, F^{\{1,3\}}$, and $F^{\{1,2,3\}}$, respectively. The performance differences attributable to model 1's inclusion are aggregated, which results in the importance score of model 1. We note that the subsets may have different weights during the aggregating process. The **modelimportance** package offers two weighting options for subsets: one assigns equal (uniform) weights to all subsets, and the other assigns weights based on their size, similar to the concept of Shapley values. Users can choose one to evaluate the contribution of each model in a manner suited to their preferred framework.

Algorithm 1 Importance score using leave one model out (LOMO) algorithm

Input:

- Set of n individual models, $A = \{1, 2, \ldots, n\}$, and their forecasts $\{F^1, F^2, ..., F^n\}$
- Truth value, y
- Function g to build an ensemble
- Scoring rule ψ to evaluate forecast skills

Output: importance metric of model i, $\phi^{i,\mathrm{lomo}}$

Create an ensemble forecast F^A using g: $F^A \leftarrow g(\{F^1, F^2, ..., F^n\})$
Evaluate F^A using ψ: $\psi(F^A, y)$.
for each $i, (i = 1, 2, ..., n)$ **do**
    $F^{A^{-i}} \leftarrow g(\{F^j | j \neq i, j \in A\})$
    Compute $\psi(F^{A^{-i}}, y)$.
    $\phi^{i,\mathrm{lomo}} \leftarrow \psi(F^{A^{-i}}, y) - \psi(F^A, y)$
end for

Algorithms 1 and 2 outline the steps to implement LOMO and LASOMO for a single prediction task, respectively.

Algorithm 2 Importance score using leave all subsets of models out (LASOMO) algorithm

Input:

- Set of n individual models, $A = \{1, 2, \dots, n\}$, and their forecasts $\{F^1, F^2, \dots, F^n\}$
- Truth value, y
- Function g to build an ensemble
- Scoring rule ψ to evaluate forecast skills

Output: Importance metric of model i, $\phi^{i,\text{lasomo}}$

```
for i = 1 to n do
    φ^{i,lasomo} ← 0
    Make a list of non-empty subsets of A that does not contain i: S_1, S_2, ⋯, S_{2^{n−1}−1}
    for j = 1 to 2^{n−1} − 1 do
        Assign a weight to the subset S_j:
        if all subsets have uniform weights then
            γ_{S_j} ← 1 / (2^{n−1} − 1)
        else (subset's weight depends on its size)
            γ_{S_j} ← 1 / ((n − 1) binom(n−1, |S_j|))
        end if
        F^{S_j} ← g({F^j | j ∈ S_j})
        F^{S_j ∪ {i}} ← g({F^i, F^j | j ∈ S_j})
        Compute ψ(F^{S_j}, y) and ψ(F^{S_j ∪ {i}}, y)
        φ^{i,lasomo} ← φ^{i,lasomo} + γ_{S_j} × [ψ(F^{S_j}, y) − ψ(F^{S_j ∪ {i}}, y)]
    end for
end for
```

4. Main funtion: Evaluating ensemble members

In this section, we describe the functionalities of the main function `model_importance()`, where multiple options are available to customize the evaluation framework (Table 3).

```
> model_importance(forecast_data, oracle_output_data, ensemble_fun, weighted,
                   training_window_length, importance_algorithm, subset_wt,
                   na_action, ...)
```

`forecast_data` is a data frame containing predictions and should be or can be coerced to a `model_out_tbl` format, which is the standard S3 class model output format defined by the hubverse convention. Only one `output_type` is allowed in the data frame, and it must be one of the `mean`, `median`, `quantile`, or `pmf`.

`oracle_output_data` is a data frame containing the ground truth data for the variables that

Argument	Description	Possible Values	Default
`forecast_data`	Forecasts	Must be the model output format	N/A
`oracle_output_data`	Ground truth data	Must be the oracle output format	N/A
`ensemble_fun`	Ensemble method	`simple_ensemble`, `linear_pool`	`simple_ensemble`
`training_window_length`	Time interval of historical data used during the training process	Non-negative integer	0
`importance_algorithm`	Algorithm to calculate importance	`lomo, lasomo`	`lomo`
`subset_wt`	Method for assigning weight to subsets when using lasomo algorithm	`equal, perm_based`	`equal`
`na_action`	Method to handle missing data	`worst, average, drop`	`worst`
`...`	Optional arguments for `simple_ensemble`	Varies	`agg_fun = mean`

Table 3: Description of the arguments for the `model_importance()` function, including their purpose, possible values, and default settings.

are used to define modeling targets. This data must follow the oracle output format, which includes independent task ID columns (e.g., `location`, `target_date`, and `age_group`), the `output_type` column specifying the output type of the predictions and an `oracle_value` column for the observed values. If the `output_type` is either `"quantile"` or `"pmf"`, the `output_type_id` column is required to provide further identifying information. For `"quantile"`, it should contain numeric values between 0 and 1 indicating quantile levels (e.g., "0.1", "0.25", "0.5", "0.75", "0.9"). For `"pmf"`, it should contain categorical values such as "low", "moderate", "high", and "very high".

The `forecast_data` and `oracle_output_data` must have the same task ID columns and `output_type`, including `output_type_id` if necessary, which are used to match the predictions with the ground truth data. As aforementioned, these data frames should follow the model output format and the oracle output format, respectively, as defined by the hubverse convention.

The `ensemble_fun` argument specifies the ensemble method to be used for evaluating model importance. The currently supported methods are `"simple_ensemble"` and `"linear_pool"`. The `"simple_ensemble"` method returns the average of the predicted values from all component models per prediction task defined by task IDs, `output_type`, and `output_type_id` columns. The default aggregation function for this method is `"mean"`, but it can be customized by specifying additional arguments through `...`, such as `agg_fun="median"`. When `"linear_pool"` is specified, ensemble model outputs are created as a linear pool of component model outputs. This method supports only an `output_type` of `"mean"`, `"quantile"`, or `"pmf"`.

The `weighted` argument is a logical value that indicates whether model weighting should be done when building an ensemble using the specified `ensemble_fun`. If it is set to `TRUE`, model weights are estimated based on the previous performance of each model, and these weights are used to build the ensemble.

The `importance_algorithm` argument specifies the algorithm for model importance calculation, which can be either `"lomo"` (leave-one-model-out) and `"lasomo"` (leave all subsets of models out). The `subset_wt` argument is employed only for the `"lasomo"` algorithm. This argument has two options: `"equal"` assigns equal weight to all subsets and `"perm_based"` assigns weight averaged over all possible permutations as in the formula of Shapley values (Algorithm 2). The default values of `importance_algorithm` and `subset_wt` are `"lomo"` and `"equal"`, respectively.

The `na_action` argument allows for specifying how to handle missing values in the `forecast_data`. Three options are available: `"worst"`, `"average"`, and `"drop"`. In each specific prediction task, if a model has any missing predictions, the `"worst"` option replaces those missing values with the smallest value from the other models, while the `"average"` option replaces them with the average of the other models' predictions in that task. The `"drop"` option removes missing values, which results in the exclusion of the model from the evaluation for that task.

5. Examples

The examples in this section illustrate the use of the `model_importance()` function to evaluate the importance of component models within an ensemble, using various combinations of the arguments described in Section 4. We use some example forecast and target data from the **hubExamples** package, which provides sample datasets for multiple modeling hubs in the hubverse format.

5.1. Evaluation using untrained ensemble in LOMO algorithm for mean forecasts

The forecast data used here contains forecasts of weekly incident influenza hospitalizations in the US for Massachusetts (FIPS code 25) and Texas (FIPS code 48), generated on November 19, 2022. These forecasts are for two target end dates, November 26, 2022 (horizon 1), and December 10, 2022 (horizon 3), and were produced by three models: 'Flusight-baseline', 'MOBS-GLEAM_FLUH', and 'PSI-DICE'. The output type is `mean` and the `output_type_id` column has `NA`s as no further specification is required for this output type. We have modified the example data slightly: the forecast values were rounded to the nearest integer and some forecasts have been removed to demonstrate the handling of missing values. Therefore, 'MOBS-GLEAM_FLUH' 's forecast for Massachusetts on November 26, 2022, and 'PSI-DICE's forecast for Texas on December 10, 2022, are missing.

```
> forecast_data

# A tibble: 10 x 9
   model_id          reference_date target           horizon location
   <chr>             <date>         <chr>              <int> <chr>
```

```
 1 Flusight-baseline 2022-11-19     wk inc flu hosp      1 25
 2 Flusight-baseline 2022-11-19     wk inc flu hosp      3 25
 3 Flusight-baseline 2022-11-19     wk inc flu hosp      1 48
 4 Flusight-baseline 2022-11-19     wk inc flu hosp      3 48
 5 MOBS-GLEAM_FLUH   2022-11-19     wk inc flu hosp      3 25
 6 MOBS-GLEAM_FLUH   2022-11-19     wk inc flu hosp      1 48
 7 MOBS-GLEAM_FLUH   2022-11-19     wk inc flu hosp      3 48
 8 PSI-DICE          2022-11-19     wk inc flu hosp      1 25
 9 PSI-DICE          2022-11-19     wk inc flu hosp      3 25
10 PSI-DICE          2022-11-19     wk inc flu hosp      1 48
   target_end_date output_type output_type_id value
   <date>          <chr>       <chr>          <dbl>
 1 2022-11-26      mean        <NA>              51
 2 2022-12-10      mean        <NA>              53
 3 2022-11-26      mean        <NA>            1052
 4 2022-12-10      mean        <NA>            1053
 5 2022-12-10      mean        <NA>              47
 6 2022-11-26      mean        <NA>            1073
 7 2022-12-10      mean        <NA>             701
 8 2022-11-26      mean        <NA>              92
 9 2022-12-10      mean        <NA>             159
10 2022-11-26      mean        <NA>            1222
```

The corresponding target data contains the observed hospitalization counts for these dates and locations.

```
> target_data
```

```
# A tibble: 4 x 3
  target_end_date location oracle_value
  <date>          <chr>           <dbl>
1 2022-11-26      25                221
2 2022-11-26      48               1929
3 2022-12-10      25                578
4 2022-12-10      48               1781
```

Overall, the forecasts tend to have larger prediction errors for the target end date of December 10, 2022, compared to November 26, 2022, which is expected due to increased uncertainty at longer horizons. Additionally, the forecasts for Massachusetts are relatively more accurate compared to those for Texas, which tend to have higher errors.

We can evaluate the importance of each model in the ensemble using the `model_importance()` function. The following code evaluates the importance of each model in the simple mean ensemble using the LOMO algorithm, without training the ensemble (`weighted = FALSE`). The `na_action` argument is set to `"drop"`, which represents that any missing values in the forecasts will be excluded from the evaluation.

```
> model_importance(
        forecast_data = forecast_data, oracle_output_data = target_data,
        ensemble_fun = "simple_ensemble", weighted = FALSE,
        importance_algorithm = "lomo", na_action = "drop"
        )
```

This call generates both the result and informative messages, summarizing the input data, including the number of dates on which forecasts were produced and the number of models with ids as follows.

```
Forecasts from 2022-11-19 to 2022-11-19 (a total of 1 forecast date(s)).
The available model IDs are:
     Flusight-baseline
     MOBS-GLEAM_FLUH
     PSI-DICE
(a total of 3 models)
```

The function output is a data frame containing model IDs and their corresponding importance scores in the `mean_importance` column, ordered from most to least importance. The column name `mean_importance` indicates the average of task-specific importance scores.

```
# A tibble: 3 x 2
  model_id          mean_importance
  <chr>                       <dbl>
1 Flusight-baseline          69149.
2 MOBS-GLEAM_FLUH          -113477.
3 PSI-DICE                   44303.
```

The results show that the model 'Flusight-baseline' has the highest importance score, followed by 'PSI-DICE' and 'MOBS-GLEAM_FLUH'. That is, 'Flusight-baseline' contributes the most to improving the ensemble's predictive performance, whereas MOBS-GLEAM_FLUH, which has a negative score, detracts from the ensemble's performance. The low importance score of 'MOBS-GLEAM_FLUH' is mainly due to a substantially larger prediction error for Texas on the target end date of December 10, 2022, compared to other models, while its missing forecast for Massachusetts for November 26, 2022, was not factored into the evaluation. This single large error significantly affected its contribution score.

Another approach to handling missing values is to use the `"worst"` option for `na_action`, which replaces missing values with the worst (i.e., minimum) score among the other models for the same task.

```
> model_importance(
        forecast_data = forecast_data, oracle_output_data = target_data,
        ensemble_fun = "simple_ensemble", weighted = FALSE,
        importance_algorithm = "lomo", na_action = "worst"
        )
```

```
# A tibble: 3 x 2
  model_id          mean_importance
  <chr>                       <dbl>
1 Flusight-baseline          69149.
2 MOBS-GLEAM_FLUH           -86535.
3 PSI-DICE                  -38581.
```

The results show that the importance scores of 'Flusight-baseline' is unchanged because it has no missing forecast. We observe that the importance score of 'PSI-DICE', which was previously positive, has now decreased to a negative value when compared to the evaluation using the `"drop"` option for `na_action`. Moreover, 'MOBS-GLEAM_FLUH' still ranks the lowest, but the importance score has increased. This change is related to the varying forecast accuracy across different tasks. For the target end date of November 26, 2022, in Massachusetts, most forecasts are relatively accurate. Thus, even if the 'MOBS-GLEAM_FLUH' is assigned the worst value of importance score for its missing forecast, including this value in the averaging is not detrimental to the overall importance metric; rather, it is more beneficial than excluding it. In contrast, for the target end date of December 10, 2022, in Texas, the forecasts have much larger errors across the board, and assigning the worst value of importance score to the missing forecast of 'PSI-DICE' in this task has a detrimental effect on averaging importance scores. This is because the scale of the importance scores is influenced by the magnitude of the prediction errors: for tasks with small errors, the scores remain moderate, while tasks with large errors can yield importance scores of much greater magnitude.

It is also possible to impute the missing scores with intermediate values by assigning the average importance scores of other models in the same task. This strategy may offer a more balanced trade-off by mitigating the influence of the missing data without overly penalizing or overlooking them.

```
> model_importance(
        forecast_data = forecast_data, oracle_output_data = target_data,
        ensemble_fun = "simple_ensemble", weighted = FALSE,
        importance_algorithm = "lomo", na_action = "average"
        )
```

```
# A tibble: 3 x 2
  model_id          mean_importance
  <chr>                       <dbl>
1 Flusight-baseline          69149.
2 MOBS-GLEAM_FLUH           -85003.
3 PSI-DICE                   40971.
```

In this simple example with only three models, the ranking of the importance scores remains unchanged. However, in more complex scenarios with a larger number of models, the choice of `na_action` can impact the importance scores and their interpretation. More extensive application can be found in Kim *et al.* (2024).

5.2. Evaluation using trained ensemble in LASOMO algorithm for quantile-based forecasts

TO DO

6. Summary and discussion

Multi-model ensemble forecasts often provide better accuracy and robustness than single models, and are widely used in decision-making and policy planning across various domains. The contribution of each component model to the accuracy of the ensemble depends on its own unique characteristics. The **modelimportance** package enables the quantification of the value that each component model adds to the ensemble performance in different evaluation contexts.

The primary function of the package is `model_importance()`, which returns a data frame with component models and their importance metrics. Users can choose the various ensemble methods to apply and model importance algorithm between LOMO and LASOMO. Additionally, customizable options are available for handling missing values. These features enable the package to serve as a versatile tool to aid collaborative efforts to construct an effective ensemble model across a wide range of forecasting tasks.

There is a room to enhance the current version of this package. Although this package supports four different output types ('mean', 'median', 'quantile, and 'pmf'), other output types are widely used in practice. For example, 'sample' output type is commonly used in the US Flu Scenario Modeling Hub (Flu Scenario Modeling Hub 2024). This format includes multiple simulated values (samples) from the forecast distribution. The `output_type_id` is specified for each sample, which typically indexes the samples or indicates their source, depending on the context. The package can be extended to support this output type, which is under consideration for future releases. These extensions would aim to broaden the scope of applications in real-world forecasting tasks.

References

Bosse NI, Gruson H, Cori A, van Leeuwen E, Funk S, Abbott S (2022). "Evaluating forecasts with scoringutils in R." *arXiv preprint arXiv:2205.07090*.

Bracher J, Ray EL, Gneiting T, Reich NG (2021). "Evaluating epidemic forecasts in an interval format." *PLOS computational biology*, **17**(2), e1008618. `doi:10.1371/journal.pcbi.1008618`.

Chen T, He T, Benesty M, Khotilovich V, Tang Y, Cho H, Chen K, Mitchell R, Cano I, Zhou T, Li M, Xie J, Lin M, Geng Y, Li Y, Yuan J (2024). *xgboost: Extreme Gradient Boosting*. R package version 1.7.7.1, URL https://CRAN.R-project.org/package=xgboost.

Consortium of Infectious Disease Modeling Hubs (2024). "The hubverse: open tools for collaborative forecasting." Accessed: 2025-01-14, URL https://hubverse.io/en/latest/index.html.

Flu Scenario Modeling Hub (2024). "Flu Scenario Modeling Hub." Accessed: 2025-07-25, URL https://fluscenariomodelinghub.org/index.html.

Gneiting T, Raftery AE (2005). "Weather forecasting with ensemble methods." *Science*, **310**(5746), 248–249.

Gneiting T, Raftery AE (2007). "Strictly Proper Scoring Rules, Prediction, and Estimation." *Journal of the American Statistical Association*, **102**(477), 359–378. `doi:10.1198/016214506000001437`.

Hamner B, Frasco M (2018). *Metrics: Evaluation Metrics for Machine Learning*. R package version 0.1.4, URL https://CRAN.R-project.org/package=Metrics.

Hastie T, Tibshirani R, Friedman J (2001). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

He K, Yang Q, Ji L, Pan J, Zou Y (2023). "Financial Time Series Forecasting with the Deep Learning Ensemble Model." *Mathematics*, **11**(4). `doi:10.3390/math11041054`.

Hyndman RJ, Khandakar Y (2008). "Automatic time series forecasting: the forecast package for R." *Journal of Statistical Software*, **27**(3), 1–22. doi:10.18637/jss.v027.i03.

Jordan A, Krüger F, Lerch S (2019). "Evaluating Probabilistic Forecasts with scoringRules." *Journal of Statistical Software*, **90**(12), 1–37. doi:10.18637/jss.v090.i12.

Jordan A Guerra, Sophie A Murray, D Shaun Bloomfield, Peter T Gallagher (2020). "Ensemble forecasting of major solar flares: methods for combining models." *J. Space Weather Space Clim.*, **10**, 38. doi:10.1051/swsc/2020042.

Kim M, Ray EL, Reich NG (2024). "Beyond forecast leaderboards: Measuring individual model importance based on contribution to ensemble accuracy." URL https://arxiv.org/abs/2412.08916.

Kuhn M (2008). "Building Predictive Models in R Using the caret Package." *Journal of Statistical Software*, **28**(5), 1–26. doi:10.18637/jss.v028.i05. URL https://www.jstatsoft.org/index.php/jss/article/view/v028i05.

Liaw A, Wiener M (2002). "Classification and Regression by randomForest." *R News*, **2**(3), 18–22. URL https://CRAN.R-project.org/doc/Rnews/.

Lutz CS, Huynh MP, Schroeder M, Anyatonwu S, Dahlgren FS, Danyluk G, Fernandez D, Greene SK, Kipshidze N, Liu L, Mgbere O, McHugh LA, Myers JF, Siniscalchi A, Sullivan AD, West N, Johansson MA, Biggerstaff M (2019). "Applying infectious disease forecasting to public health: a path forward using influenza forecasting examples." *BMC Public Health*, **19**(1), 1659. doi:10.1186/s12889-019-7966-8.

Ray EL, Reich NG (2018). "Prediction of infectious disease epidemics via weighted density ensembles." *PLOS computational biology*, **14**(2), e1005910. doi:10.1371/journal.pcbi.1005910.

Reich NG, McGowan CJ, Yamana TK, Tushar A, Ray EL, Osthus D, Kandula S, Brooks LC, Crawford-Crudell W, Gibson GC, Moore E, Silva R, Biggerstaff M, Johansson MA, Rosenfeld R, Shaman J (2019). "Accuracy of real-time multi-model ensemble forecasts for seasonal influenza in the U.S." *PLOS computational biology*, **15**(11), e1007486. doi:10.1371/journal.pcbi.1007486.

Ridgeway G, Developers G (2024). *gbm: Generalized Boosted Regression Models*. R package version 2.2.2, URL https://CRAN.R-project.org/package=gbm.

Shandross L, Howerton E, Contamin L, Hochheiser H, Krystalli A, Reich NG, Ray EL, *et al.* (2024). "hubEnsembles: Ensembling Methods in R." *medRxiv*. doi:10.1101/2024.06.24.24309416. https://www.medrxiv.org/content/early/2024/06/25/2024.06.24.24309416.full.pdf.

Sun S, Wang S, Wei Y (2020). "A new ensemble deep learning approach for exchange rates forecasting and trading." *Advanced Engineering Informatics*, **46**, 101160. doi:https://doi.org/10.1016/j.aei.2020.101160.

Viboud C, Sun K, Gaffey R, Ajelli M, Fumanelli L, Merler S, Zhang Q, Chowell G, Simonsen L, Vespignani A (2018). "The RAPIDD ebola forecasting challenge: Synthesis and lessons learnt." *Epidemics*, **22**, 13–21. doi:10.1016/j.epidem.2017.08.002.

Yan Y (2024). *MLmetrics: Machine Learning Evaluation Metrics*. R package version 1.1.3, URL https://CRAN.R-project.org/package=MLmetrics.

Affiliation:

Minsu Kim
E-mail: minsu@umass.edu

Evan Rays

Nicholas G. Reich

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd