

Package: accuracylevel (via r-universe)

June 18, 2026

Title Robust Accuracy-Level Metrics for Predictive Model Evaluation

Version 0.1.0

Author Achmad Syahrul Choir [cre, aut], Mety Agustini [aut], Kartika Fithriasari [aut], Dedy Dwi Prastyo [aut]

Maintainer Achmad Syahrul Choir <madsyair@stis.ac.id>

Description Implements novel accuracy-level metrics for evaluating continuous data prediction models. Four metrics are provided: Counted Squared Error (CSE), Counted Absolute Error (CAE), Counted Absolute Percentage Error (CAPE), and Symmetric Counted Absolute Percentage Error (SCAPE). These metrics offer robust, consistent, and interpretable evaluation on a 0-100% scale, addressing limitations of conventional metrics like RMSE, MAE, and MAPE. The package integrates with 'caret', 'tidymodels', and common forecasting frameworks. Based on Agustini, Fithriasari, and Prastyo (2026)
<doi:10.1016/j.dajour.2025.100661>.

License GPL-3

Encoding UTF-8

URL <https://github.com/madsyair/accuracylevel>

BugReports <https://github.com/madsyair/accuracylevel/issues>

Depends R (>= 3.5.0)

Imports stats, graphics, utils

Suggests rlang (>= 0.4.0), caret, yardstick, forecast, testthat (>= 3.0.0), knitr, rmarkdown, ggplot2, tibble, dplyr

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Config/roxygen2/version 8.0.0

Repository <https://cran.r-universe.dev>

Date/Publication 2026-06-18 13:20:02 UTC

RemoteUrl <https://github.com/cran/accuracylevel>

RemoteRef HEAD

RemoteSha 182c4a56e183ad6cd45ef56a009b0206ec0b07fc

Contents

absolute_error	3
absolute_percentage_error	3
accuracy_level	4
accuracy_level_metrics	6
al_compare_forecasts	7
al_extended_accuracy	8
al_forecast_accuracy	8
al_metric_set	9
al_tsCV	10
auto_threshold	11
cae	12
cae_l1	13
calculate_threshold	13
cape	15
cape_l1	16
caret_single_metric	16
caret_summary	17
caret_summary_extended	18
compare_all_metrics	18
compare_models	19
conventional_metrics	20
cse	21
cse_l1	22
get_all_levels	23
print.al_threshold	24
robust_metrics	24
scape	25
scape_l1	26
squared_error	27
symmetric_absolute_percentage_error	27

Index

29

absolute_error	<i>Calculate Absolute Error</i>
----------------	---------------------------------

Description

Compute absolute error between actual and predicted values, as defined in Equation (2) of Agustini et al. (2026).

Usage

```
absolute_error(actual, predicted, na.rm = FALSE)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
na.rm	Logical. If TRUE, remove NA pairs before computing. Default is FALSE.

Value

Numeric vector of absolute errors.

Examples

```
actual <- c(10, 20, 30, 40, 50)
predicted <- c(11, 19, 32, 38, 51)
absolute_error(actual, predicted)
```

absolute_percentage_error	<i>Calculate Absolute Percentage Error</i>
---------------------------	--

Description

Compute absolute percentage error between actual and predicted values, as defined in Equation (3) of Agustini et al. (2026). Note that the result is on the proportion scale (not multiplied by 100).

Usage

```
absolute_percentage_error(actual, predicted, na.rm = FALSE)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
na.rm	Logical. If TRUE, remove NA pairs before computing. Default is FALSE.

Value

Numeric vector of absolute percentage errors. Returns Inf for observations where actual is zero.

Examples

```
actual <- c(10, 20, 30, 40, 50)
predicted <- c(11, 19, 32, 38, 51)
absolute_percentage_error(actual, predicted)
```

accuracy_level *Compute Accuracy-Level Metrics*

Description

Calculate accuracy-level metrics (CSE, CAE, CAPE, SCAPE) for evaluating prediction model performance. These metrics assess the proportion of observations falling within predefined error threshold levels, providing a robust and interpretable evaluation on a 0–100 scale.

Usage

```
accuracy_level(
  actual,
  predicted,
  threshold = NULL,
  baseline_actual = NULL,
  baseline_predicted = NULL,
  na.rm = FALSE
)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
threshold	An <code>al_threshold</code> object created by <code>calculate_threshold</code> or <code>auto_threshold</code> . When supplied, the baseline quartiles stored inside this object are used to derive per-error-type thresholds (see Details). If NULL (default), thresholds are automatically determined via <code>auto_threshold</code> .
baseline_actual	Numeric vector of actual values from the baseline model. Used only when <code>threshold</code> is NULL.
baseline_predicted	Numeric vector of predicted values from the baseline model. Used only when <code>threshold</code> is NULL.
na.rm	Logical. If TRUE, remove NA pairs before computing. Default is FALSE.

Details

The accuracy-level method introduces four metrics:

CSE (Counted Squared Error) Proportion of observations within squared error threshold levels.

CAE (Counted Absolute Error) Proportion of observations within absolute error threshold levels.

CAPE (Counted Absolute Percentage Error) Proportion of observations within absolute percentage error threshold levels.

SCAPE (Symmetric Counted Absolute Percentage Error) Proportion of observations within symmetric absolute percentage error threshold levels.

For each metric, four accuracy levels are defined:

- Level 1: $\varepsilon < T$ (highest accuracy)
- Level 2: $T \leq \varepsilon < 2T$
- Level 3: $2T \leq \varepsilon < 5T$
- Level 4: $\varepsilon \geq 5T$ (lowest accuracy)

where T is the base threshold determined from the **baseline** model's error distribution. Crucially, thresholds for CSE, CAE, CAPE, and SCAPE are each derived from the same quartile of the baseline model's SE, AE, APE, and sAPE respectively (Figure 2 of the paper).

Edge cases are handled as follows: observations with a non-finite error (for example APE when actual is zero, or sAPE when both actual and predicted are zero) are assigned to Level 4. When the baseline threshold T equals zero (a perfectly fitting baseline), a machine-epsilon boundary is used so that exact-zero errors fall into Level 1.

Value

An object of class "accuracy_level" with elements: `metrics` (data frame with accuracy percentages for each level and metric type), `mean_errors` (data frame with mean errors per level), `threshold` (the primary threshold object used), `thresholds_all` (per-error-type threshold objects), `n_obs` (number of observations), and `counts` (count of observations at each level).

References

Agustini, M., Fithriasari, K., & Prastyo, D.D. (2026). An accuracy-level method for robust evaluation in predictive analytics. *Decision Analytics Journal*, 18, 100661. doi:10.1016/j.dajour.2025.100661

See Also

[calculate_threshold](#), [auto_threshold](#), [cse](#), [cae](#), [cape](#), [scape](#)

Examples

```
# ---- Paper Table 4: Simple case ----
actual <- c(7, 6.03, 2.02, 5.1, 9, 1, 3, 4.38, 1, 8.07)
m1 <- c(6.05, 5.02, 1.32, 5.15, 8, 2.2, 2.7, 3.48, 1, 7.56)
m3 <- c(7.01, 6.04, 2.09, 5.11, 9.01, 5.1, 3.01, 4.39, 1, 8.1)

# Model 1 as baseline, Q2
```

```
thresh <- calculate_threshold(actual, m1, quartile = 2)

# Evaluate Model 3 (paper expects 90% at L1)
result <- accuracy_level(actual, m3, threshold = thresh)
print(result)
```

accuracy_level_metrics

Full Accuracy-Level Metrics for yardstick

Description

Full Accuracy-Level Metrics for yardstick

Usage

```
accuracy_level_metrics(data, truth, estimate, na_rm = TRUE, ...)

## S3 method for class 'data.frame'
accuracy_level_metrics(data, truth, estimate, na_rm = TRUE, ...)
```

Arguments

data	A data frame containing truth and estimate columns.
truth	Column name for actual values (unquoted).
estimate	Column name for predicted values (unquoted).
na_rm	Logical. Remove NAs? Default TRUE.
...	Additional arguments (ignored).

Value

A tibble with 16 rows (4 metrics x 4 levels).

Examples

```
if (requireNamespace("rlang", quietly = TRUE)) {
  df <- data.frame(truth = c(10, 20, 30, 40, 50),
                  estimate = c(11, 19, 32, 38, 51))
  accuracy_level_metrics(df, truth, estimate)
}
```

al_compare_forecasts *Compare Multiple Forecast Models*

Description

Compare Multiple Forecast Models

Usage

```
al_compare_forecasts(  
  ...,  
  test = NULL,  
  metric = c("cae", "cape", "cse", "scape"),  
  threshold = NULL  
)
```

Arguments

...	Named forecast objects or named lists with forecast and test elements.
test	Test data (used when forecast objects are supplied directly).
metric	Metric for determining the optimal model.
threshold	Shared al_threshold object or NULL.

Value

A list with optimal_model, comparison table, and full_results.

Examples

```
actual <- c(10, 20, 30, 40, 50)  
res <- al_compare_forecasts(  
  A = list(forecast = c(11, 19, 32, 38, 51), test = actual),  
  B = list(forecast = c(15, 25, 35, 45, 55), test = actual)  
)  
res$comparison
```

al_extended_accuracy *Extended Forecast Accuracy Summary*

Description

Extended Forecast Accuracy Summary

Usage

```
al_extended_accuracy(forecast_obj, test, threshold = NULL)
```

Arguments

forecast_obj A forecast object or numeric predictions.
test Actual test values.
threshold Optional threshold object.

Value

A data frame combining traditional and accuracy-level metrics.

Examples

```
pred <- c(11, 19, 32, 38, 51)
actual <- c(10, 20, 30, 40, 50)
al_extended_accuracy(pred, actual)
```

al_forecast_accuracy *Accuracy-Level Metrics for Forecast Objects*

Description

Calculate accuracy-level metrics for forecast objects from the **forecast** package, or for plain numeric predictions.

Usage

```
al_forecast_accuracy(object, test, threshold = NULL)
```

```
## Default S3 method:
```

```
al_forecast_accuracy(object, test, threshold = NULL)
```

```
## S3 method for class 'forecast'
```

```
al_forecast_accuracy(object, test, threshold = NULL)
```

Arguments

object	A forecast object or numeric vector of predictions.
test	Numeric vector or time series of test (actual) values.
threshold	An al_threshold object or NULL.

Value

An accuracy_level object.

Examples

```
# With plain numeric vectors
pred <- c(11, 19, 32, 38, 51)
actual <- c(10, 20, 30, 40, 50)
al_forecast_accuracy(pred, actual)
```

al_metric_set *Create Metric Set for tidymodels*

Description

Create Metric Set for tidymodels

Usage

```
al_metric_set(include_traditional = TRUE)
```

Arguments

include_traditional
Logical. If TRUE (default), also include rmse, mae, and rsq from **yardstick**.

Value

A metric set function.

Note

The level-1 metrics in the set derive their error thresholds from the data being evaluated (a self-referential baseline). In resampling or cross-validation, each fold therefore uses its own threshold, which limits strict cross-fold comparability. For a fixed baseline across folds, evaluate with [accuracy_level](#) using a pre-computed al_threshold object (see [calculate_threshold](#)). Case weights are not supported; any case_weights passed by **tune** are ignored.

Examples

```
if (requireNamespace("yardstick", quietly = TRUE)) {
  al_metrics <- al_metric_set()
}
```

al_tsCV

Time Series Cross-Validation with Accuracy-Level Metrics

Description

Time Series Cross-Validation with Accuracy-Level Metrics

Usage

```
al_tsCV(
  y,
  forecastfunction,
  h = 1,
  initial = 10,
  window = NULL,
  metric = c("cae", "cape", "cse", "scape"),
  ...
)
```

Arguments

y	Numeric time series data.
forecastfunction	Function that accepts (x, h, ...) and returns predictions (either a forecast object or numeric vector).
h	Forecast horizon.
initial	Initial training window size.
window	Rolling window size (NULL for expanding window).
metric	Metric to report.
...	Additional arguments passed to forecastfunction.

Value

A data frame of class "al_tsCV" with per-fold results.

Examples

```
ma_fc <- function(x, h) rep(mean(x), h)
y <- sin(seq(0, 4 * pi, length.out = 50)) * 10 + 50
res <- al_tsCV(y, ma_fc, h = 5, initial = 20)
print(res)
```

auto_threshold

Automatic Threshold Selection

Description

Automatically select the best quartile for threshold calculation based on the absolute percentage error approaching a target value (default 0.1), following the recommendation in Section 3.4.5 of Agustini et al. (2026).

Usage

```
auto_threshold(
  actual,
  predicted,
  target_ape = 0.1,
  error_type = "ape",
  multipliers = c(2, 5)
)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
target_ape	Target APE value for threshold selection. Default is 0.1 (10 percent).
error_type	Error type to calculate threshold for. Default is "ape".
multipliers	Numeric vector of multipliers. Default is c(2, 5).

Value

An `al_threshold` object with automatically selected quartile.

Examples

```
actual <- c(10, 20, 30, 40, 50, 60, 70, 80, 90, 100)
predicted <- c(11, 19, 32, 38, 51, 58, 72, 78, 92, 98)
thresh <- auto_threshold(actual, predicted)
print(thresh)
```

cae *Counted Absolute Error (CAE)*

Description

Counted Absolute Error (CAE)

Usage

```
cae(  
  actual,  
  predicted,  
  level = 1,  
  threshold = NULL,  
  baseline_actual = NULL,  
  baseline_predicted = NULL,  
  as_decimal = FALSE  
)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
level	Integer (1–4). Level to return. Default is 1.
threshold	An <code>al_threshold</code> object or NULL for automatic calculation.
baseline_actual	Actual values for baseline model threshold calculation (used only if <code>threshold</code> is NULL).
baseline_predicted	Predicted values for baseline model threshold calculation (used only if <code>threshold</code> is NULL).
as_decimal	Logical. If TRUE, return as proportion (0–1); if FALSE (default), return as percentage (0–100).

Value

Numeric scalar.

Examples

```
actual <- c(10, 20, 30, 40, 50)  
predicted <- c(11, 19, 32, 38, 51)  
cae(actual, predicted, level = 1)
```

cae_l1	<i>CAE Level 1 Metric for yardstick</i>
--------	---

Description

CAE Level 1 Metric for yardstick

Usage

```
cae_l1(data, truth, estimate, na_rm = TRUE, ...)

## S3 method for class 'data.frame'
cae_l1(data, truth, estimate, na_rm = TRUE, ...)
```

Arguments

data	A data frame containing truth and estimate columns.
truth	Column name for actual values (unquoted).
estimate	Column name for predicted values (unquoted).
na_rm	Logical. Remove NAs? Default TRUE.
...	Additional arguments (ignored).

Value

A tibble.

calculate_threshold	<i>Calculate Error Thresholds from a Baseline Model</i>
---------------------	---

Description

Calculate error threshold levels based on a baseline model's error distribution. The threshold is determined using quartiles of the specified error type, following the procedure in Figure 2 of Agustini et al. (2026).

Quartiles are computed using the inverse empirical CDF (R's `type = 1`), consistent with the paper.

Usage

```
calculate_threshold(
  actual,
  predicted,
  error_type = c("ape", "sape", "se", "ae"),
  quartile = 2,
  multipliers = c(2, 5)
)
```

Arguments

actual	Numeric vector of actual (observed) values from the baseline model.
predicted	Numeric vector of predicted values from the baseline model.
error_type	Character string specifying the error type used to select the quartile. One of "ape" (default, absolute percentage error), "sape" (symmetric APE), "se" (squared error), or "ae" (absolute error).
quartile	Integer (1, 2, or 3) specifying which quartile to use. Default is 2 (median). The recommended approach from the paper is to select the quartile where the APE value is close to 0.1 (10 percent error).
multipliers	Numeric vector of length 2 specifying the multipliers for level boundaries. Default is c(2, 5), creating levels L1: error < T, L2: T <= error < 2T, L3: 2T <= error < 5T, L4: error >= 5T.

Value

A list of class "al_threshold" with elements: threshold (the base threshold value T), levels (a list with L1–L4 boundary pairs), error_type (the error type used), quartile (the quartile used), multipliers (the multiplier values), and baseline_quartiles (a named list with the selected quartile value for every error type: se, ae, ape, sape – this is the key element that lets [accuracy_level](#) derive thresholds for all four metrics from a single baseline model).

References

Agustini, M., Fithriasari, K., & Prastyo, D.D. (2026). An accuracy-level method for robust evaluation in predictive analytics. *Decision Analytics Journal*, 18, 100661. doi:10.1016/j.dajour.2025.100661

See Also

[accuracy_level](#), [auto_threshold](#)

Examples

```
# --- Paper Table 4: simple case, Model 1 as baseline ---
actual <- c(7, 6.03, 2.02, 5.1, 9, 1, 3, 4.38, 1, 8.07)
model1 <- c(6.05, 5.02, 1.32, 5.15, 8, 2.2, 2.7, 3.48, 1, 7.56)

# Q2 of APE ~ 0.1111, close to the target 0.10
thresh <- calculate_threshold(actual, model1, quartile = 2)
print(thresh)

# Stricter thresholds via Q1
thresh_q1 <- calculate_threshold(actual, model1, quartile = 1)
```

cape	<i>Counted Absolute Percentage Error (CAPE)</i>
------	---

Description

Counted Absolute Percentage Error (CAPE)

Usage

```
cape(  
  actual,  
  predicted,  
  level = 1,  
  threshold = NULL,  
  baseline_actual = NULL,  
  baseline_predicted = NULL,  
  as_decimal = FALSE  
)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
level	Integer (1–4). Level to return. Default is 1.
threshold	An <code>al_threshold</code> object or NULL for automatic calculation.
baseline_actual	Actual values for baseline model threshold calculation (used only if <code>threshold</code> is NULL).
baseline_predicted	Predicted values for baseline model threshold calculation (used only if <code>threshold</code> is NULL).
as_decimal	Logical. If TRUE, return as proportion (0–1); if FALSE (default), return as percentage (0–100).

Value

Numeric scalar.

Examples

```
actual <- c(10, 20, 30, 40, 50)  
predicted <- c(11, 19, 32, 38, 51)  
cape(actual, predicted, level = 1)
```

cape_l1	<i>CAPE Level 1 Metric for yardstick</i>
---------	--

Description

CAPE Level 1 Metric for yardstick

Usage

```
cape_l1(data, truth, estimate, na_rm = TRUE, ...)
```

```
## S3 method for class 'data.frame'
cape_l1(data, truth, estimate, na_rm = TRUE, ...)
```

Arguments

data	A data frame containing truth and estimate columns.
truth	Column name for actual values (unquoted).
estimate	Column name for predicted values (unquoted).
na_rm	Logical. Remove NAs? Default TRUE.
...	Additional arguments (ignored).

Value

A tibble.

caret_single_metric	<i>Create Single Metric caret Summary</i>
---------------------	---

Description

Create Single Metric caret Summary

Usage

```
caret_single_metric(
  metric_type = c("cse", "cae", "cape", "scape"),
  level = 1,
  threshold = NULL
)
```

Arguments

metric_type	One of "cse", "cae", "cape", "scape".
level	Level to optimise (1–4). Default is 1.
threshold	An al_threshold object or NULL.

Value

A function suitable for summaryFunction in caret::trainControl.

Examples

```
if (requireNamespace("caret", quietly = TRUE)) {  
  fn <- caret_single_metric("cae", level = 1)  
}
```

caret_summary	<i>Create Custom caret Metrics</i>
---------------	------------------------------------

Description

Create a summary function for use with caret's trainControl. Returns L1 accuracy for all four metrics plus traditional metrics.

Usage

```
caret_summary(threshold = NULL)
```

Arguments

threshold An optional al_threshold object for consistent thresholds across folds. If NULL, thresholds are calculated per fold.

Value

A function suitable for summaryFunction in caret::trainControl.

Examples

```
if (requireNamespace("caret", quietly = TRUE)) {  
  al_summary <- caret_summary()  
}
```

`caret_summary_extended`*Create Extended caret Summary with All Levels*

Description

Create Extended caret Summary with All Levels

Usage

```
caret_summary_extended(threshold = NULL)
```

Arguments

`threshold` An optional `al_threshold` object for consistent thresholds across folds. If `NULL`, thresholds are calculated per fold.

Value

A function suitable for `summaryFunction` in `caret::trainControl`.

Examples

```
if (requireNamespace("caret", quietly = TRUE)) {  
  al_ext <- caret_summary_extended()  
}
```

`compare_all_metrics`*Compare All Metric Types*

Description

Comprehensive comparison of conventional, robust, and accuracy-level metrics.

Usage

```
compare_all_metrics(actual, predicted, threshold = NULL)
```

Arguments

`actual` Numeric vector of actual values.
`predicted` Numeric vector of predicted values.
`threshold` Threshold object for accuracy-level metrics.

Value

A list of class "metrics_comparison" with conventional, robust, accuracy_level, and summary elements.

Examples

```
actual <- c(10, 20, 30, 40, 50, 60, 70, 80, 90, 100)
predicted <- c(11, 19, 32, 38, 51, 58, 72, 78, 92, 98)
result <- compare_all_metrics(actual, predicted)
print(result)
```

compare_models

Compare Multiple Models

Description

Compare multiple prediction models using accuracy-level metrics and identify the optimal one following the model-selection procedure in Figure 3 of Agustini et al. (2026).

Usage

```
compare_models(
  ...,
  metric = c("cse", "cae", "cape", "scape"),
  threshold = NULL
)
```

Arguments

...	Named arguments, each a list with actual and predicted elements.
metric	Metric for comparison. Default is "cse".
threshold	Shared al_threshold object. When NULL (default), the <i>first</i> model is used as the baseline.

Details

The optimal model is selected by the Figure 3 algorithm:

1. Compare the Level 1 accuracy across models; the model with the highest value is selected.
2. If two or more models tie on Level 1 accuracy, the tie is broken using the mean error (ME) of the corresponding level (lower ME is better).
3. If the ME values are also equal, the comparison proceeds to the next accuracy level, repeating until the optimal model is identified.

Earlier releases used the simpler rule of ranking by Level 1 then Level 2 accuracy; this version implements the full ME-based tie-break.

Value

A list with `optimal_model`, comparison table (accuracy and mean error per level), `metric_used`, and `full_results`.

Examples

```
actual <- c(7, 6.03, 2.02, 5.1, 9, 1, 3, 4.38, 1, 8.07)
m1 <- list(actual = actual,
           predicted = c(6.05, 5.02, 1.32, 5.15, 8, 2.2, 2.7, 3.48, 1, 7.56))
m3 <- list(actual = actual,
           predicted = c(7.01, 6.04, 2.09, 5.11, 9.01, 5.1, 3.01, 4.39, 1, 8.1))

res <- compare_models(Model1 = m1, Model3 = m3, metric = "cape")
print(res$comparison)
res$optimal_model
```

conventional_metrics *Calculate Conventional Metrics*

Description

Compute commonly used evaluation metrics including R-squared, RMSE, NRMSE, MAE, MAPE, and SMAPE.

Usage

```
conventional_metrics(actual, predicted, na.rm = FALSE)
```

Arguments

<code>actual</code>	Numeric vector of actual (observed) values.
<code>predicted</code>	Numeric vector of predicted values.
<code>na.rm</code>	Logical. If TRUE, remove NA pairs before computing. Default is FALSE.

Details

NRMSE is normalised by the mean of actual (returned as NA when that mean is zero). `R_squared` is the usual $1 - SS_{res}/SS_{tot}$ and may be negative for models worse than the mean (unlike the 0–1 range quoted in Table 1 of the paper). MAPE and SMAPE are returned on the percentage scale and ignore non-finite per-observation terms (e.g. division by a zero actual value).

Value

A named numeric vector with six elements.

Examples

```
actual <- c(10, 20, 30, 40, 50)
predicted <- c(11, 19, 32, 38, 51)
conventional_metrics(actual, predicted)
```

cse

Counted Squared Error (CSE)

Description

Compute the Counted Squared Error metric at a specified level.

Usage

```
cse(
  actual,
  predicted,
  level = 1,
  threshold = NULL,
  baseline_actual = NULL,
  baseline_predicted = NULL,
  as_decimal = FALSE
)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
level	Integer (1–4). Level to return. Default is 1.
threshold	An <code>al_threshold</code> object or NULL for automatic calculation.
baseline_actual	Actual values for baseline model threshold calculation (used only if <code>threshold</code> is NULL).
baseline_predicted	Predicted values for baseline model threshold calculation (used only if <code>threshold</code> is NULL).
as_decimal	Logical. If TRUE, return as proportion (0–1); if FALSE (default), return as percentage (0–100).

Value

Numeric scalar: the percentage (or proportion) of observations at the specified accuracy level.

Examples

```
actual <- c(7, 6.03, 2.02, 5.1, 9, 1, 3, 4.38, 1, 8.07)
predicted <- c(6.05, 5.02, 1.32, 5.15, 8, 2.2, 2.7, 3.48, 1, 7.56)

cse(actual, predicted, level = 1)
sapply(1:4, function(l) cse(actual, predicted, level = 1))
```

cse_l1

CSE Level 1 Metric for yardstick

Description

CSE Level 1 Metric for yardstick

Usage

```
cse_l1(data, truth, estimate, na_rm = TRUE, ...)

## S3 method for class 'data.frame'
cse_l1(data, truth, estimate, na_rm = TRUE, ...)
```

Arguments

data	A data frame containing truth and estimate columns.
truth	Column name for actual values (unquoted).
estimate	Column name for predicted values (unquoted).
na_rm	Logical. Remove NAs? Default TRUE.
...	Additional arguments (ignored).

Value

A tibble with `.metric`, `.estimator`, `.estimate`.

Examples

```
if (requireNamespace("rlang", quietly = TRUE)) {
  df <- data.frame(truth = c(10, 20, 30), estimate = c(11, 19, 28))
  cse_l1(df, truth, estimate)
}
```

get_all_levels	<i>Get All Levels for a Metric</i>
----------------	------------------------------------

Description

Convenience function to obtain all four levels at once.

Usage

```
get_all_levels(  
  actual,  
  predicted,  
  metric = c("cse", "cae", "cape", "scape"),  
  threshold = NULL,  
  baseline_actual = NULL,  
  baseline_predicted = NULL  
)
```

Arguments

actual	Numeric vector of actual values.
predicted	Numeric vector of predicted values.
metric	One of "cse", "cae", "cape", "scape".
threshold	An al_threshold object or NULL.
baseline_actual	Actual values for baseline model.
baseline_predicted	Predicted values for baseline model.

Value

Named numeric vector of length 4.

Examples

```
actual <- c(10, 20, 30, 40, 50)  
predicted <- c(11, 19, 32, 38, 51)  
get_all_levels(actual, predicted, "cae")
```

`print.al_threshold` *Print Method for al_threshold Objects*

Description

Print Method for `al_threshold` Objects

Usage

```
## S3 method for class 'al_threshold'  
print(x, ...)
```

Arguments

`x` An `al_threshold` object.
`...` Additional arguments (ignored).

Value

Invisibly returns the input object.

`robust_metrics` *Calculate Robust Metrics*

Description

Compute robust evaluation metrics including Median Absolute Error, Trimmed Mean Squared Error, Huber Loss, and Quantile Loss.

Usage

```
robust_metrics(  
  actual,  
  predicted,  
  trim_percent = 0.1,  
  huber_delta = 1,  
  tau = 0.5,  
  na.rm = FALSE  
)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
trim_percent	Proportion to trim for TMSE (default 0.1).
huber_delta	Delta parameter for Huber loss (default 1).
tau	Quantile for quantile loss (default 0.5 for median).
na.rm	Logical. Default is FALSE.

Value

A named numeric vector with four elements.

Examples

```
actual <- c(10, 20, 30, 40, 50, 1000)
predicted <- c(11, 19, 32, 38, 51, 60)
robust_metrics(actual, predicted)
```

scape

Symmetric Counted Absolute Percentage Error (SCAPE)

Description

Symmetric Counted Absolute Percentage Error (SCAPE)

Usage

```
scape(
  actual,
  predicted,
  level = 1,
  threshold = NULL,
  baseline_actual = NULL,
  baseline_predicted = NULL,
  as_decimal = FALSE
)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
level	Integer (1–4). Level to return. Default is 1.
threshold	An <code>al_threshold</code> object or NULL for automatic calculation.

baseline_actual	Actual values for baseline model threshold calculation (used only if threshold is NULL).
baseline_predicted	Predicted values for baseline model threshold calculation (used only if threshold is NULL).
as_decimal	Logical. If TRUE, return as proportion (0–1); if FALSE (default), return as percentage (0–100).

Value

Numeric scalar.

Examples

```
actual <- c(10, 20, 30, 40, 50)
predicted <- c(11, 19, 32, 38, 51)
scape(actual, predicted, level = 1)
```

scape_l1

SCAPE Level 1 Metric for yardstick

Description

SCAPE Level 1 Metric for yardstick

Usage

```
scape_l1(data, truth, estimate, na_rm = TRUE, ...)
```

```
## S3 method for class 'data.frame'
```

```
scape_l1(data, truth, estimate, na_rm = TRUE, ...)
```

Arguments

data	A data frame containing truth and estimate columns.
truth	Column name for actual values (unquoted).
estimate	Column name for predicted values (unquoted).
na_rm	Logical. Remove NAs? Default TRUE.
...	Additional arguments (ignored).

Value

A tibble.

squared_error	<i>Calculate Squared Error</i>
---------------	--------------------------------

Description

Compute squared error between actual and predicted values, as defined in Equation (1) of Agustini et al. (2026).

Usage

```
squared_error(actual, predicted, na.rm = FALSE)
```

Arguments

actual	Numeric vector of actual (observed) values.
predicted	Numeric vector of predicted values.
na.rm	Logical. If TRUE, remove NA pairs before computing. Default is FALSE.

Value

Numeric vector of squared errors.

References

Agustini, M., Fithriasari, K., & Prastyo, D.D. (2026). An accuracy-level method for robust evaluation in predictive analytics. *Decision Analytics Journal*, 18, 100661. doi:10.1016/j.dajour.2025.100661

Examples

```
actual <- c(10, 20, 30, 40, 50)
predicted <- c(11, 19, 32, 38, 51)
squared_error(actual, predicted)
```

symmetric_absolute_percentage_error	<i>Calculate Symmetric Absolute Percentage Error</i>
-------------------------------------	--

Description

Compute symmetric absolute percentage error between actual and predicted values, as defined in Equation (4) of Agustini et al. (2026). Note that the result is on the proportion scale (not multiplied by 100).

Usage

```
symmetric_absolute_percentage_error(actual, predicted, na.rm = FALSE)
```

Arguments

<code>actual</code>	Numeric vector of actual (observed) values.
<code>predicted</code>	Numeric vector of predicted values.
<code>na.rm</code>	Logical. If TRUE, remove NA pairs before computing. Default is FALSE.

Value

Numeric vector of symmetric absolute percentage errors. Returns NaN when both actual and predicted are zero.

Examples

```
actual <- c(10, 20, 30, 40, 50)
predicted <- c(11, 19, 32, 38, 51)
symmetric_absolute_percentage_error(actual, predicted)
```

Index

absolute_error, 3
absolute_percentage_error, 3
accuracy_level, 4, 9, 14
accuracy_level_metrics, 6
al_compare_forecasts, 7
al_extended_accuracy, 8
al_forecast_accuracy, 8
al_metric_set, 9
al_tsCV, 10
auto_threshold, 4, 5, 11, 14

cae, 5, 12
cae_l1, 13
calculate_threshold, 4, 5, 9, 13
cape, 5, 15
cape_l1, 16
caret_single_metric, 16
caret_summary, 17
caret_summary_extended, 18
compare_all_metrics, 18
compare_models, 19
conventional_metrics, 20
cse, 5, 21
cse_l1, 22

get_all_levels, 23

print.al_threshold, 24

robust_metrics, 24

scape, 5, 25
scape_l1, 26
squared_error, 27
symmetric_absolute_percentage_error,
27