Package 'scoringutils'

June 14, 2020

Title Utilities for Scoring and Assessing Predictions

Version 0.1.0

Description Combines a collection of metrics and proper scoring rules (Tilmann Gneiting & Adrian E Raftery (2007) <doi:10.1198/016214506000001437>) with an easy to use wrapper that can be used to automatically evaluate predictions. Apart from proper scoring rules functions are provided to assess bias, sharpness and calibration (Sebastian Funk, Anton Camacho, Adam J. Kucharski, Rachel Lowe, Rosalind M. Eggo, W. John Edmunds (2019) <doi:10.1371/journal.pcbi.1006785>) of forecasts. Several types of predictions can be evaluated: probabilistic forecasts (generally predictive samples generated by Markov Chain Monte Carlo procedures), quantile forecasts or point forecasts. Observed values and predictions can be either continuous, integer, or binary. Users can either choose to apply these rules separately in a vector / matrix format that can be flexibly used within other packages, or they can choose to do an automatic evaluation of their forecasts. This is implemented with 'data.table' and provides a consistent and very efficient framework for evaluating various types of predictions.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports data.table, goftest, graphics, scoringRules, stats

Suggests testthat, knitr, rmarkdown

RoxygenNote 7.1.0

URL https://github.com/epiforecasts/scoringutils

BugReports https://github.com/epiforecasts/scoringutils/issues

VignetteBuilder knitr

Depends R (>= 2.10)

NeedsCompilation no

Author Nikos Bosse [aut, cre] (<https://orcid.org/0000-0002-7750-5280>),
Sam Abbott [aut] (<https://orcid.org/0000-0001-8057-8037>),
Joel Hellewell [ctb] (<https://orcid.org/0000-0003-2683-0849>),
Sophie Meakins [ctb],
James Munday [ctb],
Katharine Sherratt [ctb],
Sebastian Funk [aut]

Maintainer Nikos Bosse <nikosbosse@gmail.com>

Repository CRAN

Date/Publication 2020-06-14 15:00:03 UTC

R topics documented:

`bias` *Determines bias of forecasts*

Description

Determines bias from predictive Monte-Carlo samples. The function automatically recognises, whether forecasts are continuous or integer valued and adapts the Bias function accordingly.

Usage

```
bias(true_values, predictions)
```

Arguments

true_values	A vector with the true observed values of size n
predictions	nxN matrix of predictive samples, n (number of rows) being the number of data points and N (number of columns) the number of Monte Carlo samples

Details

For continuous forecasts, Bias is measured as

$$B_t(P_t, x_t) = 1 - 2 * (P_t(x_t))$$

where P_t is the empirical cumulative distribution function of the prediction for the true value x_t. Computationally, $P_t(x_t)$ is just calculated as the fraction of predictive samples for x_t that are smaller than x_t.

For integer valued forecasts, Bias is measured as

$$B_t(P_t, x_t) = 1 - (P_t(x_t) + P_t(x_t + 1))$$

to adjust for the integer nature of the forecasts.

In both cases, Bias can assume values between -1 and 1 and is 0 ideally.

Value

vector of length n with the biases of the predictive samples with respect to the true values.

Author(s)

Nikos Bosse <nikosbosse@gmail.com>

References

The integer valued Bias function is discussed in Assessing the performance of real-time epidemic forecasts: A case study of Ebola in the Western Area region of Sierra Leone, 2014-15 Funk S, Camacho A, Kucharski AJ, Lowe R, Eggo RM, et al. (2019) Assessing the performance of real-time epidemic forecasts: A case study of Ebola in the Western Area region of Sierra Leone, 2014-15. PLOS Computational Biology 15(2): e1006785. https://doi.org/10.1371/journal.pcbi.1006785

Examples

```
## integer valued forecasts
true_values <- rpois(30, lambda = 1:30)
predictions <- replicate(200, rpois(n = 30, lambda = 1:30))
bias(true_values, predictions)

## continuous forecasts
true_values <- rnorm(30, mean = 1:30)
predictions <- replicate(200, rnorm(30, mean = 1:30))
```

```
bias(true_values, predictions)
```

`binary_example_data` *Binary Example Data*

Description

A toy dataset for a probability forecast of a binary outcome variable

Usage

```
binary_example_data
```

Format

A data.table with 60 rows and 4 variables:

id unique identifier for true observed values

true_values true observed values

model name of the model that generated the forecasts

predictions predicted probability that the corresponding true value will be 1

`brier_score` *Brier Score*

Description

Computes the Brier Score for probabilistic forecasts of binary outcomes.

Usage

```
brier_score(true_values, predictions)
```

Arguments

`true_values`	A vector with the true observed values of size n
`predictions`	A vector with a predicted probability that true_value = 1.

Details

The Brier score is a proper score rule that assesses the accuracy of probabilistic binary predictions. The outcomes can be either 0 or 1, the predictions must be a probability that the true outcome will be 1.

The Brier Score is then computed as the mean squared error between the probabilistic prediction and the true outcome.

$$Brier_Score = \frac{1}{N}\sum_{t=1}^{n}(prediction_t - outcome_t)^2$$

Value

A numeric value with the Brier Score, i.e. the mean squared error of the given probability forecasts

Examples

```
true_values <- sample(c(0,1), size = 30, replace = TRUE)
predictions <- runif(n = 30, min = 0, max = 1)

brier_score(true_values, predictions)
```

---

continuous_example_data

Continuous Example Data

Description

A toy dataset for a probabilistic forecast of a continuous outcome variable

Usage

```
continuous_example_data
```

Format

A data frame with 3000 rows and 5 variables:

id unique identifier for true observed values

model name of the model that generated the forecasts

true_values true observed values

sample number that identifies the predictive sample generated by a specific model for a specific observed value

predictions predictive sample for the corresponding true value

crps *Ranked Probability Score*

Description

Wrapper around the crps_sample function from the scoringRules package. Can be used for continuous as well as integer valued forecasts

Usage

```
crps(true_values, predictions)
```

Arguments

true_values	A vector with the true observed values of size n
predictions	nxN matrix of predictive samples, n (number of rows) being the number of data points and N (number of columns) the number of Monte Carlo samples

Value

vector with the scoring values

References

Alexander Jordan, Fabian Krüger, Sebastian Lerch, Evaluating Probabilistic Forecasts withscoringRules, https://arxiv.org/pdf/1709.04743.pdf

Examples

```
true_values <- rpois(30, lambda = 1:30)
predictions <- replicate(200, rpois(n = 30, lambda = 1:30))
crps(true_values, predictions)
```

dss *Dawid-Sebastiani Score*

Description

Wrapper around the dss_sample function from the scoringRules package.

Usage

```
dss(true_values, predictions)
```

Arguments

true_values	A vector with the true observed values of size n
predictions	nxN matrix of predictive samples, n (number of rows) being the number of data points and N (number of columns) the number of Monte Carlo samples

Value

vector with scoring values

References

Alexander Jordan, Fabian Krüger, Sebastian Lerch, Evaluating Probabilistic Forecasts withscoringRules, https://arxiv.org/pdf/1709.04743.pdf

Examples

```
true_values <- rpois(30, lambda = 1:30)
predictions <- replicate(200, rpois(n = 30, lambda = 1:30))
dss(true_values, predictions)
```

---

eval_forecasts *Evaluate forecasts*

Description

The function `eval_forecasts` is an easy to use wrapper of the lower level functions in the `scoringutils` package. It can be used to score probabilistic or quantile forecasts of continuous, integer-valued or binary variables.

Usage

```
eval_forecasts(data, summarised = TRUE)
```

Arguments

data	A data.frame or data.table with the following columns: • `true_values` the true observed values • `id` A unique identifier of the true values. Could be a date or just a running index • `predictions` predictions or predictive samples for one true value. Not necessary in case of quantile forecasts • `model` name of the model that generated the predictions • `sample` an index to identify the predictive samples in the predictiions column generated by one model for one true value. Only necessary for continuous and integer forecasts, not for quantile or binary predictions.

- quantile forecasts a number of pairs of columns with quantile predictions for a certain range. For a 50 (corresponding to the 25 75 `upper_50`. For the median, there has to a column `lower_0` and one `upper_0`

`summarised` if `TRUE`, only one average score is returned per model

Details

the following metrics are used where appropriate:

- Interval Score for quantile forecasts. Smaller is better. See `interval_score` for more information.
- Brier Score for a probability forecast of a binary outcome. Smaller is better. See `brier_score` for more information.
- Bias 0 is good, 1 and -1 are bad. See `bias` for more information.
- Sharpness Smaller is better. See `sharpness` for more information.
- Calibration represented through the p-value of the Anderson-Darling test for the uniformity of the Probability Integral Transformation (PIT). For integer valued forecasts, this p-value also has a standard deviation. Larger is better. See `pit` for more information.
- DSS Dawid-Sebastiani-Score. Smaller is better. See `dss` for more information.
- CRPS Continuous Ranked Probability Score. Smaller is better. See `crps` for more information.
- LogS Log Score. Smaller is better. Only for continuous forecasts. See `logs` for more information.

Value

A data.table with appropriate scores. For binary predictions, the Brier Score will be returned, for quantile predictions the interval score. For integer forecasts, Sharpness, Bias, DSS, CRPS, LogS, and pit_p_val (as an indicator of calibration) are returned. For integer forecasts, pit_sd is returned (to account for the randomised PIT), but no Log Score is returned (the internal estimation relies on a kernel density estimate which is difficult for integer-valued forecasts). If `summarised = TRUE` the average score per model is returned.

Author(s)

Nikos Bosse <nikosbosse@gmail.com>

References

Funk S, Camacho A, Kucharski AJ, Lowe R, Eggo RM, Edmunds WJ (2019) Assessing the performance of real-time epidemic forecasts: A case study of Ebola in the Western Area region of Sierra Leone, 2014-15. PLoS Comput Biol 15(2): e1006785. https://doi.org/10.1371/journal.pcbi.1006785

Examples

```
## Probability Forecast for Binary Target
binary_example <- data.table::setDT(scoringutils::binary_example_data)
eval <- scoringutils::eval_forecasts(binary_example)
eval <- scoringutils::eval_forecasts(binary_example, summarised = FALSE)

## Quantile Forecasts
quantile_example <- data.table::setDT(scoringutils::quantile_example_data)
eval <- scoringutils::eval_forecasts(quantile_example)
eval <- scoringutils::eval_forecasts(quantile_example, summarised = FALSE)

## Integer Forecasts
integer_example <- data.table::setDT(scoringutils::integer_example_data)
eval <- scoringutils::eval_forecasts(integer_example)
eval <- scoringutils::eval_forecasts(integer_example, summarised = FALSE)

## Continuous Forecasts
continuous_example <- data.table::setDT(scoringutils::continuous_example_data)
eval <- scoringutils::eval_forecasts(continuous_example)
eval <- scoringutils::eval_forecasts(continuous_example, summarised = FALSE)
```

`hist_PIT` *PIT Histogram*

Description

Make a simple histogram of the probability integral transformed values to visually check whether a uniform distribution seems likely.

Usage

```
hist_PIT(PIT_samples, num_bins = NULL)
```

Arguments

`PIT_samples`	A vector with the PIT values of size n
`num_bins`	the number of bins in the PIT histogram. If not given, the square root of n will be used

Value

vector with the scoring values

integer_example_data *Integer Example Data*

Description

A toy dataset for a probabilistic forecast of an integer outcome variable

Usage

```
integer_example_data
```

Format

A data frame with 3000 rows and 5 variables:

id unique identifier for true observed values

model name of the model that generated the forecasts

true_values true observed values

sample number that identifies the predictive sample generated by a specific model for a specific observed value

predictions predictive sample for the corresponding true value

interval_score *Interval Score*

Description

Proper Scoring Rule to score quantile predictions, following Gneiting and Raftery (2007). Smaller values are better.

The score is computed as

$$score = (upper - lower) + 2/alpha * (lower - true_value) * 1(true_values < lower) + 2/alpha * (true_value - upper) * 1(tru$$

where $1()$ is the indicator function and alpha is the decimal value that indicates how much is outside the prediction interval. To improve usability, the user is asked to provide an interval range in percentage terms, i.e. interval_range = 90 (percent) for a 90 percent prediction interval. Correspondingly, the user would have to provide the 5% and 95% quantiles (the corresponding alpha would then be 0.1). No specific distribution is assumed, but the range has to be symmetric (i.e you can't use the 0.1 quantile as the lower bound and the 0.7 quantile as the upper).

The interval score is a proper scoring rule that scores a quantile forecast

Usage

```
interval_score(true_values, lower, upper, interval_range = NULL, weigh = FALSE)
```

Arguments

true_values	A vector with the true observed values of size n
lower	vector of size n with the lower quantile of the given range
upper	vector of size n with the upper quantile of the given range
interval_range	the range of the prediction intervals. i.e. if you're forecasting the 0.05 and 0.95 quantile, the interval_range would be 90. Can be either a single number or a vector of size n, if the range changes for different forecasts to be scored. This corresponds to (100-alpha)/100 in Gneiting and Raftery (2007). Internally, the range will be transformed to alpha.
weigh	if TRUE, weigh the score by alpha / 4, so it can be averaged into an interval score that, in the limit, corresponds to CRPS. Default: FALSE.

Value

vector with the scoring values

References

Strictly Proper Scoring Rules, Prediction,and Estimation, Tilmann Gneiting and Adrian E. Raftery, 2007, Journal of the American Statistical Association, Volume 102, 2007 - Issue 477

Evaluating epidemic forecasts in an interval format, Johannes Bracher, Evan L. Ray, Tilmann Gneiting and Nicholas G. Reich, <arXiv:2005.12881v1>

Examples

```
true_values <- rnorm(30, mean = 1:30)
interval_range = 90
alpha = (100 - interval_range) / 100
lower = qnorm(alpha/2, rnorm(30, mean = 1:30))
upper = qnorm((1- alpha/2), rnorm(30, mean = 1:30))

interval_score(true_values = true_values,
               lower = lower,
               upper = upper,
               interval_range = interval_range)
```

---

logs *LogS*

Description

Wrapper around the logs_sample function from the scoringRules package. Used to score continuous predictions. While the Log Score is in theory also applicable to integer forecasts, the problem lies in the implementation: The Log Score needs a kernel density estimation, which is not well defined with integer-valued Monte Carlo Samples. The Log Score can be used for specific integer valued probabiliy distributions. See the scoringRules package for more details.

Usage

```
logs(true_values, predictions)
```

Arguments

true_values	A vector with the true observed values of size n
predictions	nxN matrix of predictive samples, n (number of rows) being the number of data points and N (number of columns) the number of Monte Carlo samples

Value

vector with the scoring values

References

Alexander Jordan, Fabian Krüger, Sebastian Lerch, Evaluating Probabilistic Forecasts withscoringRules, https://arxiv.org/pdf/1709.04743.pdf

Examples

```
true_values <- rpois(30, lambda = 1:30)
predictions <- replicate(200, rpois(n = 30, lambda = 1:30))
logs(true_values, predictions)
```

---

mse *Mean Squared Error*

Description

Mean Squared Error MSE of point forecasts. Calculuated as

$$mean((true_values - predicted_values)^2)$$

Usage

```
mse(true_values, predictions)
```

Arguments

true_values	A vector with the true observed values of size n
predictions	A vector with predicted values of size n

Value

vector with the scoring values

Examples

```
true_values <- rnorm(30, mean = 1:30)
predicted_values <- rnorm(30, mean = 1:30)
mse(true_values, predicted_values)
```

`pit` *Probability Integral Transformation*

Description

Uses a Probability Integral Transformation (PIT) (or a randomised PIT for integer forecasts) to assess the calibration of predictive Monte Carlo samples. Returns a p-values resulting from an Anderson-Darling test for uniformity of the (randomised) PIT as well as a PIT histogram if specified.

Usage

```
pit(
  true_values,
  predictions,
  plot = TRUE,
  full_output = FALSE,
  n_replicates = 20,
  num_bins = NULL
)
```

Arguments

`true_values`	A vector with the true observed values of size n
`predictions`	nxN matrix of predictive samples, n (number of rows) being the number of data points and N (number of columns) the number of Monte Carlo samples
`plot`	logical. If TRUE, a histogram of the PIT values will be returned as well
`full_output`	return all individual p_values and computed u_t values for the randomised PIT. Usually not needed.
`n_replicates`	the number of tests to perform, each time re-randomising the PIT
`num_bins`	the number of bins in the PIT histogram (if plot == TRUE) If not given, the square root of n will be used

Details

Calibration or reliability of forecasts is the ability of a model to correctly identify its own uncertainty in making predictions. In a model with perfect calibration, the observed data at each time point look as if they came from the predictive probability distribution at that time.

Equivalently, one can inspect the probability integral transform of the predictive distribution at time t,

$$u_t = F_t(x_t)$$

where x_t is the observed data point at time $tint_1, \ldots, t_n$, n being the number of forecasts, and F_t is the (continuous) predictive cumulative probability distribution at time t. If the true probability distribution of outcomes at time t is G_t then the forecasts eqnF_t are said to be ideal if eqnF_t = G_t at all times t. In that case, the probabilities ut are distributed uniformly.

In the case of discrete outcomes such as incidence counts, the PIT is no longer uniform even when forecasts are ideal. In that case a randomised PIT can be used instead:

$$u_t = P_t(k_t) + v * (P_t(k_t) - P_t(k_t - 1))$$

where k_t is the observed count, $P_t(x)$ is the predictive cumulative probability of observing incidence k at time t, eqnP_t (-1) = 0 by definition and v is standard uniform and independent of k. If P_t is the true cumulative probability distribution, then u_t is standard uniform.

The function checks whether integer or continuous forecasts were provided. It then applies the (randomised) probability integral and tests the values u_t for uniformity using the Anderson-Darling test.

As a rule of thumb, there is no evidence to suggest a forecasting model is miscalibrated if the p-value found was greater than a threshold of p >= 0.1, some evidence that it was miscalibrated if 0.01 < p < 0.1, and good evidence that it was miscalibrated if p <= 0.01. In this context it should be noted, though, that uniformity of the PIT is a necessary but not sufficient condition of calibration.

Value

a list with the following components:

- `p_value`: p-value of the Anderson-Darling test on the PIT values. In case of integer forecasts, this will be the mean p_value from the 'n_replicates' replicates
- `sd`: standard deviation of the p_value returned. In case of continuous forecasts, this will be NA as there is only one p_value returned.
- `hist_PIT` a plot object with the PIT histogram. Only returned if `plot == TRUE`. Call `plot(PIT(...)$hist_PIT)` to display the histogram.
- `p_values`: all p_values generated from the Anderson-Darling tests on the randomised PIT. Only returned for integer forecasts and if `full_output = TRUE`
- `u`: the u_t values internally computed. Only returned if `full_output = TRUE`

References

Sebastian Funk, Anton Camacho, Adam J. Kucharski, Rachel Lowe, Rosalind M. Eggo, W. John Edmunds (2019) Assessing the performance of real-time epidemic forecasts: A case study of Ebola in the Western Area region of Sierra Leone, 2014-15, <doi:10.1371/journal.pcbi.1006785>

Examples

```
## continuous predictions
true_values <- rnorm(30, mean = 1:30)
predictions <- replicate(200, rnorm(n = 30, mean = 1:30))
pit(true_values, predictions)

## integer predictions
true_values <- rpois(100, lambda = 1:100)
predictions <- replicate(5000, rpois(n = 100, lambda = 1:100))
pit(true_values, predictions,  n_replicates = 5)
```

---

quantile_example_data *Quantile Example Data*

Description

A toy dataset for quantile forecasts of an outcome variable

Usage

```
quantile_example_data
```

Format

A data frame with 60 rows and 9 variables:

true_values true observed values

id unique identifier for true observed values

model name of the model that generated the forecasts

lower_90 prediction for the lower value of the 90% interval range (corresponding to the 5% quantile)

lower_50 prediction for the lower value of the 50% interval range (corresponding to the 25% quantile)

lower_0 prediction for the lower value of the 0% interval range (corresponding to the 50% quantile, i.e. the median. For computational reasons there need be a column with `lower_0` and `upper_0`)

upper_0 prediction for the upper value of the 0 (corresponding to the 50% quantile, i.e. the median)

upper_50 prediction for the upper value of the 50% interval range (corresponding to the 75% quantile)

upper_90 prediction for the lower value of the 90% interval range (corresponding to the 95% quantile)

scoringutils *scoringutils*

Description

This package is designed to help with assessing the quality of predictions. It provides a collection of proper scoring rules and metrics as well that can be accessed independently or collectively through a higher-level wrapper function.

Predicitions can be either probabilistic forecasts (generally predictive samples generated by Markov Chain Monte Carlo procedures), quantile forecasts or point forecasts. The true values can be either continuous, integer, or binary.

A collection of different metrics and scoring rules can be accessed through the function `eval_forecasts`. Given a data.frame of the correct form the function will automatically figure out the type of prediction and true values and return appropriate scoring metrics.

Alternatively, the following functions can be accessed directly:

- `brier_score`
- `pit`
- `bias`
- `sharpness`
- `crps`
- `logs`
- `dss`

sharpness *Determines sharpness of a probabilistic forecast*

Description

Determines sharpness of a probabilistic forecast

Usage

```
sharpness(predictions)
```

Arguments

`predictions`	nxN matrix of predictive samples, n (number of rows) being the number of data points and N (number of columns) the number of Monte Carlo samples

Details

Sharpness is the ability of the model to generate predictions within a narrow range. It is a data-independent measure, and is purely a feature of the forecasts themselves.

Shaprness of predictive samples corresponding to one single true value is measured as the normalised median of the absolute deviation from the median of the predictive samples. For details, see mad

Value

vector with sharpness values

References

Funk S, Camacho A, Kucharski AJ, Lowe R, Eggo RM, Edmunds WJ (2019) Assessing the performance of real-time epidemic forecasts: A case study of Ebola in the Western Area region of Sierra Leone, 2014-15. PLoS Comput Biol 15(2): e1006785. https://doi.org/10.1371/journal.pcbi.1006785

Examples

```
predictions <- replicate(200, rpois(n = 30, lambda = 1:30))
sharpness(predictions)
```

Index