Package ‘CalibratR’

August 27, 2018

Type  Package
Title  Mapping ML Scores to Calibrated Predictions
Version  0.1.1
Author  Johanna Schwarz
Maintainer  Dominik Heider <heiderd@mathematik.uni-marburg.de>
Description  Transforms your uncalibrated Machine Learning scores to well-calibrated prediction estimates that can be interpreted as probability estimates. The implemented BBQ (Bayes Binning in Quantiles) model is taken from Naeini (2015, ISBN:0-262-51129-0).
License  LGPL-3
Encoding  UTF-8
LazyData  true
Depends  R (>= 2.10.0)
Imports  ggplot2, pROC, reshape2, parallel, foreach, stats, fitdistrplus, doParallel
RoxygenNote  6.0.1
NeedsCompilation  no
Repository  CRAN
Date/Publication  2018-08-27 09:56:41 UTC

R topics documented:

BBQ.CV .................................................. 2
binom_for_histogram ........................................ 3
build_BBQ ................................................ 4
build_GUESS ............................................. 4
build_hist_binning ....................................... 5
calibrate ................................................. 5
calibrate_me ............................................... 7
calibrate_me.CV.errors ................................. 8
compare_models_visual ................................. 9
evaluate_discrimination ............................... 9
Description

trains and evaluates the BBQ calibration model using folds-Cross-Validation (CV). The predicted values are partitioned into n subsets. A BBQ model is constructed on (n-1) subsets; the remaining set is used for testing the model. All test set predictions are merged and used to compute error metrics for the model.

Usage

BBQ_CV(actual, predicted, method_for_prediction = 0, n_folds = 10, seed, input)
Arguments

actual  vector of observed class labels (0/1)
predicted  vector of uncalibrated predictions
method_for_prediction  0=selection, 1=averaging, Default: 0
n_folds  number of folds in the cross-validation, Default: 10
seed  random seed to alternate the split of data set partitions
input  specify if the input was scaled or transformed, scaled=1, transformed=2

Value

list object containing the following components:

error  list object that summarizes discrimination and calibration errors obtained during the CV
pred_idx  which BBQ prediction method was used during CV, 0=selection, 1=averaging
type  "BBQ"
probs_cv  vector of calibrated predictions that was used during the CV
actual_cv  respective vector of true values (0 or 1) that was used during the CV

Examples

```r
## Loading dataset in environment
data(example)
actual <- example$actual
predicted <- example$predicted
BBQ_model <- CalibratR::BBQ_CV(actual, predicted, method_for_prediction=0, n_folds=4, 123, 1)
```

---

### Description

P-values from stats::binom.test for each bin, if bin is empty, a p-value of 2 is returned

### Usage

```r
binom_for_histogram(n_x)
```

### Arguments

- `n_x`  numeric vector of two integers. The first one is the number of cases in the bin; the second the number of instances in the bin

### Value

P-value from stats::binom.test method
build_BBQ

Description
This method builds a BBQ calibration model using the trainings set provided.

Usage
build_BBQ(actual, predicted)

Arguments
- actual: vector of observed class labels (0/1)
- predicted: vector of uncalibrated predictions

Details
Based on the paper (and matlab code): "Obtaining Well Calibrated Probabilities Using Bayesian Binning" by Naeini, Cooper and Hauskrecht; https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4410090/

Value
returns the BBQ model which includes models for all evaluated binning schemes; the pruned model contains only a selection of BBQ models with the best Bayesian score

build_GUESS

Description
This method builds a GUESS calibration model using the trainings set provided.

Usage
build_GUESS(actual, predicted)

Arguments
- actual: vector of observed class labels (0/1)
- predicted: vector of uncalibrated predictions

Value
returns the trained GUESS model that can be used to calibrate a test set using the predict_GUESS method
**build_hist_binning**

**See Also**
- denscomp

---

**Description**

Calculate estimated probability per bin, input predicted and real score as numeric vector; builds a histogram binning model which can be used to calibrate uncalibrated predictions using the predict_histogramm_binning method.

**Usage**

`build_hist_binning(actual, predicted, bins = NULL)`

**Arguments**

- `actual` vector of observed class labels (0/1)
- `predicted` vector of uncalibrated predictions
- `bins` number of bins that should be used to build the binning model, Default: decide_on_break estimates optimal number of bins

**Details**

If trainings set is smaller than threshold (15 bins*5 elements=75), number of bins is decreased.

**Value**

Returns the trained histogram model that can be used to calibrate a test set using the `predict_hist_binning` method.

---

**calibrate**

**Description**

Builds selected calibration models on the supplied trainings values `actual` and `predicted` and returns them to the user. New test instances can be calibrated using the `predict_calibratr` function. Returns cross-validated calibration and discrimination error values for the models if `evaluate_CV_error` is set to TRUE. Repeated cross-Validation can be time-consuming.
Usage

```r
calibrate(actual, predicted, model_idx = c(1, 2, 3, 4, 5),
           evaluate_no.CV.error = TRUE, evaluate.CV.error = TRUE, folds = 10,
           n_seeds = 30, nCores = 4)
```

Arguments

- `actual`: vector of observed class labels (0/1)
- `predicted`: vector of uncalibrated predictions
- `model_idx`: which calibration models should be implemented, 1=hist_scaled, 2=hist_transformed, 3=BBQ_scaled, 4=BBQ_transformed, 5=GUESS, Default: c(1, 2, 3, 4, 5)
- `evaluate_no.CV.error`: computes internal errors for calibration models that were trained on all available `actual`/`predicted` tuples. Testing is performed with the same set. Be careful to interpret those error values, as they are not cross-validated. Default: TRUE
- `evaluate.CV.error`: computes cross-validation error. `folds` times cross validation is repeated `n_seeds` times with changing seeds. The trained models and their calibration and discrimination errors are returned. Evaluation of CV errors can take some time to compute, depending on the number of repetitions specified in `n_seeds`, Default: TRUE
- `folds`: number of folds in the cross-validation of the calibration model. If `folds` is set to 1, no CV is performed and `summary.CV` can be calculated. Default: 10
- `n_seeds`: `n_seeds` determines how often random data set partition is repeated with varying seed. If `folds` is 1, `n_seeds` should be set to 1, too. Default: 30
- `nCores`: how many cores should be used during parallelisation. Default: 4

Details

parallelised execution of random data set splits for the Cross-Validation procedure over `n_seeds`

Value

A list object with the following components:

- `calibration_models`: a list of all trained calibration models, which can be used in the `predict_calibratr` method.
- `summary.CV`: a list containing information on the CV errors of the implemented models
- `summary.no.CV`: a list containing information on the internal errors of the implemented models
- `predictions`: calibrated predictions for the original predicted values
- `n_seeds`: number of random data set partitions into training and test set for `folds`-times CV

Author(s)

Johanna Schwarz
**Examples**

```r
## Loading dataset in environment
data(example)
actual <- example$actual
predicted <- example$predicted

## Create calibration models
calibration_model <- calibrate(actual, predicted,
   model_idx = c(1,2),
   FALSE, FALSE, folds = 10, n_seeds = 1, nCores = 2)
```

---

**Description**

trains calibration models on the training set of predicted/actual value pairs. `model_idx` specifies which models should be trained.

**Usage**

`calibrate_me(actual, predicted, model_idx)`

**Arguments**

- `actual` vector of observed class labels (0/1)
- `predicted` vector of uncalibrated predictions
- `model_idx` a single number from 1 to 5, indicating which calibration model should be implemented, 1=hist_scaled, 2=hist_transformed, 3=BBQ_scaled, 4=BBQ_transformed, 5=GUESS

**Value**

depending on the value of `model_idx`, the respective calibration model is build on the input from `actual` and `predicted`
calibrate_me_CV_errors

Description

trains and evaluates calibration models using \( n \_seeds \)-times repeated \( n \_folds \)-Cross-Validation (CV). \( model \_idx \) specifies which models should be trained. Model training and evaluation is repeated \( n \_seeds \)-times with a different training/test set partition scheme for the CV each time.

Usage

\[
calibrate\_me\_CV\_errors(\text{actual}, \text{predicted}, \text{model\_idx}, \text{folds} = 10, n\_seeds, n\_cores)
\]

Arguments

- `actual`: vector of observed class labels (0/1)
- `predicted`: vector of uncalibrated predictions
- `model_idx`: which calibration models should be implemented, 1=hist_scaled, 2=hist_transformed, 3=BBQ_scaled, 4=BBQ_transformed, 5= GUESS
- `folds`: number of folds in the cross-validation, Default: 10
- `n_seeds`: \( n \_seeds \) determines how often random data set partition is repeated with varying seed
- `nCores`: \( n \_Cores \) how many cores should be used during parallelisation. Default: 4

Details

parallelised execution over \( n \_seeds \)

Value

returns all trained calibration models that were built during the \( n \_seeds \)-times repeated \( n \_folds \)-CV. Error values for each of the \( n \_seeds \) CV runs are given.
**compare_models_visual**

**Description**

FUNCTION_DESCRIPTION

**Usage**

```r
compare_models_visual(models, seq = NULL)
```

**Arguments**

- `models`: PARAM_DESCRIPTION
- `seq`: sequence for which the calibrated predictions should be plotted, Default: NULL

**Details**

DETAILS

**Value**

OUTPUT_DESCRIPTION

**See Also**

- `ggplot`, `geom_line`, `aes`, `ylim`, `theme`, `labs`, `scale_color_brewer`, `melt`

**evaluate_discrimination**

**Description**

computes various discrimination error values, namely: sensitivity, specificity, accuracy, positive predictive value (ppv), negative predictive value (npv) and AUC

**Usage**

```r
evaluate_discrimination(actual, predicted, cutoff = NULL)
```

**Arguments**

- `actual`: vector of observed class labels (0/1)
- `predicted`: vector of uncalibrated predictions
- `cutoff`: cut-off to be used for the computation of npv, ppv, sensitivity and specificity, Default: value that maximizes sensitivity and specificity (Youden-Index)
Value

list object with the following components:

sens        sensitivity
spec        specificity
acc         accuracy
ppv         positive predictive value
npv         negative predictive value
cutoff      cut-off that was used to compute the error values
auc         AUC value

See Also

roc

example

Description

list object containing 1) the simulated classifiers for two classes. Distributions are simulated from Gaussian distributions with Normal(mean=1.5, sd=0) for class 1 and Normal(mean=0, sd=0) for class 0 instances. Each class consists of 100 instances. and 2) A test set of 100 instances

Usage

data(example)

Format

predicted=vector of 200 simulated classifier values; actual=their respective true class labels (0/1)

format_values

Description

returns formatted input. If specified, the uncalibrated input is mapped to the [0;1] range using scaling (scale_me) or transforming (transform_me)

Usage

format_values(cases, control, input, min = NULL, max = NULL, mean = NULL)
getECE

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cases</td>
<td>instances from class 1</td>
</tr>
<tr>
<td>control</td>
<td>instances from class 0</td>
</tr>
<tr>
<td>input</td>
<td>single integer (0, 1 or 2). specify if the input should be formatted (=0), formatted and scaled (=1) or formatted and transformed (=2)</td>
</tr>
<tr>
<td>min</td>
<td>min value of the original data set, default=calculated on input</td>
</tr>
<tr>
<td>max</td>
<td>max value of the original data set, default=calculated on input</td>
</tr>
<tr>
<td>mean</td>
<td>mean value of the original data set, default=calculated on input</td>
</tr>
</tbody>
</table>

Value

list object with the following components:

- formatted_values
  - formatted input. If input is set to 1 (2), the input is additionally scaled (transformed) using the method `scale_me (transform_me)`
- min
  - minimum value among all instances
- max
  - maximum value among all instances
- mean
  - mean value among all instances

Description

Expected Calibration Error (ECE); the model is divided into 10 equal-width bins (default) and the mean of the observed (0/1) vs. mean of predicted is calculated per bin, weighted by empirical frequency of elements in bin i

Usage

```
getECE(actual, predicted, n_bins = 10)
```

Arguments

- actual
  - vector of observed class labels (0/1)
- predicted
  - vector of uncalibrated predictions
- n_bins
  - number of bins of the underlying equal-frequency histogram, Default: 10

Value

equal-frequency ECE value
getMCE

Description

Maximum Calibration Error (MCE), returns maximum calibration error for equal-frequency binning model

Usage

getMCE(actual, predicted, n_bins = 10)

Arguments

- actual: vector of observed class labels (0/1)
- predicted: vector of uncalibrated predictions
- n_bins: number of bins of the underlying equal-frequency histogram, Default: 10

Value

- equal-frequency MCE value

getRMSE

Description

calculates the root of mean square error (RMSE) in the test set of calibrated predictions

Usage

getRMSE(actual, predicted)

Arguments

- actual: vector of observed class labels (0/1)
- predicted: vector of uncalibrated predictions

Value

- RMSE value
**get_Brier_score**

Description

FUNCTION DESCRIPTION

Usage

get_Brier_score(actual, predicted)

Arguments

- actual: vector of observed class labels (0/1)
- predicted: vector of uncalibrated predictions

Details

DETAILS

Value

OUTPUT_DESCRIPTION

**get_CLE_class**

Description

calculates the class-specific classification error CLE in the test set. The method computes the deviation of the calibrated predictions of class 1 instances from their true value 1. For class 0 instances, get_CLE_class computes the deviation from 0. Class 1 CLE is 0 when all class 1 instances have a calibrated prediction of 1 regardless of potential miscalibration of class 0 instances. CLE calculation is helpful when miscalibration and classification is more cost-sensitive for one class than for the other.

Usage

get_CLE_class(actual, predicted, bins = 10)

Arguments

- actual: vector of observed class labels (0/1)
- predicted: vector of uncalibrated predictions
- bins: number of bins for the equal-width binning model, default=10
Value

object of class list containing the following components:

class_1  CLE of class 1 instances
class_0  CLE of class 0 instances

See Also

ggplot, geom_line, aes, position_dodge, labs, scale_colour_manual

Description

visualises how class 1 and class 0 classification error (CLE) differs in each trained calibration model. Comparing class-specific CLE helps to choose a calibration model for applications were classification error is cost-sensitive for one class. See get_CLE_class for details on the implementation.

Usage

get_CLE_comparison(list_models)

Arguments

list_models  list object that contains all error values for all trained calibration models. For the specific format, see the calling function visualize_calibratR.

Value

ggplot2

Description

Expected Calibration Error (ECE); the model is divided into 10 equal-width bins (default) and the mean of the observed (0/1) vs. mean of predicted is calculated per bin, weighted by empirical frequency of elements in bin i

Usage

get_ECE_equal_width(actual, predicted, bins = 10)
**get_MCE_equal_width**

**Arguments**

- actual: vector of observed class labels (0/1)
- predicted: vector of uncalibrated predictions
- bins: number of bins for the equal-width binning model

**Value**

- equal-width ECE value

**Description**

Maximum Calibration Error (MCE), returns maximum calibration error for equal-width binning model

**Usage**

```r
get_MCE_equal_width(actual, predicted, bins = 10)
```

**Arguments**

- actual: vector of observed class labels (0/1)
- predicted: vector of uncalibrated predictions
- bins: number of bins for the binning model

**Value**

- equal-width MCE value

---

**GUESS_CV**

**Description**

trains and evaluates the GUESS calibration model using folds-Cross-Validation (CV). The predicted values are partitioned into n subsets. A GUESS model is constructed on (n-1) subsets; the remaining set is used for testing the model. All test set predictions are merged and used to compute error metrics for the model.

**Usage**

```r
GUESS_CV(actual, predicted, n_folds = 10, method_of_prediction = 2, seed, input)
```
hist_binning_CV

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>actual</code></td>
<td>vector of observed class labels (0/1)</td>
</tr>
<tr>
<td><code>predicted</code></td>
<td>vector of uncalibrated predictions</td>
</tr>
<tr>
<td><code>n_folds</code></td>
<td>number of folds for the cross-validation, Default: 10</td>
</tr>
<tr>
<td><code>method_of_prediction</code></td>
<td>PARAM_DESCRIPTION, Default: 2</td>
</tr>
<tr>
<td><code>seed</code></td>
<td>random seed to alternate the split of data set partitions</td>
</tr>
<tr>
<td><code>input</code></td>
<td>specify if the input was scaled or transformed, scaled=1, transformed=2</td>
</tr>
</tbody>
</table>

Value

- `error` list object that summarizes discrimination and calibration errors obtained during the CV
- `type` "GUESS"
- `pred_idx` which prediction method was used during CV
- `probs_cv` vector of calibrated predictions that was used during the CV
- `actual_cv` respective vector of true values (0 or 1) that was used during the CV

Description

trains and evaluates the histogram binning calibration model repeated folds-Cross-Validation (CV). The predicted values are partitioned into n subsets. A histogram binning model is constructed on (n-1) subsets; the remaining set is used for testing the model. All test set predictions are merged and used to compute error metrics for the model.

Usage

hist_binning_CV(actual, predicted, n_bins = 15, n_folds = 10, seed, input)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>actual</code></td>
<td>vector of observed class labels (0/1)</td>
</tr>
<tr>
<td><code>predicted</code></td>
<td>vector of uncalibrated predictions</td>
</tr>
<tr>
<td><code>n_bins</code></td>
<td>number of bins used in the histogram binning scheme, Default: 15</td>
</tr>
<tr>
<td><code>n_folds</code></td>
<td>number of folds in the cross-validation, Default: 10</td>
</tr>
<tr>
<td><code>seed</code></td>
<td>random seed to alternate the split of data set partitions</td>
</tr>
<tr>
<td><code>input</code></td>
<td>specify if the input was scaled or transformed, scaled=1, transformed=2</td>
</tr>
</tbody>
</table>
Value

- list object containing the following components:
  - `error` list object that summarizes discrimination and calibration errors obtained during the CV
  - `type` "hist"
  - `probs.CV` vector of calibrated predictions that was used during the CV
  - `actual.CV` respective vector of true values (0 or 1) that was used during the CV

Description

plots the returned conditional class probabilities P(x|C) of GUESS_1 or GUESS_2 models. Which GUESS model is plotted can be specified in `pred_idx`.

Usage

`plot_class_distributions(build_guess_object, pred_idx)`

Arguments

- `build_guess_object` output from `build_GUESS()`
- `pred_idx` if `pred_idx=1` GUESS_1 is plotted; if `pred_idx=2` GUESS_2 is plotted

Value

ggplot object that visualizes the returned calibrated prediction estimates by GUESS_1 or GUESS_2

See Also

`melt ggplot geom_line aes.scale_colour_manual theme labs geom_vline geom_text`
**Description**

This method visualizes all implemented calibration models as a mapping function between original ML scores (x-axis) and calibrated predictions (y-axis).

**Usage**

```
plot_model(calibration_model, seq = NULL)
```

**Arguments**

- `calibration_model`: output from the `calibrate` method.
- `seq`: sequence of ML scores over which the mapping function should be evaluated. Default: 100 scores from the minimum to the maximum of the original ML scores.

**Value**

`ggplot` object

**See Also**

`melt`, `ggplot`, `geom_line`, `aes`, `ylim`, `scale_colour_manual`, `theme`, `labs`, `geom_text`, `geom_vline`

---

**Description**

FUNCTION_DESCRIPTION

**Usage**

```
predict_BBQ(bbq, new, option)
```

**Arguments**

- `bbq`: output from the `build_BBQ` method.
- `new`: vector of uncalibrated probabilities.
- `option`: either 1 or 0; averaging=1, selecting=0.
**predict_calibratR**

**Details**

Based on the paper (and matlab code): "Obtaining Well Calibrated Probabilities Using Bayesian Binning" by Naeini, Cooper and Hauskrecht; https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4410090/

**Value**

a list object containing the following components:

- **predictions**: contains a vector of calibrated predictions
- **pred_idx**: which option was used (averaging or selecting)
- **significance_test_set**: the percentage of new instances that was evaluated using significant prediction estimates
- **pred_per_bin**: number of instances new in each bin of the selected model

**Usage**

`predict_calibratR(calibration_models, new = NULL, nCores = 4)`

**Arguments**

- **calibration_models**: list of trained calibration models that were constructed using the `calibrate` method. The list components `calibration_models` from the `calibrate` output can be used directly.
- **new**: vector of new uncalibrated instances. Default: 100 scores from the minimum to the maximum of the original ML scores
- **nCores**: nCores how many cores should be used during parallelisation. Default: 4

**Details**

if no new value is given, the function will evaluate a sequence of numbers ranging from the minimum to the maximum of the original values in the training set
predict_GUESS

Value

list object with the following components:

- predictions: a list containing the calibrated predictions for each calibration model
- significance_test_set: a list containing the percentage of new instances for which prediction estimates are statistically significant
- pred_per_bin: a list containing the number of instances in each bin for the binning models

Author(s)

Johanna Schwarz

Examples

```r
## Loading dataset in environment
data(example)
test_set <- example$test_set
calibration_model <- example$calibration_model

## Predict for test set
predictions <- predict_calibratr(calibration_model$calibration_models, new=test_set, nCores = 2)
```

Description

returns calibrated predictions for the instances new using the trained GUESS calibration model build_guess_object. Two different evaluation methods are available. Method 1: returns the p-value for the score new under the distribution that is handed over in the build_guess_object Method 2: returns the probability density value for the score new under the distribution that is handed over in the build_guess_object

Usage

```r
predict_GUESS(build_guess_object, new, density_evaluation = 2, return_class_density = FALSE)
```

Arguments

- `build_guess_object`: output from the build_GUESS method
- `new`: vector of uncalibrated probabilities
- `density_evaluation`: which density evaluation method should be used to infer calculate probabilities, Default: 2
predict_hist_binning

return_class_density
    if set to TRUE, class densities p(x|class) are returned, Default: FALSE

Details
    dens_case and dens_control are only returned when return_class_density is set to TRUE

Value
    a list object containing the following components:
        predictions contains a vector of calibrated predictions
        pred_idx which density evaluation method was used
        significance_test_set the percentage of new instances that was evaluated using significant prediction estimates
        dens_case a vector containing the p(x|case) values
        dens_control a vector containing the p(x|control) values

Description
    predict for a new element using histogram binning

Usage
    predict_hist_binning(histogram, new)

Arguments
    histogram the output of build_hist_binning
    new vector of uncalibrated probabilities

Value
    a list object containing the following components
        predictions contains a vector of calibrated predictions
        significance_test_set the percentage of new instances that was evaluated using significant prediction estimates
        pred_per_bin a table containing the number of instances from new for each bin of the final binning scheme of histogram
**predict_model**

Description

Calibrates the uncalibrated predictions `new` using `calibration_model`.

Usage

`predict_model(new, calibration_model, min, max, mean, inputtype)`

Arguments

- `new` vector of uncalibrated predictions
- `calibration_model` calibration model to be used for the calibration. Can be the output of `build_BBQ`, `build_hist_binning` or `build_GUESS`.
- `min` minimum value of the original data set
- `max` maximum value of the original data set
- `mean` mean value of the original data set
- `inputtype` specify if the model was build on original (=0), scaled(=1) or transformed (=2) data

Value

vector of calibrated predictions

---

**rd_multiple_runs**

Description

This function plots all `n` reliability diagrams that were constructed during `n`-times repeated `m`-fold cross-validation (CV). During calibration model evaluation, CV is repeated `n` times, so that eventually `n` reliability diagrams are obtained.

Usage

`rd_multiple_runs(list_models)`

Arguments

- `list_models` list object that contains `n`-times the output from the `reliability_diagramm` method.
reliability_diagramm

Value

A list object that contains a reliability diagram that visualises all reliability diagrams that were constructed during n-times repeated m-fold cross-validation.

See Also

melt ggplot geom_line aes geom_abline ylab xlim ylim coord_fixed geom_text scale_color_discrete

Description

Reliability curves allow checking if the predicted probabilities of a

Usage

reliability_diagramm(actual, predicted, bins = 10, plot_rd = TRUE)

Arguments

actual vector of observed class labels (0/1)
predicted vector of uncalibrated predictions
bins number of bins in the reliability diagram, Default: 10
plot_rd should the reliability diagram be plotted, Default: TRUE

Value

A list object containing the following elements
calibration_error
discrimination_error
rd_breaks
histogram_plot
diagram_plot
mean_pred_per_bin
accuracy_per_bin
freq_per_bin
sign

See Also

ggplot stat_bin aes scale_fill_manual theme labs geom_point xlim ylim geom_abline geom_line geom_text geom_text geom_text geom_text
Description

maps all instances in x to the [0;1] range using the equation:
\[ y = \frac{x - \text{min}}{\text{max} - \text{min}} \]
If no values for \text{min} and \text{max} are given, they are calculated per default as \text{min} = \text{min}(x) and \text{max} = \text{max}(x)

Usage

\[ \text{scale}_\text{me}(x, \text{min} = \text{NULL}, \text{max} = \text{NULL}) \]

Arguments

- \text{x}  vector of predictions
- \text{min}  minimum of \text{x}, Default: \text{NULL}
- \text{max}  maximum of \text{x}, Default: \text{NULL}

Details

if \text{x} is greater (smaller) than \text{max} (\text{min}), its calibrated prediction is set to 1 (0) and warning is triggered.

Value

scaled values of \text{x}

Description

this method offers a variety of statistical evaluation methods for the output of the \text{calibrate} method. All returned error values represent mean error values over the n_seeds times repeated 10-fold CV.

Usage

\text{statistics_calibratR}(\text{calibrate_object, t.test_partitions = TRUE,}}
\text{significance_models = TRUE})
Arguments

calibrate_object

list that is returned from the calibrate function. The parameter n_seeds is available as a list component of the calibrate_object

t.test_partitions

Performs a paired two sided t.test over the error values (ECE, CLE1, CLE0, MCE, AUC, sensitivity and specificity) from the random partition splits comparing a possible significant difference in mean among the calibration models. All models and the original, scaled and transformed values are tested against each other. The p_value and the effect size of the t.test are returned to the user. Can only be performed, if the calibrate_object contains a summary_CV list object, else, an error is returned. Default: TRUE

significance_models

returns important characteristics of the implemented calibration models, Default: TRUE

Details

DETAILS

Value

An object of class list, with the following components:

mean_calibration

mean of calibration error values (ECE_equal_width, MCE_equal_width, ECE_equal_freq, MCE_equal_freq, RMSE, Class 1 CLE, Class 0 CLE, Brier Score, Class 1 Brier Score, Class 0 Brier Score) over n_seeds times repeated 10-fold CV. ECE and MCE are computed once using equal-width and once using equal-frequency binning for the construction of the underlying binning scheme. Only returned, if calibrate_object contains a summary_CV list object.

standard_deviation

standard deviation of calibration error values over n_seeds times repeated 10-fold CV. Only returned, if calibrate_object contains a summary_CV list object.

var_coeff_calibration

variation coefficient of calibration error values over n_seeds times repeated 10-fold CV. Only returned, if calibrate_object contains a summary_CV list object.

mean_discrimination

mean of discrimination error (sensitivity, specificity, AUC, positive predictive value, negative predictive value, accuracy) values over n_seeds times repeated 10-fold CV. The "cut-off" is the cut-off value that maximizes sensitivity and specificity. Only returned, if calibrate_object contains a summary_CV list object.

sd_discrimination

standard deviation of discrimination error values over n_seeds times repeated 10-fold CV. Only returned, if calibrate_object contains a summary_CV list object.
transform_me

var_coeff_discrimination
variation coefficient of discrimination error values over n_seeds times repeated
10-fold CV. Only returned, if calibrate_object contains a summary_CV list
object.

t.test_calibration
=list(p_value=t.test.calibration, effect_size=effect_size_calibration), only returned
if t.test=TRUE
t.test_discrimination
=list(p_value=t.test.discrimination, effect_size=effect_size_discrimination), only
returned if t.test=TRUE

significance_models
only returned if significance_models=TRUE

n_seeds
number of random data set partitions into training and test set for folds-times
CV

original_values
list object that consists of the actual and predicted values of the original
scores

Author(s)
Johanna Schwarz

See Also
t.test,friedman.test

Examples

## Loading dataset in environment
data(example)
calibration_model <- example$calibration_model

statistics <- statistics_calibratR(calibration_model)

transform_me

transform_me(x_unscaled, mean)
**Arguments**

- `x_unscaled`: vector of predictions
- `mean`: mean of `x`

**Details**

Values greater than `exp(700)/` or smaller than `exp(-700)` are returned as "Inf". To avoid NaN values, these "Inf." values are turned into min(y) or max(y).

**Value**

transformed values of `x_unscaled`

---

**Description**

Performs n_folds-CV but with only input-preprocessing the test set. No calibration model is trained and evaluated in this method. The predicted values are partitioned into n subsets. The training set is constructed on (n-1) subsets; the remaining set is used for testing. Since no calibration model is used in this method, the test set predictions are only input-preprocessed (either scaled or transformed, depending on input). All test set predictions are merged and used to compute error metrics for the input-preprocessing methods.

**Usage**

```
uncalibrated_cv(actual, predicted, n_folds = 10, seed, input)
```

**Arguments**

- `actual`: vector of observed class labels (0/1)
- `predicted`: vector of uncalibrated predictions
- `n_folds`: number of folds for the cross-validation, Default: 10
- `seed`: random seed to alternate the split of data set partitions
- `input`: specify if the input was scaled or transformed, scaled=1, transformed=2

**Value**

List object containing the following components:

- `error`: list object that summarizes discrimination and calibration errors obtained during the CV
- `type`: "uncalibrated"
- `probs_cv`: vector of input-preprocessed predictions that was used during the CV
- `actual_cv`: respective vector of true values (0 or 1) that was used during the CV
visualize_calibrated_test_set

Description
plots a panel for all calibrated predictions from the respective calibration model. Allows visual
collection of the models output and their optimal cut off

Usage
visualize_calibrated_test_set(actual, predicted_list, cutoffs)

Arguments
actual vector of observed class labels (0/1)
predicted_list predict_calibratR$predictions object (list of calibrated predictions from calibration models)
cutoffs vector of optimal cut-off thresholds for each calibration model

Value
ggplot2 element for visual comparison of the evaluated calibration models

See Also
ggplot, geom_point, scale_colour_manual, xlab, ylab, geom_hline, ylim
Arguments

- **calibrate_object**
  - the list component `calibration_models` from the `calibrate` method

- **visualize_models**
  - returns the list components `plot_calibration_models` and `plot_single_models`

- **plot_distributions**
  - returns a density distribution plot of the calibrated predictions after CV (External) or without CV (internal)

- **rd_partitions**
  - returns a reliability diagram for each model

- **training_set_calibrated**
  - returns a list of ggplots. Each plot represents the calibrated predictions by the respective calibration model of the training set. If the list object `predictions` in the `calibrate_object` is empty, `training_set_calibrated` is returned as NULL.

Value

An object of class list, with the following components:

- **histogram_distribution**
  - returns a histogram of the original ML score distribution

- **density_calibration_internal**
  - returns a list of density distribution plots for each calibration method, the original and the two input-preprocessing methods scaling and transforming. The plot visualises the density distribution of the calibrated predictions of the training set. In this case, training and test set values are identical, so be careful to evaluate the plots.

- **density_calibration_external**
  - returns a list of density distribution plots for each calibration method, the original and the two input-preprocessing methods scaling and transforming. The plot visualises the density distribution of the calibrated predictions, that were returned during Cross Validation. If more than one repetition of CV was performed, run number 1 is evaluated

- **plot_calibration_models**
  - maps the original ML scores to their calibrated prediction estimates for each model. This enables easy model comparison over the range of ML scores. See also `compare_models_visual`.

- **plot_single_models**
  - returns a list of ggplots for each calibration model, also mapping the original ML scores to their calibrated prediction. Significance values are indicated. See also `plot_model`

- **rd_plot**
  - returns a list of reliability diagrams for each of the implemented calibration models and the two input-preprocessing methods "scaled" and "transformed". The returned plot visualises the calibrated predictions that were returned for the test set during each of the n run of the n-times repeated CV. Each grey line represents one of the n runs. The blue line represents the median of all calibrated bin predictions. Insignificant bin estimates are indicated with "ns". If no CV
was performed during calibration model building using the `calibrate` method, `rd_plot` is returned as NULL.

`calibration_error`
returns a list of boxplots for the calibration error metrics ECE, MCE, CLE and RMSE. The \( n \) values for each model represent the obtained error values during the \( n \) times repeated CV. If no CV was performed during calibration model building using the `calibrate` method, `calibration_error` is returned as NULL.

`discrimination_error`
returns a list of boxplots for the discrimination error AUC, sensitivity and specificity. The \( n \) values for each model represent the obtained error values during the \( n \) times repeated CV. If no CV was performed during calibration model building using the `calibrate` method, `discrimination_error` is returned as NULL.

`cle_class_specific_error`
If no CV was performed during calibration model building using the `calibrate` method, `cle_class_specific_error` is returned as NULL.

`training_set_calibrated`
returns a list of ggplots. Each plot represents the calibrated predictions by the respective calibration model of the training set. If the list object predictions in the `calibrate_object` is empty, `training_set_calibrated` is returned as NULL.

`GUESS_1_final_model`
plots the the returned conditional probability \( p(x|\text{Class}) \) values of the GUESS_1 model.

`GUESS_2_final_model`
plots the the returned conditional probability \( p(x|\text{Class}) \) values of the GUESS_2 model.

**Author(s)**
Johanna Schwarz

**See Also**
`ggplot`, `geom_density`, `aes`, `scale_colour_manual`, `scale_fill_manual`, `labs`, `geom_point`, `geom_hline`, `theme`, `element_text`, `melt`

**Examples**
```r
## Loading dataset in environment
data(example)
calibration_model <- example$calibration_model

visualisation <- visualize_calibratR(calibration_model, plot_distributions=FALSE, rd_partitions=FALSE, training_set_calibrated=FALSE)
```
**visualize_distribution**

**Description**

FUNCTION_DESCRIPTION

**Usage**

visualize_distribution(actual, predicted)

**Arguments**

- **actual**: vector of observed class labels (0/1)
- **predicted**: vector of uncalibrated predictions

**Value**

- **plot_distribution**: ggplot histogram that visualizes the observed class distributions
- **parameter**: list object that summarizes all relevant parameters (mean, sd, number) of the observed class distributions

**See Also**

ggplot, geom_histogram, aes, scale_colour_manual, scale_fill_manual, labs

**visualize_error_boxplot**

**Description**

compares error values among different calibration models. A boxplots is created from the n error values that were obtained during the n-times repeated Cross-Validation procedure. Different error values are implemented and can be compared:

- discrimination error = sensitivity, specificity, accuracy, AUC (when discrimination=TRUE)
- calibration error = ece, mce, rmse, class 0 cle, class 1 cle (when discrimination=FALSE)

For the calculation of the errors, see the respective methods listed in the "see also" section

**Usage**

visualize_error_boxplot(list_models, discrimination = TRUE)
Arguments

- **list_models**: list object that contains all error values for all trained calibration models. For the specific format, see the calling function `visualize_calibratR`.
- **discrimination**: boolean (TRUE or FALSE). If TRUE, discrimination errors are compared between models; if FALSE calibration error is compared. Default: TRUE

Value

An object of class list, with the following components:

- if `discrimination=TRUE`
  - **sens**: ggplot2 boxplot that compares all evaluated calibration models with regard to sensitivity.
  - **spec**: ggplot2 boxplot that compares all evaluated calibration models with regard to specificity.
  - **acc**: ggplot2 boxplot that compares all evaluated calibration models with regard to accuracy.
  - **auc**: ggplot2 boxplot that compares all evaluated calibration models with regard to AUC.
  - **list_errors**: list object that contains all discrimination error values that were used to construct the boxplots.

- if `discrimination=FALSE`
  - **ece**: ggplot2 boxplot that compares all evaluated calibration models with regard to expected calibration error.
  - **mce**: ggplot2 boxplot that compares all evaluated calibration models with regard to maximum expected calibration error (MCE).
  - **rmse**: ggplot2 boxplot that compares all evaluated calibration models with regard to root mean square error (RMSE).
  - **cle_0**: ggplot2 boxplot that compares all evaluated calibration models with regard to class 0 classification error (CLE).
  - **cle_1**: ggplot2 boxplot that compares all evaluated calibration models with regard to class 1 classification error (CLE).
  - **list_errors**: list object that contains all calibration error values that were used to construct the boxplots.

See Also

- `ggplot`, `aes`, `ggtitle`, `scale_x_discrete`, `geom_boxplot`, `theme`, `element_text`, `melt`, `get_CLE_class`, `getECE`, `getMCE`, `getRMSE`, `evaluate_discrimination`
Index

*Topic datasets
  example, 10

aes, 9, 14, 17, 18, 23, 30–32

BBQ.CV, 2
binom_for_histogram, 3
build_BBQ, 4, 18, 22
build_GUESS, 4, 20, 22
build_hist_binning, 5, 21, 22
calibrate, 5, 18, 19, 24, 25, 29, 30
calibrate_me, 7
calibrate_me_CV_errors, 8
compare_models_visual, 9, 29
coord_fixed, 23
denscomp, 5
element_text, 30, 32
evaluate_discrimination, 9, 32
example, 10
format_values, 10
friedman.test, 26
geom_abline, 23
geom_boxplot, 32
geom_density, 30
geom_histogram, 31
geom_hline, 28, 30
geom_label, 23
geom_line, 9, 14, 17, 18, 23
geom_point, 23, 28, 30
geom_text, 17, 18, 23
geom_vline, 17, 18
get_Brier_score, 13
get_CLE_class, 13, 14, 32
get_CLE_comparison, 14
get_ECE_equal_width, 14
get_MCE_equal_width, 15
getECE, 11, 32
getMCE, 12, 32
getRMSE, 12, 32
ggplot, 9, 14, 17, 18, 23, 28, 30–32
ggtitle, 23, 32
GUESS.CV, 15
hist_binning.CV, 16
labs, 9, 14, 17, 18, 23, 30, 31
melt, 9, 14, 17, 18, 23, 30, 32
plot_class_distributions, 17
plot_model, 18, 29
position_dodge, 14
predict_BBQ, 18
predict_calibratR, 5, 6, 19
predict_GUESS, 4, 20
predict_hist_binning, 5, 21
predict_model, 22
rd_multiple_runs, 22
reliability_diagramm, 22, 23
roc, 10
scale_color_brewer, 9
scale_color_discrete, 23
scale_colour_manual, 14, 17, 18, 23, 30, 31
scale_fill_manual, 23, 30, 31
scale_me, 10, 11, 24
scale_x_discrete, 32
stat_bin, 23
statistics_calibratR, 24
t.test, 26
theme, 9, 17, 18, 23, 30, 32
transform_me, 10, 11, 26
uncalibrated.CV, 27
visualize_calibrated_test_set, 28
visualize_calibratR, 14, 28, 32
visualize_distribution, 31
visualize_error_boxplot, 31

xlab, 23, 28
xlim, 23

ylab, 23, 28
ylim, 9, 18, 23, 28