

# Package ‘CalibratR’

May 7, 2026

**Type** Package

**Title** Mapping ML Scores to Calibrated Predictions

**Version** 0.1.2

**Author** Johanna Schwarz, Dominik Heider

**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). Please cite this paper: Schwarz J and Heider D, Bioinformatics 2019, 35(14):2458-2465.

**License** LGPL-3

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 2.10.0)

**Imports** ggplot2, pROC, reshape2, parallel, foreach, stats,  
fitdistrplus, doParallel

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-08-19 13:00:05 UTC

## Contents

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

example . . . . .	10
format_values . . . . .	10
getECE . . . . .	11
getMCE . . . . .	12
getRMSE . . . . .	12
get_Brier_score . . . . .	13
get_CLE_class . . . . .	13
get_CLE_comparison . . . . .	14
get_ECE_equal_width . . . . .	14
get_MCE_equal_width . . . . .	15
GUESS_CV . . . . .	15
hist_binning_CV . . . . .	16
plot_class_distributions . . . . .	17
plot_model . . . . .	18
predict_BBQ . . . . .	18
predict_calibratR . . . . .	19
predict_GUESS . . . . .	20
predict_hist_binning . . . . .	21
predict_model . . . . .	22
rd_multiple_runs . . . . .	22
reliability_diagramm . . . . .	23
scale_me . . . . .	24
statistics_calibratR . . . . .	24
transform_me . . . . .	26
uncalibrated_CV . . . . .	27
visualize_calibrated_test_set . . . . .	28
visualize_calibratR . . . . .	28
visualize_distribution . . . . .	31
visualize_error_boxplot . . . . .	31

<b>Index</b>	<b>33</b>
--------------	-----------

---

BBQ\_CV

*BBQ\_CV*


---

## 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**

```
## 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)
```

---

binom\_for\_histogram    *binom\_for\_histogram*

---

**Description**

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

**Usage**

```
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`*build\_BBQ*

---

**Description**

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

**Usage**

```
build_BBQ(actual, predicted)
```

**Arguments**

<code>actual</code>	vector of observed class labels (0/1)
<code>predicted</code>	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 `prunedmodel` contains only a selection of BBQ models with the best Bayesian score

---

`build_GUESS`*build\_GUESS*

---

**Description**

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

**Usage**

```
build_GUESS(actual, predicted)
```

**Arguments**

<code>actual</code>	vector of observed class labels (0/1)
<code>predicted</code>	vector of uncalibrated predictions

**Value**

returns the trained GUESS model that can be used to calibrate a test set using the [predict\\_GUESS](#) method

**See Also**[denscomp](#)


---

build_hist_binning	<i>build_hist_binning</i>
--------------------	---------------------------

---

**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 then 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	<i>calibrate</i>
-----------	------------------

---

**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**

```
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 the 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	nCores how many cores should be used during parallelisation. Default: 4

**Details**

parallised 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 <a href="#">predict_calibratR</a> 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

```
## 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)
```

---

calibrate_me	<i>calibrate_me</i>
--------------	---------------------

---

## 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

*calibrate\_me\_CV\_errors*


---

### Description

trains and evaluates calibration models using `n_seeds`-times repeated 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(actual, predicted, model_idx, folds = 10, n_seeds,
                       nCores)
```

### Arguments

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

### Details

parallised execution over `n_seeds`

### Value

returns all trained calibration models that were built during the `n_seeds`-times repeated folds-CV. Error values for each of the `n_seeds` CV runs are given.

---

compare\_models\_visual *compare\_models\_visual*

---

**Description**

FUNCTION\_DESCRIPTION

**Usage**

```
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  
*evaluate\_discrimination*

---

**Description**

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

**Usage**

```
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	<i>example</i>
---------	----------------

---

**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	<i>format_values</i>
---------------	----------------------

---

**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)
```

**Arguments**

cases	instances from class 1
control	instances from class 0
input	single integer (0, 1 or 2). specify if the input should be formatted (=0), formatted and scaled (=1) or formatted and transformed (=2)
min	min value of the original data set, default=calculated on input
max	max value of the original data set, default=calculated on input
mean	mean value of the original data set, default=calculated on input

**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 <code>scale_me</code> ( <code>transform_me</code> )
min	minimum value among all instances
max	maximum value among all instances
mean	mean value among all instances

---

 getECE

*getECE*


---

**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`*getMCE*

---

**Description**

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

**Usage**

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

**Arguments**

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

**Value**

equal-frequency MCE value

---

`getRMSE`*getRMSE*

---

**Description**

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

**Usage**

```
getRMSE(actual, predicted)
```

**Arguments**

<code>actual</code>	vector of observed class labels (0/1)
<code>predicted</code>	vector of uncalibrated predictions

**Value**

RMSE value

---

get_Brier_score	<i>get_Brier_score</i>
-----------------	------------------------

---

**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	<i>get_CLE_class</i>
---------------	----------------------

---

**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:

<code>class_1</code>	CLE of class 1 instances
<code>class_0</code>	CLE of class 0 instances

**See Also**

[melt](#) [ggplot](#), [geom\\_line](#), [aes](#), [position\\_dodge](#), [labs](#), [scale\\_colour\\_manual](#)

`get_CLE_comparison`     *get\_CLE\_comparison*

**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 where classification error is cost-sensitive for one class. See [get\\_CLE\\_class](#) for details on the implementation.

**Usage**

```
get_CLE_comparison(list_models)
```

**Arguments**

<code>list_models</code>	list object that contains all error values for all trained calibration models. For the specific format, see the calling function <a href="#">visualize_calibratR</a> .
--------------------------	--

**Value**

ggplot2

`get_ECE_equal_width`     *get\_ECE\_equal\_width*

**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)
```

**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

---

<code>get_MCE_equal_width</code>	<i>get_MCE_equal_width</i>
----------------------------------	----------------------------

---

**Description**

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

**Usage**

```
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

---

<code>GUESS_CV</code>	<i>GUESS_CV</i>
-----------------------	-----------------

---

**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**

```
GUESS_CV(actual, predicted, n_folds = 10, method_of_prediction = 2, 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
method_of_prediction	PARAM_DESCRIPTION, Default: 2
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	"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

---

hist_binning_CV	<i>hist_binning_CV</i>
-----------------	------------------------

---

**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**

actual	vector of observed class labels (0/1)
predicted	vector of uncalibrated predictions
n_bins	number of bins used in the histogram binning scheme, Default: 15
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
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

---

plot\_class\_distributions  
*plot\_class\_distributions*

---

**Description**

plots the 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](#)

---

plot_model	<i>plot_model</i>
------------	-------------------

---

### Description

this methods 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 <a href="#">calibrate</a> 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](#)

---

predict_BBQ	<i>predict_BBQ</i>
-------------	--------------------

---

### Description

FUNCTION\_DESCRIPTION

### Usage

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

### Arguments

bbq	output from the <a href="#">build_BBQ</a> method
new	vector of uncalibrated probabilities
option	either 1 or 0; averaging=1, selecting=0

**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

---

predict\_calibratR      *predict\_calibratR*

---

**Description**

maps the uncalibrated predictions new into calibrated predictions using the passed over calibration models

**Usage**

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

**Arguments**

calibration_models	list of trained calibration models that were constructed using the <code>calibrate</code> method. The list components <code>calibration_models</code> from the <code>calibrate</code> 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

**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**

```
## 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)
```

---

`predict_GUESS`                      *predict\_GUESS*

---

**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**

```
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

return\_class\_density  
 if set to TRUE, class densities p(xlclass) 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(xlcase) values
- dens\_control a vector containing the p(xlcontrol) values

---

predict\_hist\_binning *predict\_hist\_binning*

---

**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	<i>predict_model</i>
---------------	----------------------

---

**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 <a href="#">build_BBQ</a> , <a href="#">build_hist_binning</a> or <a href="#">build_GUESS</a> .
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	<i>rd_multiple_runs</i>
------------------	-------------------------

---

**Description**

This functions 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 <a href="#">reliability_diagramm</a> . method.
-------------	--

**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](#), [xlab](#), [xlim](#), [ylim](#), [coord\\_fixed](#), [geom\\_text](#), [scale\\_color\\_discrete](#), [ggti](#)

---

reliability\_diagramm    *reliability\_diagramm*

---

**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](#)

---

scale_me	<i>scale_me</i>
----------	-----------------

---

**Description**

maps all instances in  $x$  to the  $[0;1]$  range using the equation:

$$y = (x - \min) / (\max - \min)$$

If no values for  $\min$  and  $\max$  are given, they are calculated per default as  $\min = \min(x)$  and  $\max = \max(x)$

**Usage**

```
scale_me(x, min = NULL, max = NULL)
```

**Arguments**

$x$	vector of predictions
$\min$	minimum of $x$ , Default: NULL
$\max$	maximum of $x$ , Default: NULL

**Details**

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

**Value**

scaled values of  $x$

---

statistics_calibratR	<i>statistics_calibratR</i>
----------------------	-----------------------------

---

**Description**

this method offers a variety of statistical evaluation methods for the output of the [calibrate](#) method. All returned error values represent mean error values over the  $n\_seeds$  times repeated 10-fold CV.

**Usage**

```
statistics_calibratR(calibrate_object, t.test_partitions = TRUE,
  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.

`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*

---

**Description**

maps all instances in `x_unscaled` to the [0;1] range using the equation:  
 $y = \exp(x) / (1 + \exp(x))$

**Usage**

```
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

---

uncalibrated_CV	<i>uncalibrated_CV</i>
-----------------	------------------------

---

**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
      visualize_calibrated_test_set
```

---

**Description**

plots a panel for all calibrated predictions from the respective calibration model. Allows visual comparison 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](#)

---

```
visualize_calibratR  visualize_calibratR
```

---

**Description**

this method offers a variety of visualisations to compare implemented calibration models

**Usage**

```
visualize_calibratR(calibrate_object, visualize_models = FALSE,
  plot_distributions = FALSE, rd_partitions = FALSE,
  training_set_calibrated = FALSE)
```

**Arguments**

calibrate_object	the list component calibration_models from the <a href="#">calibrate</a> 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 <a href="#">compare_models_visual</a> .
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 <a href="#">plot_model</a>
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  $p(x|Class)$  values of the GUESS\_1 model

`GUESS_2_final_model`

plots the the returned conditional probability  $p(x|Class)$  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

```
## 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
      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**

list object containing the following components:

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
      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

- \* **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, 28, 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](#)