

# Package ‘MetricsWeighted’

November 8, 2019

**Type** Package

**Title** Weighted Metrics, Scoring Functions and Performance Measures for Machine Learning

**Version** 0.3.0

**Date** 2019-11-08

**Maintainer** Michael Mayer <mayermichael79@gmail.com>

**Description** Provides weighted versions of several metrics, scoring functions and performance measures used in machine learning, including average unit deviances of the Bernoulli, Tweedie, Poisson, and Gamma distributions, see Jorgensen B. (1997, ISBN: 978-0412997112). The package also contains a weighted version of generalized R-squared, see e.g. Cohen, J. et al. (2002, ISBN: 978-0805822236). Furthermore, 'dplyr' chains are supported.

**License** GPL (>= 2)

**Depends** R (>= 3.5.0)

**VignetteBuilder** knitr

**Encoding** UTF-8

**LazyData** true

**Imports** stats

**Suggests** dplyr, knitr

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Author** Michael Mayer [aut, cre, cph],  
Christian Lorentzen [rev]

**Repository** CRAN

**Date/Publication** 2019-11-08 18:10:02 UTC

**R topics documented:**

accuracy	2
AUC	3
classification_error	4
deviance_bernoulli	5
deviance_gamma	5
deviance_normal	6
deviance_poisson	7
deviance_tweedie	8
f1_score	9
gini_coefficient	10
logLoss	10
mae	11
mape	12
medae	12
mse	13
performance	14
precision	15
recall	16
rmse	17
r_squared	18
weighted_mean	19
weighted_median	20
weighted_quantile	20
weighted_var	21

<b>Index</b>	<b>23</b>
--------------	-----------

---

accuracy	<i>Accuracy</i>
----------	-----------------

---

**Description**

Calculates weighted accuracy, i.e. the weighted proportion of elements in predicted that are equal to those in actual. The higher, the better.

**Usage**

```
accuracy(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
...	Further arguments passed to weighted_mean.

**Value**

A numeric vector of length one.

**See Also**

[classification\\_error](#).

**Examples**

```
accuracy(c(0, 0, 1, 1), c(0, 0, 1, 1))
accuracy(c(1, 0, 0, 1), c(0, 0, 1, 1))
accuracy(c(1, 0, 0, 1), c(0, 0, 1, 1), w = 1:4)
```

---

AUC

*Area under the ROC*

---

**Description**

Function copied from glmnet package (modified to ensure deterministic results). Calculates weighted AUC, i.e. the area under the receiver operating curve. The larger, the better.

**Usage**

```
AUC(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values (0 or 1).
predicted	Predicted values of any value (not necessarily between 0 and 1).
w	Optional case weights.
...	Further arguments passed by other methods.

**Details**

The unweighted version can be different from the weighted one with unit weights due to ties in predicted.

**Value**

A numeric vector of length one.

**See Also**

[gini\\_coefficient](#).

**Examples**

```

AUC(c(0, 0, 1, 1), c(0.1, 0.1, 0.9, 0.8))
AUC(c(1, 0, 0, 1), c(0.1, 0.1, 0.9, 0.8))
AUC(c(1, 0, 0, 1), 2 * c(0.1, 0.1, 0.9, 0.8))
AUC(c(1, 0, 0, 1), c(0.1, 0.1, 0.9, 0.8), w = rep(1, 4)) # different from last due to ties
AUC(c(1, 0, 0, 1), c(0.1, 0.2, 0.9, 0.8))
AUC(c(1, 0, 0, 1), c(0.1, 0.2, 0.9, 0.8), w = rep(1, 4)) # same as last (no ties)
AUC(c(0, 0, 1, 1), c(0.1, 0.1, 0.9, 0.8), w = 1:4)

```

---

classification\_error *Classification Error*

---

**Description**

Calculates weighted classification error, i.e. the weighted proportion of elements in predicted that are unequal to those in observed. Equals 1 - accuracy, thus lower values are better.

**Usage**

```
classification_error(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
...	Further arguments passed to accuracy.

**Value**

A numeric vector of length one.

**See Also**

[accuracy](#).

**Examples**

```

classification_error(c(0, 0, 1, 1), c(0, 0, 1, 1))
classification_error(c(1, 0, 0, 1), c(0, 0, 1, 1))
classification_error(c(1, 0, 0, 1), c(0, 0, 1, 1), w = 1:4)

```

---

deviance_bernoulli	<i>Bernoulli Deviance</i>
--------------------	---------------------------

---

**Description**

Calculates weighted average of unit Bernoulli deviance. Defined as twice logLoss. The smaller the deviance, the better.

**Usage**

```
deviance_bernoulli(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values (0 or 1).
predicted	Predicted values strictly between 0 and 1.
w	Optional case weights.
...	Further arguments passed to logLoss.

**Value**

A numeric vector of length one.

**See Also**

[logLoss](#).

**Examples**

```
deviance_bernoulli(c(0, 0, 1, 1), c(0.1, 0.1, 0.9, 0.8))
deviance_bernoulli(c(1, 0, 0, 1), c(0.1, 0.1, 0.9, 0.8))
deviance_bernoulli(c(0, 0, 1, 1), c(0.1, 0.1, 0.9, 0.8), w = 1:4)
```

---

deviance_gamma	<i>Gamma Deviance</i>
----------------	-----------------------

---

**Description**

Weighted average of (unscaled) unit Gamma deviance, see e.g. [1]. Special case of Tweedie deviance with Tweedie parameter 2. The smaller the deviance, the better.

**Usage**

```
deviance_gamma(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Strictly positive observed values.
predicted	Strictly positive predicted values.
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**References**

[1] Jorgensen, B. (1997). The Theory of Dispersion Models. Chapman & Hall/CRC. ISBN 978-0412997112.

**See Also**

[deviance\\_tweedie](#).

**Examples**

```
deviance_gamma(1:10, c(1:9, 12))
deviance_gamma(1:10, c(1:9, 12), w = rep(1, 10))
deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 2)
deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 1.99)
deviance_gamma(1:10, c(1:9, 12), w = 1:10)
```

---

deviance_normal	<i>Normal Deviance</i>
-----------------	------------------------

---

**Description**

Weighted average of (unscaled) unit normal deviance. This equals the weighted mean-squared error, see e.g. [1]. The smaller the deviance, the better.

**Usage**

```
deviance_normal(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
...	Further arguments passed to <code>mse</code> .

**Value**

A numeric vector of length one.

**References**

[1] Jorgensen, B. (1997). The Theory of Dispersion Models. Chapman & Hall/CRC. ISBN 978-0412997112.

**See Also**

[deviance\\_tweedie](#), [mse](#).

**Examples**

```
deviance_normal(1:10, c(1:9, 12))
deviance_normal(1:10, c(1:9, 12), w = rep(1, 10))
deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 0)
deviance_normal(1:10, c(1:9, 12), w = 1:10)
```

---

deviance_poisson	<i>Poisson Deviance</i>
------------------	-------------------------

---

**Description**

Weighted average of unit Poisson deviance, see [1]. Special case of Tweedie deviance with Tweedie parameter 1.

**Usage**

```
deviance_poisson(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed non-negative values.
predicted	Strictly positive predicted values.
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**References**

[1] Jorgensen, B. (1997). The Theory of Dispersion Models. Chapman & Hall/CRC. ISBN 978-0412997112.

**See Also**

[deviance\\_tweedie](#).

**Examples**

```
deviance_poisson(0:2, c(0.1, 1, 3))
deviance_poisson(0:2, c(0.1, 1, 3), w = c(1, 1, 1))
deviance_tweedie(0:2, c(0.1, 1, 3), tweedie_p = 1)
deviance_tweedie(0:2, c(0.1, 1, 3), tweedie_p = 1.01)
deviance_poisson(0:2, c(0.1, 1, 3), w = 1:3)
```

---

deviance_tweedie	<i>Tweedie Deviance</i>
------------------	-------------------------

---

**Description**

Weighted average of (unscaled) unit Tweedie deviance with parameter  $p$ . This includes the normal deviance ( $p = 0$ ), the Poisson deviance ( $p = 1$ ), as well as the Gamma deviance ( $p = 2$ ), see [1] for a reference and [https://en.wikipedia.org/wiki/Tweedie\\_distribution](https://en.wikipedia.org/wiki/Tweedie_distribution) for the specific deviance formula. For  $0 < p < 1$ , the distribution is not defined. The smaller the deviance, the better.

**Usage**

```
deviance_tweedie(actual, predicted, w = NULL, tweedie_p = 0, ...)
```

**Arguments**

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
tweedie_p	Tweedie power.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**References**

[1] Jorgensen, B. (1997). The Theory of Dispersion Models. Chapman & Hall/CRC. ISBN 978-0412997112.

**See Also**

[deviance\\_normal](#), [deviance\\_poisson](#), [deviance\\_gamma](#).



**Examples**

```

deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 0)
deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 1)
deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 2)
deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 1.5)
deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 1.5, w = rep(1, 10))
deviance_tweedie(1:10, c(1:9, 12), tweedie_p = 1.5, w = 1:10)

```

---

f1\_score

*F1 Score*


---

**Description**

Calculates weighted F1 score or F measure defined as the harmonic mean of precision and recall, see [https://en.wikipedia.org/wiki/Precision\\_and\\_recall](https://en.wikipedia.org/wiki/Precision_and_recall) for some background. The higher, the better.

**Usage**

```
f1_score(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values (0 or 1).
predicted	Predicted values (0 or 1).
w	Optional case weights.
...	Further arguments passed to precision and recall.

**Value**

A numeric vector of length one.

**See Also**

[precision](#), [recall](#).

**Examples**

```

f1_score(c(0, 0, 1, 1), c(0, 0, 1, 1))
f1_score(c(1, 0, 0, 1), c(0, 0, 1, 1))
f1_score(c(1, 0, 0, 1), c(0, 0, 1, 1), w = 1:4)

```

---

gini_coefficient	<i>Gini Coefficient</i>
------------------	-------------------------

---

**Description**

Calculates weighted Gini coefficient, obtained as  $2 * \text{AUC} - 1$ . Up to ties in predicted equivalent to Somer's D. The larger the Gini coefficient, the better.

**Usage**

```
gini_coefficient(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values (0 or 1).
predicted	Predicted values of any value (not necessarily between 0 and 1).
w	Optional case weights.
...	Further arguments passed to AUC.

**Value**

A numeric vector of length one.

**See Also**

[AUC](#).

**Examples**

```
gini_coefficient(c(0, 0, 1, 1), 2 * c(0.1, 0.1, 0.9, 0.8))
gini_coefficient(c(0, 0, 1, 1), c(0.1, 0.6, 0.9, 0.5))
gini_coefficient(c(0, 0, 1, 1), c(0.1, 0.6, 0.9, 0.5), w = 1:4)
```

---

logLoss	<i>Log Loss/Binary Cross Entropy</i>
---------	--------------------------------------

---

**Description**

Calculates weighted logloss resp. cross entropy. Equals half of the unit Bernoulli deviance. The smaller, the better.

**Usage**

```
logLoss(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values (0 or 1).
predicted	Predicted values strictly larger than 0 and smaller than 1.
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**See Also**

[deviance\\_bernoulli](#).

**Examples**

```
logLoss(c(0, 0, 1, 1), c(0.1, 0.1, 0.9, 0.8))
logLoss(c(1, 0, 0, 1), c(0.1, 0.1, 0.9, 0.8))
logLoss(c(0, 0, 1, 1), c(0.1, 0.1, 0.9, 0.8), w = 1:4)
```

---

mae

---

*Mean Absolute Error*


---

**Description**

Calculates weighted mean absolute error of predicted values. The smaller the value, the better.

**Usage**

```
mae(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**Examples**

```
mae(1:10, c(1:9, 12))
mae(1:10, c(1:9, 12), w = rep(1, 10))
mae(1:10, c(1:9, 12), w = 1:10)
```

---

mape	<i>Mean Absolute Percentage Error</i>
------	---------------------------------------

---

**Description**

Calculates weighted mean absolute percentage error of predicted values. The smaller, the better.

**Usage**

```
mape(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Strictly positive observed values.
predicted	Predicted values.
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**Examples**

```
mape(1:10, c(1:9, 12))
mape(1:10, c(1:9, 12), w = rep(1, 10))
mape(1:10, c(1:9, 12), w = 1:10)
```

---

medae	<i>Median Absolute Error</i>
-------	------------------------------

---

**Description**

Calculates weighted median absolute error of predicted values. The smaller the value, the better.

**Usage**

```
medae(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**Examples**

```
medae(1:10, c(2:10, 100))
medae(1:10, c(2:10, 100), w = rep(1, 10))
medae(1:10, c(2:10, 100), w = 1:10)
```

---

mse

*Mean-Squared Error*

---

**Description**

Calculates weighted mean-squared error of prediction. Equals mean unit normal deviance. The smaller, the better.

**Usage**

```
mse(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**See Also**

[rmse](#), [deviance\\_normal](#).

**Examples**

```
mse(1:10, c(1:9, 12))
mse(1:10, c(1:9, 12), w = rep(1, 10))
mse(1:10, c(1:9, 12), w = 1:10)
```

---

performance	<i>Performance</i>
-------------	--------------------

---

### Description

Applies one or more metrics to a `data.frame` containing columns with actual and predicted values as well as an optional column with case weights. The results are returned as a `data.frame` and can be used in a `dplyr` chain.

### Usage

```
performance(data, actual, predicted, w = NULL, metrics = rmse,
            key = "metric", value = "value", ...)
```

### Arguments

<code>data</code>	A <code>data.frame</code> containing actual, predicted and possibly <code>w</code> .
<code>actual</code>	The column name in <code>data</code> referring to actual values.
<code>predicted</code>	The column name in <code>data</code> referring to predicted values.
<code>w</code>	The optional column name in <code>data</code> referring to case weights.
<code>metrics</code>	Either a function or a named list of functions. Each function represents a metric and has four arguments: observed, predicted, case weights and <code>...</code> . If not a named list but a single function, the name of the function is guessed by <code>deparse(substitute(...))</code> , which would not provide the actual name of the function if called within <code>lapply</code> etc. In such cases, you can pass a named list with one element, e.g. <code>list(rmse = rmse)</code> .
<code>key</code>	Name of the resulting column containing the name of the metric. Defaults to "metric".
<code>value</code>	Name of the resulting column with the value of the metric. Defaults to "value".
<code>...</code>	Further arguments passed to the metric functions, e.g. if the metric is "r_squared", you could pass the relevant deviance function as additional argument (see examples).

### Value

Data frame with one row per metric and two columns: `key` and `value`.

### Examples

```
ir <- iris
fit_num <- lm(Sepal.Length ~ ., data = ir)
ir$fitted <- fit_num$fitted
performance(ir, "Sepal.Length", "fitted")
performance(ir, "Sepal.Length", "fitted", metrics = r_squared)
performance(ir, "Sepal.Length", "fitted", metrics = c(`R-squared` = r_squared, rmse = rmse))
performance(ir, "Sepal.Length", "fitted", metrics = r_squared,
```

```

      deviance_function = deviance_gamma)
performance(ir, "Sepal.Length", "fitted", metrics = r_squared,
  deviance_function = deviance_tweedie)
performance(ir, "Sepal.Length", "fitted", metrics = r_squared,
  deviance_function = deviance_tweedie, tweedie_p = 2)
performance(ir, "Sepal.Length", "fitted", metrics = r_squared,
  deviance_function = deviance_tweedie, tweedie_p = 0)
## Not run:
library(dplyr)

iris %>%
  mutate(pred = predict(fit_num, data = .)) %>%
  performance("Sepal.Length", "pred")

# Same
iris %>%
  mutate(pred = predict(fit_num, data = .)) %>%
  performance("Sepal.Length", "pred", metrics = rmse)

# Grouped by Species
iris %>%
  mutate(pred = predict(fit_num, data = .)) %>%
  group_by(Species) %>%
  do(performance(., "Sepal.Length", "pred"))

# Multiple measures
iris %>%
  mutate(pred = predict(fit_num, data = .)) %>%
  performance("Sepal.Length", "pred",
    metrics = list(rmse = rmse, mae = mae, `R-squared` = r_squared))

# Grouped by Species
iris %>%
  mutate(pred = predict(fit_num, data = .)) %>%
  group_by(Species) %>%
  do(performance(., "Sepal.Length", "pred",
    metrics = list(rmse = rmse, mae = mae, `R-squared` = r_squared)))

## End(Not run)

```

---

```
precision
```

```
Precision
```

---

## Description

Calculates weighted precision, see [https://en.wikipedia.org/wiki/Precision\\_and\\_recall](https://en.wikipedia.org/wiki/Precision_and_recall) for the (unweighted) version. The higher, the better.

## Usage

```
precision(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values (0 or 1).
predicted	Predicted values (0 or 1).
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**See Also**

[recall](#), [f1\\_score](#).

**Examples**

```
precision(c(0, 0, 1, 1), c(0, 0, 1, 1))
precision(c(1, 0, 0, 1), c(0, 0, 1, 1))
precision(c(1, 0, 0, 1), c(0, 0, 1, 1), w = 1:4)
```

---

recall

*Recall*

---

**Description**

Calculates weighted recall, see [https://en.wikipedia.org/wiki/Precision\\_and\\_recall](https://en.wikipedia.org/wiki/Precision_and_recall) for the (unweighted) definition. The higher, the better.

**Usage**

```
recall(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values (0 or 1).
predicted	Predicted values (0 or 1).
w	Optional case weights.
...	Further arguments passed to <code>weighted_mean</code> .

**Value**

A numeric vector of length one.

**See Also**

[precision](#), [f1\\_score](#).



**Examples**

```
recall(c(0, 0, 1, 1), c(0, 0, 1, 1))
recall(c(1, 0, 0, 1), c(0, 0, 1, 1))
recall(c(1, 0, 0, 1), c(0, 0, 1, 1), w = 1:4)
```

---

rmse	<i>Root-Mean-Squared Error</i>
------	--------------------------------

---

**Description**

Weighted root-mean-squared error of predicted values. Equals the square root of mean-squared error. Smaller values are better.

**Usage**

```
rmse(actual, predicted, w = NULL, ...)
```

**Arguments**

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
...	Further arguments passed to mse.

**Value**

A numeric vector of length one.

**See Also**

[mse](#).

**Examples**

```
rmse(1:10, c(1:9, 12))
rmse(1:10, c(1:9, 12), w = rep(1, 10))
rmse(1:10, c(1:9, 12), w = 1:10)
```

---

r_squared	<i>Pseudo R-Squared</i>
-----------	-------------------------

---

### Description

Returns (weighted) proportion of deviance explained, see e.g. [1]. For the mean-squared error as deviance, this equals the usual (weighted) R-squared. The higher, the better.

### Usage

```
r_squared(actual, predicted, w = NULL, deviance_function = mse, ...)
```

### Arguments

actual	Observed values.
predicted	Predicted values.
w	Optional case weights.
deviance_function	A positive (deviance) function taking four arguments: "actual", "predicted", "w" and "...".
...	Further arguments passed to weighted_mean and deviance_function.

### Value

A numeric vector of length one.

### References

[1] Cohen, Jacob. et al. (2002). Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences (3rd ed.). Routledge. ISBN 978-0805822236.

### See Also

[deviance\\_normal](#), [mse](#).

### Examples

```
r_squared(1:10, c(1, 1:9))
r_squared(1:10, c(1, 1:9), w = rep(1, 10))
r_squared(1:10, c(1, 1:9), w = 1:10)
r_squared(1:10, c(1, 1:9), deviance_function = deviance_normal)
r_squared(0:2, c(0.1, 1, 2), deviance_function = deviance_poisson)
r_squared(0:2, c(0.1, 1, 2), w = rep(1, 3), deviance_function = deviance_poisson)
r_squared(0:2, c(0.1, 1, 2), deviance_function = deviance_tweedie, tweedie_p = 1)
r_squared(0:2, c(0.1, 1, 2), w = rep(1, 3),
  deviance_function = deviance_tweedie, tweedie_p = 1)
```

```
# respect to own deviance formula
myTweedie <- function(actual, predicted, w = NULL, ...) {
  deviance_tweedie(actual, predicted, w, tweedie_p = 1.5, ...)
}
r_squared(1:10, c(1, 1:9), deviance_function = myTweedie)
```

---

weighted_mean	<i>Weighted Mean</i>
---------------	----------------------

---

### Description

Returns weighted mean of a numeric vector. In contrast to `stats::weighted.mean`, `w` does not need to be specified.

### Usage

```
weighted_mean(x, w = NULL, ...)
```

### Arguments

<code>x</code>	Numeric vector.
<code>w</code>	Optional non-negative, non-missing case weights.
<code>...</code>	Further arguments passed to <code>mean</code> or <code>weighted.mean</code> .

### Value

A length-one numeric vector.

### See Also

[weighted\\_quantile](#).

### Examples

```
weighted_mean(1:10)
weighted_mean(1:10, w = NULL)
weighted_mean(1:10, w = 1:10)
```

---

weighted_median	<i>Weighted Median</i>
-----------------	------------------------

---

**Description**

Calculates weighted median. For odd sample sizes consistent with unweighted quantiles.

**Usage**

```
weighted_median(x, w = NULL, ...)
```

**Arguments**

x	Numeric vector.
w	Optional non-negative case weights.
...	Further arguments passed to weighted_quantile.

**See Also**

[weighted\\_quantile](#).

**Examples**

```
n <- 21
x <- seq_len(n)
quantile(x, probs = 0.5)
weighted_median(x, w = rep(1, n))
weighted_median(x, w = x)
quantile(rep(x, x), probs = 0.5)
```

---

weighted_quantile	<i>Weighted Quantiles</i>
-------------------	---------------------------

---

**Description**

Calculates weighted quantiles based on the generalized inverse of the weighted ECDF. If no weights are passed, uses stats::quantile.

**Usage**

```
weighted_quantile(x, w = NULL, probs = seq(0, 1, 0.25), na.rm = TRUE,
  names = TRUE, ...)
```

**Arguments**

x	Numeric vector.
w	Optional non-negative case weights.
probs	Vector of probabilities.
na.rm	Ignore missing data?
names	Return names?
...	Further arguments passed to <code>stats::quantile</code> in the unweighted case. Not used in the weighted case.

**See Also**

[weighted\\_median](#).

**Examples**

```
n <- 10
x <- seq_len(n)
quantile(x)
weighted_quantile(x)
weighted_quantile(x, w = rep(1, n))
quantile(x, type = 1)
weighted_quantile(x, w = x) # same as Hmisc::wtd.quantile
weighted_quantile(x, w = x, names = FALSE)
weighted_quantile(x, w = x, probs = 0.5, names = FALSE)

# Example with integer weights
x <- c(1, 1:11, 11, 11)
w <- seq_along(x)
weighted_quantile(x, w)
quantile(rep(x, w)) # same
```

---

weighted\_var

*Weighted Variance*

---

**Description**

Calculates weighted variance, see `stats::cov.wt` or [https://en.wikipedia.org/wiki/Sample\\_mean\\_and\\_covariance#Weighted\\_samples](https://en.wikipedia.org/wiki/Sample_mean_and_covariance#Weighted_samples) for details.

**Usage**

```
weighted_var(x, w = NULL, method = c("unbiased", "ML"),
  na.rm = FALSE, ...)
```

**Arguments**

x	Numeric vector.
w	Optional non-negative, non-missing case weights.
method	Specifies how the result is scaled. If "unbiased", the denominator is reduced by -1, unlike "ML". See <code>stats::cov.wt</code> for details.
na.rm	Should missing values in x be removed? Default is FALSE.
...	Further arguments passed to <code>stats::cov.wt</code> .

**Value**

A length-one numeric vector.

**See Also**

[weighted\\_mean](#).

**Examples**

```
weighted_var(1:10)
weighted_var(1:10, w = NULL)
weighted_var(1:10, w = rep(1, 10))
weighted_var(1:10, w = 1:10)
weighted_var(1:10, w = 1:10, method = "ML")
```

# Index

accuracy, [2](#), [4](#)

AUC, [3](#), [10](#)

classification\_error, [3](#), [4](#)

deviance\_bernoulli, [5](#), [11](#)

deviance\_gamma, [5](#), [8](#)

deviance\_normal, [6](#), [8](#), [13](#), [18](#)

deviance\_poisson, [7](#), [8](#)

deviance\_tweedie, [6–8](#), [8](#)

f1\_score, [9](#), [16](#)

gini\_coefficient, [3](#), [10](#)

logLoss, [5](#), [10](#)

mae, [11](#)

mape, [12](#)

medae, [12](#)

mse, [7](#), [13](#), [17](#), [18](#)

performance, [14](#)

precision, [9](#), [15](#), [16](#)

r\_squared, [18](#)

recall, [9](#), [16](#), [16](#)

rmse, [13](#), [17](#)

weighted\_mean, [19](#), [22](#)

weighted\_median, [20](#), [21](#)

weighted\_quantile, [19](#), [20](#), [20](#)

weighted\_var, [21](#)