Package ‘Metrics’

November 3, 2017

Version 0.1.3

Title Evaluation Metrics for Machine Learning

Description An implementation of evaluation metrics in R that are commonly used in supervised machine learning. It implements metrics for regression, time series, binary classification, classification, and information retrieval problems. It has zero dependencies and a consistent, simple interface for all functions.

Maintainer Michael Frasco <mfrasco6@gmail.com>

Suggests testthat

URL https://github.com/mfrasco/Metrics

BugReports https://github.com/mfrasco/Metrics/issues

License BSD_3_clause + file LICENSE

RoxygenNote 6.0.1

NeedsCompilation no

Author Ben Hamner [aut, cph],
Michael Frasco [aut, cre],
Erin LeDell [ctb]

Repository CRAN

Date/Publication 2017-11-03 08:12:38

R topics documented:

accuracy ................................................. 2
ae ....................................................... 3
ape ..................................................... 3
apk ...................................................... 4
auc ...................................................... 5
bias ..................................................... 6
ce ....................................................... 6
f1 ....................................................... 7
ll ....................................................... 8
**Description**

*accuracy* is defined as the proportion of elements in *actual* that are equal to the corresponding element in *predicted*.

**Usage**

```r
accuracy(actual, predicted)
```

**Arguments**

- `actual` The ground truth vector, where elements of the vector can be any variable type.
- `predicted` The predicted vector, where elements of the vector represent a prediction for the corresponding value in *actual*.

**See Also**

- `ce`
Examples

```r
actual <- c('a', 'a', 'c', 'b', 'c')
predicted <- c('a', 'b', 'c', 'b', 'a')
accuracy(actual, predicted)
```

---

**ae**  
*Absolute Error*

**Description**

ae computes the elementwise absolute difference between two numeric vectors.

**Usage**

```r
ae(actual, predicted)
```

**Arguments**

- `actual`: The ground truth numeric vector.
- `predicted`: The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.

**See Also**

- mae  
- mdae  
- mape

**Examples**

```r
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
ae(actual, predicted)
```

---

**ape**  
*Absolute Percent Error*

**Description**

ape computes the elementwise absolute percent difference between two numeric vectors.

**Usage**

```r
ape(actual, predicted)
```
Arguments

| actual  | The ground truth numeric vector. |
| predicted | The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual. |

Details

\[
\text{ape} = \frac{(\text{actual} - \text{predicted})}{\text{actual}}.
\]
This means that the function will return \(-\infty\), \(\infty\), or \(\text{NaN}\) if \(\text{actual}\) is zero.

See Also

mape, smape

Examples

```r
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
ape(actual, predicted)
```

---

Description

\text{apk} computes the average precision at \(k\), in the context of information retrieval problems.

Usage

\text{apk}(k, \text{actual}, \text{predicted})

Arguments

| k | The number of elements of \text{predicted} to consider in the calculation. |
| actual | The ground truth vector of relevant documents. The vector can contain any numeric or character values, order does not matter, and the vector does not need to be the same length as \text{predicted}. |
| predicted | The predicted vector of retrieved documents. The vector can contain any numeric or character values. However, unlike \text{actual}, order does matter, with the most documents deemed most likely to be relevant at the beginning. |

Details

\text{apk} loops over the first \(k\) values of \text{predicted}. For each value, if the value is contained within \text{actual} and has not been predicted before, we increment the number of successes by one and increment our score by the number of successes divided by \(k\). Then, we return our final score divided by the number of relevant documents (i.e. the length of \text{actual}).

\text{apk} will return \text{NaN} if \text{length(}\text{actual}) equals 0.
See Also

apk f1

Examples

```r
actual <- c('a', 'b', 'd')
predicted <- c('b', 'c', 'a', 'e', 'f')
apk(3, actual, predicted)
```

```
<table>
<thead>
<tr>
<th>auc</th>
<th>Area under the ROC curve (AUC)</th>
</tr>
</thead>
</table>
```

Description

`auc` computes the area under the receiver-operator characteristic curve (AUC).

Usage

`auc(actual, predicted)`

Arguments

- `actual` The ground truth binary numeric vector containing 1 for the positive class and 0 for the negative class.
- `predicted` A numeric vector of predicted values, where the smallest values correspond to the observations most believed to be in the negative class and the largest values indicate the observations most believed to be in the positive class. Each element represents the prediction for the corresponding element in `actual`.

Details

`auc` uses the fact that the area under the ROC curve is equal to the probability that a randomly chosen positive observation has a higher predicted value than a randomly chosen negative value. In order to compute this probability, we can calculate the Mann-Whitney U statistic. This method is very fast, since we do not need to compute the ROC curve first.

Examples

```r
actual <- c(1, 1, 0, 0, 0)
predicted <- c(0.9, 0.8, 0.4, 0.5, 0.3, 0.2)
auc(actual, predicted)
```
### bias

**Bias**

**Description**

bias computes the average amount by which actual is greater than predicted.

**Usage**

`bias(actual, predicted)`

**Arguments**

- `actual`: The ground truth numeric vector.
- `predicted`: The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.

**Details**

If a model is unbiased `bias(actual, predicted)` should be close to zero. Bias is calculated by taking the average of `(actual - predicted)`.

**See Also**

`percent_bias`

**Examples**

```r
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
bias(actual, predicted)
```

### ce

**Classification Error**

**Description**

ce is defined as the proportion of elements in actual that are not equal to the corresponding element in predicted.

**Usage**

`ce(actual, predicted)`

**Examples**

```r
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
```
Arguments

actual  The ground truth vector, where elements of the vector can be any variable type.
predicted  The predicted vector, where elements of the vector represent a prediction for the corresponding value in actual.

See Also

accuracy

Examples

actual <- c('a', 'a', 'c', 'b', 'c')
predicted <- c('a', 'b', 'c', 'b', 'a')
ce(actual, predicted)

<table>
<thead>
<tr>
<th>f1</th>
<th>F1 Score</th>
</tr>
</thead>
</table>

Description

f1 computes the F1 Score in the context of information retrieval problems.

Usage

f1(actual, predicted)

Arguments

actual  The ground truth vector of relevant documents. The vector can contain any numeric or character values, order does not matter, and the vector does not need to be the same length as predicted.
predicted  The predicted vector of retrieved documents. The vector can contain any numeric or character values, order does not matter, and the vector does not need to be the same length as actual.

Details

f1 is defined as $2 \cdot \text{precision} \cdot \text{recall} / (\text{precision} + \text{recall})$. In the context of information retrieval problems, precision is the proportion of retrieved documents that are relevant to a query and recall is the proportion of relevant documents that are successfully retrieved by a query. If there are zero relevant documents that are retrieved, zero relevant documents, or zero predicted documents, f1 is defined as 0.

See Also

apk mapk
logLoss

Examples

```r
actual <- c('a', 'c', 'd')
predicted <- c('d', 'e')
fl(actual, predicted)
```

<table>
<thead>
<tr>
<th>Log Loss</th>
</tr>
</thead>
</table>

Description

`fl` computes the elementwise log loss between two numeric vectors.

Usage

`fl(actual, predicted)`

Arguments

- `actual` The ground truth binary numeric vector containing 1 for the positive class and 0 for the negative class.
- `predicted` A numeric vector of predicted values, where the values correspond to the probabilities that each observation in `actual` belongs to the positive class.

See Also

`logloss`

Examples

```r
actual <- c(1, 1, 1, 0, 0, 0)
predicted <- c(0.9, 0.8, 0.4, 0.5, 0.3, 0.2)
fl(actual, predicted)
```

logLoss

Mean Log Loss

Description

`logLoss` computes the average log loss between two numeric vectors.

Usage

`logLoss(actual, predicted)`
mae

Arguments

actual The ground truth binary numeric vector containing 1 for the positive class and 0 for the negative class.
predicted A numeric vector of predicted values, where the values correspond to the probabilities that each observation in actual belongs to the positive class.

See Also

Examples

actual <- c(1, 1, 1, 0, 0, 0)
predicted <- c(0.9, 0.8, 0.4, 0.5, 0.3, 0.2)
logLoss(actual, predicted)

mae

Mean Absolute Error

Description

mae computes the average absolute difference between two numeric vectors.

Usage

mae(actual, predicted)

Arguments

actual The ground truth numeric vector.
predicted The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.

See Also

mdae mape

Examples

actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
mae(actual, predicted)
mape

*Mean Absolute Percent Error*

**Description**

mape computes the average absolute percent difference between two numeric vectors.

**Usage**

```r
mape(actual, predicted)
```

**Arguments**

- `actual` The ground truth numeric vector.
- `predicted` The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in `actual`.

**Details**

mape is calculated as the average of \((\text{actual} - \text{predicted}) / \text{actual}\). This means that the function will return \(-\infty, \infty, \text{or} \text{NaN}\) if `actual` is zero. Due to the instability at or near zero, smape or mase are often used as alternatives.

**See Also**

mae, smape, mase

**Examples**

```r
actual <- c(1.1, 1.9, 3.0, 4.4, 5.8, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
mape(actual, predicted)
```

mapk

*Mean Average Precision at k*

**Description**

mapk computes the mean average precision at k for a set of predictions, in the context of information retrieval problems.

**Usage**

```r
mapk(k, actual, predicted)
```
**Arguments**

- **k**
  - The number of elements of `predicted` to consider in the calculation.

- **actual**
  - A list of vectors, where each vector represents a ground truth vector of relevant documents. In each vector, the elements can be numeric or character values, and the order of the elements does not matter.

- **predicted**
  - A list of vectors, where each vector represents the predicted vector of retrieved documents for the corresponding element of `actual`. In each vector, the order of the elements does matter, with the elements believed most likely to be relevant at the beginning.

**Details**

`mapk` evaluates `apk` for each pair of elements from `actual` and `predicted`.

**See Also**

- `apk` `f1`

**Examples**

```r
actual <- list(c('a', 'b'), c('a', 'c', 'd'), c('x', 'y', 'b'))
predicted <- list(c('a', 'c', 'd'), c('x', 'b', 'a', 'b'), c('y'))
mapk(2, actual, predicted)

actual <- list(c(1, 5, 7, 9), c(2, 3), c(2, 5, 6))
predicted <- list(c(5, 6, 7, 8, 9), c(1, 2, 3), c(2, 4, 6, 8))
mapk(3, actual, predicted)
```

---

**mase**

**Mean Absolute Scaled Error**

**Description**

`mase` computes the mean absolute scaled error between two numeric vectors. This function is only intended for time series data, where `actual` and `numeric` are numeric vectors ordered by time.

**Usage**

```r
mase(actual, predicted, step_size = 1)
```

**Arguments**

- **actual**
  - The ground truth numeric vector ordered in time, with most recent observation at the end of the vector.

- **predicted**
  - The predicted numeric vector ordered in time, where each element of the vector represents a prediction for the corresponding element of `actual`. 
### mdae

**Median Absolute Error**

**Description**

`mdae` computes the median absolute difference between two numeric vectors.

**Usage**

```r
mdae(actual, predicted)
```

**Arguments**

- `actual` The ground truth numeric vector.
- `predicted` The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in `actual`.

**See Also**

`mae mape`

**Examples**

```r
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
mdae(actual, predicted)
```
MeanQuadraticWeightedKappa

Mean Quadratic Weighted Kappa

Description

MeanQuadraticWeightedKappa computes the mean quadratic weighted kappa, which can optionally be weighted.

Usage

MeanQuadraticWeightedKappa(kappas, weights = rep(1, length(kappas)))

Arguments

kappas A numeric vector of possible kappas.
weights An optional numeric vector of ratings.

See Also

ScoreQuadraticWeightedKappa

Examples

kappas <- c(0.3, 0.2, 0.2, 0.5, 0.1, 0.2)
weights <- c(1.0, 2.5, 1.0, 1.0, 2.0, 3.0)
MeanQuadraticWeightedKappa(kappas, weights)

mse

Mean Squared Error

Description

mse computes the average squared difference between two numeric vectors.

Usage

mse(actual, predicted)

Arguments

actual The ground truth numeric vector.
predicted The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.
See Also

rmse mae

Examples

actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
msle(actual, predicted)

msle

Mean Squared Log Error

Description

msle computes the average of squared log error between two numeric vectors.

Usage

msle(actual, predicted)

Arguments

actual The ground truth non-negative vector
predicted The predicted non-negative vector, where each element in the vector is a prediction for the corresponding element in actual.

Details

msle adds one to both actual and predicted before taking the natural logarithm to avoid taking the natural log of zero. As a result, the function can be used if actual or predicted have zero-valued elements. But this function is not appropriate if either are negative valued.

See Also

rmsle sle

Examples

actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
msle(actual, predicted)
**Inherit Documentation for Binary Classification Metrics**

**Description**

This object provides the documentation for the parameters of functions that provide binary classification metrics

**Arguments**

- `actual`  
  The ground truth binary numeric vector containing 1 for the positive class and 0 for the negative class.

- `predicted`  
  The predicted binary numeric vector containing 1 for the positive class and 0 for the negative class. Each element represents the prediction for the corresponding element in `actual`.

**Inherit Documentation for Classification Metrics**

**Description**

This object provides the documentation for the parameters of functions that provide classification metrics

**Arguments**

- `actual`  
  The ground truth vector, where elements of the vector can be any variable type.

- `predicted`  
  The predicted vector, where elements of the vector represent a prediction for the corresponding value in `actual`.

**Inherit Documentation for Regression Metrics**

**Description**

This object provides the documentation for the parameters of functions that provide regression metrics

**Arguments**

- `actual`  
  The ground truth numeric vector.

- `predicted`  
  The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in `actual`.

percent_bias | Percent Bias

Description

percent_bias computes the average amount that actual is greater than predicted as a percentage of actual.

Usage

percent_bias(actual, predicted)

Arguments

- actual: The ground truth numeric vector.
- predicted: The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.

Details

If a model is unbiased percent_bias(actual, predicted) should be close to zero. Percent Bias is calculated by taking the average of (actual - predicted) / actual.

percent_bias will give -Inf, Inf, or NaN, if any elements of actual are 0.

See Also

bias

Examples

```
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
percent_bias(actual, predicted)
```

rae | Relative Absolute Error

Description

rae computes the relative absolute error between two numeric vectors.

Usage

rae(actual, predicted)
Arguments

```
actual The ground truth numeric vector.
predicted The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.
```

Details

**rae** divides \( \text{sum}(\text{ae}(\text{actual}, \text{predicted})) \) by \( \text{sum}(\text{ae}(\text{actual}, \text{mean}(\text{actual}))) \), meaning that it provides the absolute error of the predictions relative to a naive model that predicted the mean for every data point.

See Also

**rse rse**

Examples

```
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
rmse(actual, predicted)
```

---

**rmse**

*Root Mean Squared Error*

Description

**rmse** computes the root mean squared error between two numeric vectors

Usage

```
rmse(actual, predicted)
```

Arguments

```
actual The ground truth numeric vector.
predicted The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.
```

See Also

**mse**

Examples

```
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
rmse(actual, predicted)
```
rmsle

Root Mean Squared Log Error

Description

rmsle computes the root mean squared log error between two numeric vectors.

Usage

rmsle(actual, predicted)

Arguments

actual The ground truth non-negative vector
predicted The predicted non-negative vector, where each element in the vector is a prediction for the corresponding element in actual.

Details

rmsle adds one to both actual and predicted before taking the natural logarithm to avoid taking the natural log of zero. As a result, the function can be used if actual or predicted have zero-valued elements. But this function is not appropriate if either are negative valued.

See Also

msle sle

Examples

actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
rmsle(actual, predicted)

rrse

Root Relative Squared Error

Description

rrse computes the root relative squared error between two numeric vectors.

Usage

rrse(actual, predicted)
rse

Arguments

actual  The ground truth numeric vector.
predicted  The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.

Details

rse takes the square root of sse(actual, predicted) divided by sse(actual, mean(actual)), meaning that it provides the squared error of the predictions relative to a naive model that predicted the mean for every data point.

See Also

rse, rae

Examples

actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
rse(actual, predicted)

---

rse  Relative Squared Error

Description

rse computes the relative squared error between two numeric vectors.

Usage

rse(actual, predicted)

Arguments

actual  The ground truth numeric vector.
predicted  The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.

Details

rse divides sse(actual, predicted) by sse(actual, mean(actual)), meaning that it provides the squared error of the predictions relative to a naive model that predicted the mean for every data point.

See Also

rrse, rae

---
ScoreQuadraticWeightedKappa

Quadratic Weighted Kappa

Examples

```r
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
rse(actual, predicted)
```

Description

ScoreQuadraticWeightedKappa computes the quadratic weighted kappa between two vectors of integers.

Usage

```r
ScoreQuadraticWeightedKappa(rater.a, rater.b, min.rating = min(c(rater.a, rater.b)), max.rating = max(c(rater.a, rater.b)))
```

Arguments

- `rater.a` An integer vector of the first rater’s ratings.
- `rater.b` An integer vector of the second rater’s ratings.
- `min.rating` The minimum possible rating.
- `max.rating` The maximum possible rating.

See Also

- `MeanQuadraticWeightedKappa`

Examples

```r
rater.a <- c(1, 4, 5, 5, 2, 1)
rater.b <- c(2, 2, 4, 5, 3, 3)
ScoreQuadraticWeightedKappa(rater.a, rater.b, 1, 5)
```
se                      Squared Error

Description
se computes the elementwise squared difference between two numeric vectors.

Usage
se(actual, predicted)

Arguments
  actual        The ground truth numeric vector.
  predicted     The predicted numeric vector, where each element in the vector is a prediction
                 for the corresponding element in actual.

See Also
mse rmse

Examples
actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
se(actual, predicted)

sle                      Squared Log Error

Description
sle computes the elementwise squares of the differences in the logs of two numeric vectors.

Usage
sle(actual, predicted)

Arguments
  actual        The ground truth non-negative vector.
  predicted     The predicted non-negative vector, where each element in the vector is a prediction
                 for the corresponding element in actual.
Details

sle adds one to both actual and predicted before taking the natural logarithm of each to avoid taking the natural log of zero. As a result, the function can be used if actual or predicted have zero-valued elements. But this function is not appropriate if either are negative valued.

See Also

msle rmsle

Examples

actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
sle(actual, predicted)

smape

Symmetric Mean Absolute Percentage Error

Description

smape computes the symmetric mean absolute percentage error between two numeric vectors.

Usage

smape(actual, predicted)

Arguments

actual The ground truth numeric vector.
predicted The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in actual.

Details

smape is defined as two times the average of abs(actual - predicted) / (abs(actual) + abs(predicted)). Therefore, at the elementwise level, it will provide NaN only if actual and predicted are both zero. It has an upper bound of 2, when either actual or predicted are zero or when actual and predicted are opposite signs.
smape is symmetric in the sense that smape(x, y) = smape(y, x).

See Also

mape mase

Examples

actual <- c(1.1, 1.9, 3.0, 4.4, 5.0, 5.6)
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
smape(actual, predicted)
**sse**

*Sum of Squared Errors*

**Description**

sse computes the sum of the squared differences between two numeric vectors.

**Usage**

```r
sse(actual, predicted)
```

**Arguments**

- `actual`: The ground truth numeric vector.
- `predicted`: The predicted numeric vector, where each element in the vector is a prediction for the corresponding element in `actual`.

**See Also**

`mse`

**Examples**

```r
c(actual <- c(1.1, 1.9, 3.8, 4.4, 5.0, 5.6))
predicted <- c(0.9, 1.8, 2.5, 4.5, 5.0, 6.2)
sse(actual, predicted)
```
Index

accuracy, 2, 7
ae, 3
ape, 3
apk, 4, 5, 7, 11
auc, 5
bias, 6, 16
ce, 2, 6
f1, 5, 7, 11
ll, 8, 9
logloss, 8, 8
mae, 3, 9, 10, 12, 14
mape, 3, 4, 9, 10, 12, 22
mapk, 7, 10
mase, 10, 11, 22
mdae, 3, 9, 12
MeanQuadraticWeightedKappa, 13, 20
mse, 13, 17, 21, 23
msle, 14, 18, 22
params_binary, 15
params_classification, 15
params_regression, 15
percent_bias, 6, 16
rae, 16, 19
rmse, 14, 17, 21
rmsle, 14, 18, 22
rrse, 17, 18, 19
rse, 17, 19, 19
ScoreQuadraticWeightedKappa, 13, 20
se, 21
sle, 14, 18, 21
smape, 10, 12, 22
sse, 23