Package ‘wconf’

April 23, 2024

Type Package

Title Weighted Confusion Matrix

Version 1.1.0

URL https://www.alexandrumonahov.eu.org/projects

Description Allows users to create weighted confusion matrices and accuracy metrics that help with the model selection process for classification problems, where distance from the correct category is important. The package includes several weighting schemes which can be parameterized, as well as custom configuration options. Furthermore, users can decide whether they wish to positively or negatively affect the accuracy score as a result of applying weights to the confusion matrix. Functions are included to calculate accuracy metrics for imbalanced data. Finally, ‘wconf’ integrates well with the ‘caret’ package, but it can also work standalone when provided data in matrix form.

References:

License CC BY-SA 4.0

Encoding UTF-8

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, caret

VignetteBuilder knitr

NeedsCompilation no

Author Alexandru Monahov [aut, cre, cph]
(<https://orcid.org/0000-0001-6204-9131>)
Description
This function calculates classification accuracy scores using the sine-based formulas proposed by Starovoitov and Golub (2020). The advantage of the new method consists in producing improved results when compared with the standard balanced accuracy function, by taking into account the class distribution of errors.

Usage
balancedaccuracy(m, print.scores = TRUE)

Arguments
m the caret confusion matrix object or simple matrix.
print.scores print the accuracy metrics.

Details
The input object "m" should be a square matrix of at least size 2x2.

Value
a list containing 5 elements: 3 overall and 2 class accuracy scores

Author(s)
Alexandru Monahov, <https://www.alexandrumonahov.eu.org/>

See Also
[wconfusionmatrix()]
Examples

```r
m = matrix(c(70,0,0,10,10,0,5,3,2), ncol = 3, nrow=3)
balancedaccuracy(m, print.scores = TRUE)
```

Description

This function calculates the weighted confusion matrix from a caret ConfusionMatrix object or a simple matrix, according to one of several weighting schemas and optionally prints the weighted accuracy score.

Usage

```r
wconfusionmatrix(m, weight.type = "arithmetic", weight.penalty = FALSE, standard.deviation = 2, geometric.multiplier = 2, interval.high=1, interval.low = -1, sin.high=1.5*pi, sin.low = 0.5*pi, tanh.decay = 3, custom.weights = NA, print.weighted.accuracy = FALSE)
```

Arguments

- `m` the caret confusion matrix object or simple matrix.
- `weight.type` the weighting schema to be used. Can be one of: "arithmetic" - a decreasing arithmetic progression weighting scheme, "geometric" - a decreasing geometric progression weighting scheme, "normal" - weights drawn from the right tail of a normal distribution, "interval" - weights contained on a user-defined interval, "sin" - a weighing scheme based on a sine function, "tanh" - a weighing scheme based on a hyperbolic tangent function, "custom" - custom weight vector defined by the user.
- `weight.penalty` determines whether the weights associated with non-diagonal elements generated by the "normal", "arithmetic" and "geometric" weight types are positive or negative values. By default, the value is set to FALSE, which means that generated weights will be positive values.
- `standard.deviation` standard deviation of the normal distribution, if the normal distribution weighting schema is used.
- `geometric.multiplier` the multiplier used to construct the geometric progression series, if the geometric progression weighting scheme is used.
interval.high the upper bound of the weight interval, if the interval weighting scheme is used.
interval.low the lower bound of the weight interval, if the interval weighting scheme is used.
sin.high the upper segment of the sine function to be used in the weighting scheme.
sin.low the lower segment of the sine function to be used in the weighting scheme.
tanh.decay the decay factor of the hyperbolic tangent weighing function. Higher values increase the rate of decay and place less weight on observations farther away from the correctly predicted category.
custom.weights the vector of custom weightsto be applied, is the custom weighting scheme was selected. The vector should be equal to "n", but can be larger, with excess values being ignored.
print.weighted.accuracy print the weighted accuracy metric, which represents the sum of all weighted confusion matrix cells divided by the total number of observations.

Details
The number of categories "n" should be greater or equal to 2.

Value
an nxn weighted confusion matrix

Author(s)
Alexandru Monahov, <https://www.alexandrumonahov.eu.org/>

See Also
[weightmatrix()] for the weight matrix used in computations, [balancedaccuracy()] for accuracy metrics designed for imbalanced data.

Examples
m = matrix(c(70,0,0,10,0,5,3,2), ncol = 3, nrow=3)
wconfusionmatrix(m, weight.type="arithmetic", print.weighted.accuracy = TRUE)
wconfusionmatrix(m, weight.type="geometric", print.weighted.accuracy = TRUE)
wconfusionmatrix(m, weight.type="interval", print.weighted.accuracy = TRUE)
wconfusionmatrix(m, weight.type="normal", print.weighted.accuracy = TRUE)
wconfusionmatrix(m, weight.type="sin", print.weighted.accuracy = TRUE)
wconfusionmatrix(m, weight.type="tanh", print.weighted.accuracy = TRUE)
wconfusionmatrix(m, weight.type= "custom", custom.weights = c(1,0.1,0),
print.weighted.accuracy = TRUE)
weightmatrix

weightmatrix

Weight matrix

Description

This function compiles a weight matrix according to one of several weighting schemas and allows users to visualize the impact of the weight matrix on each element of the confusion matrix.

Usage

weightmatrix(n, weight.type = "arithmetic", weight.penalty = FALSE,
standard.deviation = 2,
geometric.multiplier = 2,
interval.high = 1, interval.low = -1,
sin.high = 1.5 * pi, sin.low = 0.5 * pi,
tanh.decay = 3,
custom.weights = NA,
plot.weights = FALSE)

Arguments

n
the number of classes contained in the confusion matrix.

weight.type
the weighting schema to be used. Can be one of: "arithmetic" - a decreasing arithmetic progression weighting scheme, "geometric" - a decreasing geometric progression weighting scheme, "normal" - weights drawn from the right tail of a normal distribution, "interval" - weights contained on a user-defined interval, "sin" - a weighing scheme based on a sine function, "tanh" - a weighing scheme based on a hyperbolic tangent function, "custom" - custom weight vector defined by the user.

weight.penalty
determines whether the weights associated with non-diagonal elements generated by the "normal", "arithmetic" and "geometric" weight types are positive or negative values. By default, the value is set to FALSE, which means that generated weights will be positive values.

standard.deviation
standard deviation of the normal distribution, if the normal distribution weighting schema is used.

geometric.multiplier
the multiplier used to construct the geometric progression series, if the geometric progression weighting scheme is used.

interval.high
the upper bound of the weight interval, if the interval weighting scheme is used.

interval.low
the lower bound of the weight interval, if the interval weighting scheme is used.

sin.high
the upper segment of the sine function to be used in the weighting scheme.

sin.low
the lower segment of the sine function to be used in the weighting scheme.
tanh.decay the decay factor of the hyperbolic tangent weighing function. Higher values increase the rate of decay and place less weight on observations farther away from the correctly predicted category.

custom.weights the vector of custom weights to be applied, is the custom weighting scheme was selected. The vector should be equal to "n", but can be larger, with excess values being ignored.

plot.weights optional setting to enable plotting of weight vector, corresponding to the first column of the weight matrix

Details

The number of categories "n" should be greater or equal to 2.

Value

an nxn matrix, containing the weights to be multiplied with the confusion matrix.

Author(s)

Alexandru Monahov, <https://www.alexandrumonahov.eu.org/> 

See Also

[wconfusionmatrix()]

Examples

weightmatrix(n=4, weight.type="arithmetic", plot.weights = TRUE)
weightmatrix(n=4, weight.type="normal", standard.deviation = 1, plot.weights = TRUE)
weightmatrix(n=4, weight.type="interval", interval.high = 1, interval.low = -0.5, plot.weights = TRUE)
weightmatrix(n=4, weight.type="geometric", geometric.multiplier = 0.6)
weightmatrix(n=10, weight.type="sin", sin.low = 0, sin.high = pi, plot.weights = TRUE)
weightmatrix(n=10, weight.type="tanh", tanh.decay = 5, plot.weights = TRUE)
weightmatrix(n=4, weight.type="custom", custom.weights = c(1,0.2,0.1,0), plot.weights = TRUE)
Index

* accuracy
  balancedaccuracy, 2
  wconfusionmatrix, 3
* arithmetic
  weightmatrix, 5
* balanced
  balancedaccuracy, 2
* confusion
  wconfusionmatrix, 3
* geometric
  weightmatrix, 5
* interval
  weightmatrix, 5
* matrix
  wconfusionmatrix, 3
  weightmatrix, 5
* metric
  balancedaccuracy, 2
* normal
  weightmatrix, 5
* progression
  weightmatrix, 5
* score
  balancedaccuracy, 2
  wconfusionmatrix, 3
* sine
  balancedaccuracy, 2
* weighted
  wconfusionmatrix, 3
* weight
  weightmatrix, 5
balancedaccuracy, 2
wconfusionmatrix, 3
weightmatrix, 5