

# Package: RcppColMetric (via r-universe)

May 25, 2026

**Title** Efficient Column-Wise Metric Computation Against Common Vector

**Version** 0.1.0

**Description** In data science, it is a common practice to compute a series of columns (e.g. features) against a common response vector. Various metrics are provided with efficient computation implemented with 'Rcpp'.

**License** MIT + file LICENSE

**Suggests** caTools, infotheo, magrittr, MASS, microbenchmark, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**URL** <https://github.com/zhuxr11/RcppColMetric>

**BugReports** <https://github.com/zhuxr11/RcppColMetric/issues>

**LinkingTo** Rcpp

**Imports** Rcpp

**Repository** <https://zhuxr11.r-universe.dev>

**Date/Publication** 2025-03-10 16:39:05 UTC

**RemoteUrl** <https://github.com/zhuxr11/rcppcolmetric>

**RemoteRef** HEAD

**RemoteSha** da07c00282b111abf8e71512808e4eb2d9e31bb7

## Contents

col_auc	2
col_auc_vec	3
col_mut_info	4
col_mut_info_vec	5

<b>Index</b>	<b>7</b>
--------------	----------

---

col_auc	<i>Column-wise area under ROC curve (AUC)</i>
---------	---

---

### Description

Calculate area under the ROC curve (AUC) for every column of a matrix or data frame. For better performance, data frame is preferred.

### Usage

```
col_auc(x, y, args = NULL)
```

### Arguments

x	Matrix or data frame. Rows contain samples and columns contain features/variables.
y	Factor of class labels for the data samples. A response vector with one label for each row/component of x.
args	NULL (default) or list of named arguments:  <b>direction</b> Character vector containing one of the following directions: ">", "<" or "auto" (default), recycled for each feature so different directions can be used for different features.

### Value

An output is a single matrix with the same number of columns as X and "n choose 2" ( $n!/((n-2)! 2!)$  =  $n(n-1)/2$ ) number of rows, where n is number of unique labels in y list. For example, if y contains only two unique class labels (  $\text{length}(\text{unique}(\text{lab}))=2$  ) then output matrix will have a single row containing AUC of each column. If more than two unique labels are present than AUC is calculated for every possible pairing of classes ("n choose 2" of them).

### Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

### See Also

`caTools::colAUC` for the original R implementation.

[col\\_auc\\_vec](#) for the vectorized version.

## Examples

```
if (require("MASS", quietly = TRUE) == TRUE) {
  data(cats)
  print(res_cpp <- col_auc(cats[, 2L:3L], cats[, 1L]))
  # Validate with caTools::colAUC()
  if (require("caTools", quietly = TRUE) == TRUE) {
    print(res_r <- caTools::colAUC(cats[, 2L:3L], cats[, 1L]))
    identical(res_cpp, res_r)
  }
}
```

---

col\_auc\_vec

*Vectorized version of function col\_auc*

---

## Description

This is the vectorized version of [col\\_auc](#).

## Usage

```
col_auc_vec(x, y, args = NULL)
```

## Arguments

x	List, where each element is an input to x in <a href="#">col_auc</a> .
y	List, where each element is an input to y in <a href="#">col_auc</a> .
args	NULL (default) or list where each element is an input to args in <a href="#">col_auc</a> .

## Value

List, where each element is an output from [col\\_auc](#).

## Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

## See Also

[col\\_auc](#) for the non-vectorized version.

**Examples**

```

if (require("MASS", quietly = TRUE) == TRUE) {
  data(cats)
  print(res_cpp <- col_auc_vec(list(cats[, 2L:3L]), list(cats[, 1L])))
  # Validate with caTools::colAUC()
  if (require("caTools", quietly = TRUE) == TRUE) {
    print(res_r <- caTools::colAUC(cats[, 2L:3L], cats[, 1L]))
    identical(res_cpp, list(res_r))
  }
}

```

---

col\_mut\_info

*Column-wise mutual information*


---

**Description**

Calculate mutual information for every column of a matrix or data frame. Only discrete values are allowed. For better performance, data frame is preferred.

**Usage**

```
col_mut_info(x, y, args = NULL)
```

**Arguments**

x	Matrix or data frame of discrete values (integers). Rows contain samples and columns contain features/variables.
y	Factor of class labels for the data samples. A response vector with one label for each row/component of x.
args	NULL (default) or list of named arguments: <b>method</b> Integer indicating computation method: 0 = empirical, 1 = Miller-Madow, 2 = shrink, 3 = Schurmann-Grassberger.

**Value**

An output is a single matrix with the same number of columns as X and 1 row.

**Note**

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

**See Also**

`infotheo::mutinformation` for the original computation of mutual information in R (and also the computation methods).

`col_mut_info_vec` for the vectorized version.

## Examples

```
if (require("MASS", quietly = TRUE) == TRUE) {
  data(cats)
  print(res_cpp <- col_mut_info(round(cats[, 2L:3L]), cats[, 1L]))
  # Validate with caTools::colAUC()
  if ((require("infotheo", quietly = TRUE) == TRUE) &&
      (require("magrittr", quietly = TRUE) == TRUE)) {
    print(res_r <- sapply(round(cats[, 2L:3L]), infotheo::mutinformation, cats[, 1L]) %>%
      {matrix(., nrow = 1L, dimnames = list(NULL, names(.)))})
    identical(res_cpp, res_r)
  }
}
```

---

col_mut_info_vec	<i>Vectorized version of function col_mut_info</i>
------------------	--

---

## Description

This is the vectorized version of [col\\_mut\\_info](#).

## Usage

```
col_mut_info_vec(x, y, args = NULL)
```

## Arguments

x	List, where each element is an input to x in <a href="#">col_mut_info</a> .
y	List, where each element is an input to y in <a href="#">col_mut_info</a> .
args	NULL (default) or list where each element is an input to args in <a href="#">col_mut_info</a> .

## Value

List, where each element is an output from [col\\_mut\\_info](#).

## Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

## See Also

[col\\_mut\\_info](#) for the non-vectorized version.

**Examples**

```
if (require("MASS", quietly = TRUE) == TRUE) {
  data(cats)
  print(res_cpp <- col_mut_info_vec(list(round(cats[, 2L:3L])), list(cats[, 1L])))
  # Validate with caTools::colAUC()
  if ((require("infotheo", quietly = TRUE) == TRUE) &&
      (require("magrittr", quietly = TRUE) == TRUE)) {
    print(res_r <- sapply(round(cats[, 2L:3L]), infotheo::mutinformation, cats[, 1L]) %>%
      {matrix(., nrow = 1L, dimnames = list(NULL, names(.)))})
    identical(res_cpp, list(res_r))
  }
}
```

# Index

col\_auc, [2](#), [3](#)

col\_auc\_vec, [2](#), [3](#)

col\_mut\_info, [4](#), [5](#)

col\_mut\_info\_vec, [4](#), [5](#)