

Package ‘plotROC’

August 25, 2025

Type Package

Title Generate Useful ROC Curve Charts for Print and Interactive Use

Version 2.3.3

Date 2025-08-22

Maintainer Michael C. Sachs <sachsmc@gmail.com>

Description Most ROC curve plots obscure the cutoff values and inhibit interpretation and comparison of multiple curves. This attempts to address those shortcomings by providing plotting and interactive tools. Functions are provided to generate an interactive ROC curve plot for web use, and print versions. A Shiny application implementing the functions is also included.

License MIT + file LICENSE

Encoding UTF-8

URL <https://sachsmc.github.io/plotROC/>

BugReports <https://github.com/sachsmc/plotROC/issues/>

Depends R (>= 3.0.0), ggplot2

Imports methods, grid, gridSVG, shiny, plyr, rlang

Suggests knitr, testthat, stringr, survivalROC, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation no

Author Michael C. Sachs [aut, cre],
Robert W. Corty [ctb],
Luis Crouch [ctb] (Modification to calc_auc)

Repository CRAN

Date/Publication 2025-08-25 11:30:03 UTC

Contents

calculate_multi_roc	2
calculate_roc	3
calc_auc	3
direct_label	4
export_interactive_roc	5
GeomRoc	6
geom_rocci	9
getD3	12
ggroc	12
melt_roc	13
multi_ggroc	14
plotROC	15
plot_interactive_roc	16
plot_journal_roc	16
roc_key	17
shiny_plotROC	18
StatRoc	18
StatRocci	21
style_roc	24
verify_d	25
Index	26

calculate_multi_roc	<i>Calculate the Empirical ROC curves for multiple biomarkers</i>
---------------------	---

Description

Deprecated, use [geom_roc](#) instead

Usage

calculate_multi_roc(data, M_string, D_string)

Arguments

data	data frame containing at least 1 marker and the common class labels, coded as 0 and 1
M_string	vector of marker column names
D_string	class label column name

Value

List of data frames containing cutoffs, and estimated true and false positive fractions

calculate_roc	<i>Calculate the Empirical ROC curve</i>
---------------	--

Description

Deprecated, use [geom_roc](#) instead

Usage

```
calculate_roc(M, D, ci = FALSE, alpha = 0.05)
```

Arguments

M	continuous marker values or predictions of class labels
D	class labels, must be coded as 0 and 1. If not numeric with 0/1, then plotROC assumes the first level in sort order is healthy status, with a warning.
ci	Logical, if true, will calculate exact joint confidence regions for the TPF and FPF
alpha	Confidence level, ignored if ci = FALSE

Details

Confidence intervals for TPF and FPF are calculated using the exact method of Clopper and Pearson (1934) each at the level $1 - \sqrt{1 - \alpha}$. Based on result 2.4 from Pepe (2003), the cross-product of these intervals yields a $1 - \alpha$

Value

A dataframe containing cutoffs, estimated true and false positive fractions, and confidence intervals if ci = TRUE.

calc_auc	<i>Calculate the Area under the ROC curve</i>
----------	---

Description

Given a ggplot object with a GeomRoc layer, computes the area under the ROC curve for each group

Usage

```
calc_auc(ggroc)
```

Arguments

ggroc	A ggplot object that contains a GeomRoc layer
-------	---

Value

A data frame with the estimated AUCs for each layer, panel and group

Examples

```
D.ex <- rbinom(50, 1, .5)
rocdata <- data.frame(D = c(D.ex, D.ex),
                      M = c(rnorm(50, mean = D.ex, sd = .4), rnorm(50, mean = D.ex, sd = 1)),
                      Z = c(rep("A", 50), rep("B", 50)))

ggroc <- ggplot(rocdata, aes(m = M, d = D)) + geom_roc()
calc_auc(ggroc)
ggroc2 <- ggplot(rocdata, aes(m = M, d = D, color = Z)) + geom_roc()
calc_auc(ggroc2)
```

direct_label	<i>Add direct labels to a ROC plot</i>
--------------	--

Description

Add direct labels to a ROC plot

Usage

```
direct_label(
  ggroc_p,
  labels = NULL,
  label.angle = 45,
  nudge_x = 0,
  nudge_y = 0,
  size = 6,
  ...
)
```

Arguments

ggroc_p	A ggplot object that contains a geom_roc layer
labels	vector of labels to add directly to the plot next to the curves. If multiple curves, must be in the same order as the grouping factor. If NULL, attempts to determine labels from the ggroc_p object
label.angle	angle of adjustment for the direct labels
nudge_x, nudge_y	Horizontal and vertical adjustment to nudge labels by. These can be scalars or vectors the same length as the number of labels
size	Size of labels
...	Other arguments passed to annotate

export_interactive_roc

Generate svg code for an ROC curve object

Description

Takes a ggplot object that contains a GeomRoc layer and returns a string that contains html suitable for creating a standalone interactive ROC curve plot.

Usage

```
export_interactive_roc(
  ggroc_p,
  add.cis = TRUE,
  hide.points = FALSE,
  prefix = "a",
  width = 6,
  height = 6,
  omit.js = FALSE,
  style = style_roc(theme = theme_grey()),
  ...
)
```

Arguments

ggroc_p	A ggplot object with a GeomRoc layer and optionally a GeomRocci layer as returned by geom_roc and/or geom_rocci . It can be modified with annotations, themes, etc.
add.cis	Logical, if true, removes the current confidence interval layer (if present) and replaces it with a denser layer of confidence regions
hide.points	Logical, if true, hides points layer so that points with cutoff values are only visible when hovering. Recommended for plots containing more than 3 curves.
prefix	A string to assign to the objects within the svg. Enables unique identification by the javascript code
width, height	Width and height in inches of plot
omit.js	Logical. If true, omit inclusion of javascript source in output. Useful for documents with multiple interactive plots
style	A call to the function style_roc
...	Other arguments passed to geom_rocci when add.cis = TRUE

Details

If you intend to include more than one of these objects in a single page, use a different prefix string for each one. To use this function in knitr, use the chunk options `fig.keep='none'` and `results='asis'`, then `cat()` the resulting string to the output. See the vignette for examples. Older browsers (< IE7) are not supported.

Value

A character object containing the html necessary to plot the ROC curve in a web browser

GeomRoc

Empirical Receiver Operating Characteristic Curve

Description

Display the empirical ROC curve. Useful for characterizing the classification accuracy of continuous measurements for predicting binary states

Usage

GeomRoc

```
geom_roc(
  mapping = NULL,
  data = NULL,
  stat = "roc",
  n.cuts = 10,
  arrow = NULL,
  lineend = "butt",
  linejoin = "round",
  linemitre = 1,
  linealpha = 1,
  pointalpha = 1,
  pointsize = 0.5,
  labels = TRUE,
  labelsize = 3.88,
  labelround = 1,
  na.rm = TRUE,
  cutoffs.at = NULL,
  cutoff.labels = NULL,
  position = "identity",
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

mapping Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.

<code>data</code>	The data to be displayed in this layer. There are three options: If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code>. A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
<code>stat</code>	Use to override the default connection between <code>geom_roc</code> and <code>stat_roc</code> .
<code>n.cuts</code>	Number of cutpoints to display along each curve
<code>arrow</code>	Arrow specification, as created by arrow
<code>lineend</code>	Line end style (round, butt, square)
<code>linejoin</code>	Line join style (round, mitre, bevel)
<code>linemitre</code>	Line mitre limit (number greater than 1)
<code>linealpha</code>	Alpha level for the lines, <code>alpha.line</code> is deprecated
<code>pointalpha</code>	Alpha level for the cutoff points, <code>alpha.point</code> is deprecated
<code>pointsize</code>	Size of cutoff points, <code>size.point</code> is deprecated
<code>labels</code>	Logical, display cutoff text labels
<code>labelsize</code>	Size of cutoff text labels
<code>labelround</code>	Integer, number of significant digits to round cutoff labels
<code>na.rm</code>	Remove missing values from curve
<code>cutoffs.at</code>	Vector of user supplied cutoffs to plot as points. If non- <code>NULL</code> , it will override the values of <code>n.cuts</code> and plot the observed cutoffs closest to the user-supplied ones.
<code>cutoff.labels</code>	vector of user-supplied labels for the cutoffs. Must be a character vector of the same length as <code>cutoffs.at</code> .
<code>position</code>	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following: • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the <code>position_</code> prefix. For example, to use <code>position_jitter()</code>, give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
<code>show.legend</code>	logical. Should this layer be included in the legends? <code>NA</code>, the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use <code>TRUE</code>. If <code>NA</code>, all levels are shown in legend, but unobserved levels are omitted.

`inherit.aes` If FALSE, overrides the default aesthetics, rather than combining with them.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `...`. Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the geom part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the stat part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `...`. This can be one of the functions described as [key glyphs](#), to change the display of the layer in the legend.

Format

An object of class `GeomRoc` (inherits from `Geom`, `ggproto`, `gg`) of length 6.

Computed variables

false_positive_fraction estimate of false positive fraction

true_positive_fraction estimate of true positive fraction

cutoffs values of `m` at which estimates are calculated

Aesthetics

`geom_roc` understands the following aesthetics (required aesthetics are in bold):

- `x` The FPF estimate. This is automatically mapped by `stat_roc`
- `y` The TPF estimate. This is automatically mapped by `stat_roc` smallest level in sort order is assumed to be 0, with a warning
- **alpha**
- **color**
- **fill**
- **linetype**
- **size**

See Also

See [geom_rocci](#) for displaying rectangular confidence regions for the empirical ROC curve, [style_roc](#) for adding guidelines and labels, and [direct_label](#) for adding direct labels to the curves. Also [export_interactive_roc](#) for creating interactive ROC curve plots for use in a web browser.

Examples

```
D.ex <- rbinom(50, 1, .5)
rocdata <- data.frame(D = c(D.ex, D.ex),
                      M = c(rnorm(50, mean = D.ex, sd = .4), rnorm(50, mean = D.ex, sd = 1)),
                      Z = c(rep("A", 50), rep("B", 50)))

ggplot(rocdata, aes(m = M, d = D)) + geom_roc()

ggplot(rocdata, aes(m = M, d = D, color = Z)) + geom_roc()
ggplot(rocdata, aes(m = M, d = D)) + geom_roc() + facet_wrap(~ Z)
ggplot(rocdata, aes(m = M, d = D)) + geom_roc(n.cuts = 20)
ggplot(rocdata, aes(m = M, d = D)) + geom_roc(cutoffs.at = c(1.5, 1, .5, 0, -.5))
ggplot(rocdata, aes(m = M, d = D)) + geom_roc(labels = FALSE)
ggplot(rocdata, aes(m = M, d = D)) + geom_roc(size = 1.25)
```

geom_rocci

Confidence regions for the ROC curve

Description

Display rectangular confidence regions for the empirical ROC curve.

Usage

```
geom_rocci(
  mapping = NULL,
  data = NULL,
  stat = "rocci",
  ci.at = NULL,
  sig.level = 0.05,
  na.rm = TRUE,
  alpha.box = 0.3,
  labels = TRUE,
  labelsize = 3.88,
  labelround = 1,
  position = "identity",
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)

GeomRocci
```

Arguments

mapping	Set of aesthetic mappings created by aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to ggplot(). A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
stat	Use to override the default connection between <code>geom_rocci</code> and <code>stat_rocci</code> .
ci.at	Vector of values in the range of the biomarker where confidence regions will be displayed
sig.level	Significance level for the confidence regions
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
alpha.box	Alpha level for the confidence regions
labels	If <code>TRUE</code> , adds text labels for the cutoffs where the confidence regions are displayed
labelsize	Size of cutoff text labels
labelround	Integer, number of significant digits to round cutoff labels
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following: • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the <code>position_</code> prefix. For example, to use <code>position_jitter()</code>, give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use <code>TRUE</code> . If <code>NA</code> , all levels are shown in legend, but unobserved levels are omitted.
inherit.aes	If <code>FALSE</code> , overrides the default aesthetics, rather than combining with them.
...	Other arguments passed on to layer() 's <code>params</code> argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the <code>position</code> argument, or aesthetics that are required can <i>not</i> be passed through Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the params. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the geom part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the stat part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `...`. This can be one of the functions described as [key glyphs](#), to change the display of the layer in the legend.

Format

An object of class `GeomRocci` (inherits from `Geom`, `ggproto`, `gg`) of length 6.

Aesthetics

`geom_rocci` understands the following aesthetics (required aesthetics are in bold). `stat_rocci` automatically maps the estimates to the required aesthetics:

- `x` The FPF estimate
- `y` The TPF estimate
- `xmin` Lower confidence limit for the FPF
- `xmax` Upper confidence limit for the FPF
- `ymin` Lower confidence limit for the TPF
- `ymax` Upper confidence limit for the TPF
- `alpha`
- `color`
- `fill`
- `linetype`
- `size`

See Also

See [geom_roc](#) for the empirical ROC curve, [style_roc](#) for adding guidelines and labels, and [direct_label](#) for adding direct labels to the curves. Also [export_interactive_roc](#) for creating interactive ROC curve plots for use in a web browser.

Examples

```
D.ex <- rbinom(50, 1, .5)
rocddata <- data.frame(D = c(D.ex, D.ex),
  M = c(rnorm(50, mean = D.ex, sd = .4), rnorm(50, mean = D.ex, sd = 1)),
  Z = c(rep("A", 50), rep("B", 50)))

ggplot(rocddata, aes(m = M, d = D)) + geom_roc() + geom_rocci()
ggplot(rocddata, aes(m = M, d = D, color = Z)) + geom_roc() + geom_rocci()
ggplot(rocddata, aes(m = M, d = D, color = Z)) + geom_roc() + geom_rocci(sig.level = .01)
ggplot(rocddata, aes(m = M, d = D)) + geom_roc(n.cuts = 0) +
  geom_rocci(ci.at = quantile(rocddata$M, c(.1, .25, .5, .75, .9)))
ggplot(rocddata, aes(m = M, d = D, color = Z)) + geom_roc() + geom_rocci(linetype = 1)
```

getD3	<i>Reads included JavaScript functions and returns them as a string for pasting into a webpage</i>
-------	--

Description

Reads included JavaScript functions and returns them as a string for pasting into a webpage

Usage

```
getD3()
```

ggroc	<i>Plot an ROC curve</i>
-------	--------------------------

Description

Deprecated, use [geom_roc](#) instead

Usage

```
ggroc(
  rocddata,
  fpf_string = "FPF",
  tpf_string = "TPF",
  c_string = "c",
  ci = FALSE,
  label = NULL,
  label.adj.x = 0,
  label.adj.y = 0,
  label.angle = 45,
  plotmath = FALSE,
  xlabel = "False positive fraction",
  ylabel = "True positive fraction"
)
```

Arguments

rocdata	Data frame containing true and false positive fractions, and cutoff values
fpf_string	Column name identifying false positive fraction column
tpf_string	Column name identifying true positive fraction column
c_string	Column name identifying cutoff values
ci	Logical, not supported
label	Not supported
label.adj.x	Not supported
label.adj.y	Not supported
label.angle	Not supported
plotmath	Not supported
xlabel	Defaults to "False positive fraction"
ylabel	Defaults to "True positive fraction"

Value

A ggplot object

melt_roc	<i>Transform biomarkers stored as wide to long</i>
----------	--

Description

Multiple biomarkers measured on the same subjects are often stored as multiple columns in a data frame. This is a convenience function that transforms the data into long format, suitable for use with ggplot and [geom_roc](#)

Usage

```
melt_roc(data, d, m, names = NULL)
```

Arguments

data	Data frame containing disease status and biomarkers stored in columns
d	Column containing binary disease status. Can be a column name or index
m	Vector of column names or indices identifying biomarkers
names	Optional vector of names to assign to the biomarkers. If NULL, names will be taken from the column names

Value

A data frame in long format with three columns: D = binary disease status, M = biomarker value, and name = biomarker name

Examples

```
D.ex <- rbinom(50, 1, .5)
widedata <- data.frame(D = D.ex, M1 = rnorm(50, mean = D.ex, sd = 1),
  M2 = rnorm(50, mean = D.ex, sd = .5))
longdata <- melt_roc(widedata, "D", c("M1", "M2"))
ggplot(longdata, aes(d = D, m = M, color = name)) + geom_roc()
```

multi_ggroc

Plot multiple ROC curves

Description

Given a list of results computed by [calculate_roc](#), plot the curve using ggplot with sensible defaults. Pass the resulting object and data to [export_interactive_roc](#), [plot_interactive_roc](#), or [plot_journal_roc](#).

Usage

```
multi_ggroc(
  datalist,
  fpf_string = rep("FPF", length(datalist)),
  tpf_string = rep("TPF", length(datalist)),
  c_string = rep("c", length(datalist)),
  label = NULL,
  legend = TRUE,
  label.adj.x = rep(0, length(datalist)),
  label.adj.y = rep(0, length(datalist)),
  label.angle = rep(45, length(datalist)),
  plotmath = FALSE,
  xlabel = "False positive fraction",
  ylabel = "True positive fraction"
)
```

Arguments

<code>datalist</code>	List of data frames each containing true and false positive fractions and cutoffs
<code>fpf_string</code>	Column names identifying false positive fraction
<code>tpf_string</code>	Column names identifying true positive fraction
<code>c_string</code>	Column names identifying cutoff values
<code>label</code>	Not supported.
<code>legend</code>	If true, draws legend
<code>label.adj.x</code>	Not supported.
<code>label.adj.y</code>	Not supported.
<code>label.angle</code>	Not supported.
<code>plotmath</code>	Logical. Not supported.
<code>xlabel</code>	Defaults to "False positive fraction"
<code>ylabel</code>	Defaults to "True positive fraction"

Value

A ggplot object

plotROC

Tools for plotting ROC Curves

Description

Generate Useful ROC Curve Charts for Print and Interactive Use. This defines a set of stats and geoms for use with ggplot2. In addition, ggplot objects created with these geoms can be exported and turned into interactive plots for use on the web. The interactive features include hovering to display hidden labels, and clicking to reveal confidence regions.

Details

To get started, see [geom_roc](#), [geom_rocci](#), or the examples below. For transforming data, we also provide the convenience function [melt_roc](#).

The vignette contains some examples, along with explanations of the results. To view, run `vignette("examples", package = "plotROC")`

Author(s)

Michael Sachs (@sachsmc)

See Also

Useful links:

- <https://sachsmc.github.io/plotROC/>
- Report bugs at <https://github.com/sachsmc/plotROC/issues/>

Examples

```
D.ex <- rbinom(50, 1, .5)
rocddata <- data.frame(D = c(D.ex, D.ex),
                      M = c(rnorm(50, mean = D.ex, sd = .4), rnorm(50, mean = D.ex, sd = 1)),
                      Z = c(rep("A", 50), rep("B", 50)))

ggplot(rocddata, aes(m = M, d = D)) + geom_roc() + geom_rocci()

ggplot(rocddata, aes(m = M, d = D)) + geom_roc() + style_roc()
ggplot(rocddata, aes(m = M, d = D, color = Z)) + geom_roc() + geom_rocci()
ggplot(rocddata, aes(m = M, d = D, color = Z)) + geom_roc() + geom_rocci(sig.level = .01)
ggplot(rocddata, aes(m = M, d = D)) + geom_roc(n.cuts = 0) +
  geom_rocci(ci.at = quantile(rocddata$M, c(.1, .25, .5, .75, .9)))
ggplot(rocddata, aes(m = M, d = D, color = Z)) + geom_roc() + geom_rocci(linetype = 1)

rocplot <- ggplot(rocddata, aes(m = M, d = D)) + geom_roc()
```

```
plot_interactive_roc(rocplot)
plot_interactive_roc(rocplot + aes(color = Z))
plot_interactive_roc(rocplot + facet_wrap(~ Z))
```

plot_interactive_roc	<i>Generate a standalone html document displaying an interactive ROC curve</i>
----------------------	--

Description

Generate a standalone html document displaying an interactive ROC curve

Usage

```
plot_interactive_roc(ggroc, file = NULL, ...)
```

Arguments

ggroc	An object as returned by ggroc or multi_ggroc . It can be modified with annotations, themes, etc.
file	A path to save the result to. If NULL, will save to a temporary directory
...	arguments passed to export_interactive_roc

Value

NULL opens an interactive document in Rstudio or the default web browser

plot_journal_roc	<i>Plot an ROC curve for use in print</i>
------------------	---

Description

Deprecated, use [style_roc](#) instead

Usage

```
plot_journal_roc(  
  ggroc_p,  
  font.size = 3,  
  n.cuts = 20,  
  ci.at = NULL,  
  opacity = 0.3,  
  lty = NULL,  
  color = NULL,  
  lwd = NULL,  
  legend = FALSE  
)
```

Arguments

ggroc_p	An object as returned by ggroc or multi_ggroc . It can be modified with annotations, themes, etc.
font.size	Not supported
n.cuts	Not supported
ci.at	Not supported
opacity	Not supported
lty	Not supported
color	Not supported
lwd	Not supported
legend	Not supported

Value

A ggplot object

roc_key	<i>Key for ROC geom</i>
---------	-------------------------

Description

Key for ROC geom

Usage

```
roc_key(data, params, size)
```

Arguments

data	Data created by stat
params	parameters
size	Size

shiny_plotROC

Start the plotROC Shiny app

Description

A convenience function to easily start the shiny application. It will open in Rstudio, or in the default web browser.

Usage

```
shiny_plotROC()
```

StatRoc

Calculate the empirical Receiver Operating Characteristic curve

Description

Given a binary outcome `d` and continuous measurement `m`, computes the empirical ROC curve for assessing the classification accuracy of `m`

Usage

```
StatRoc
```

```
stat_roc(
  mapping = NULL,
  data = NULL,
  geom = "roc",
  position = "identity",
  show.legend = NA,
  inherit.aes = TRUE,
  na.rm = TRUE,
  max.num.points = 1000,
  increasing = TRUE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by <code>aes()</code> . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
---------	--

data	The data to be displayed in this layer. There are three options: If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code>. A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
geom	The geometric object to use to display the data for this layer. When using a <code>stat_*()</code> function to construct a layer, the <code>geom</code> argument can be used to override the default coupling between stats and geoms. The <code>geom</code> argument accepts the following: • A <code>Geom</code> ggproto subclass, for example <code>GeomPoint</code>. • A string naming the geom. To give the geom as a string, strip the function name of the <code>geom_</code> prefix. For example, to use <code>geom_point()</code>, give the geom as "point". • For more information and other ways to specify the geom, see the layer geom documentation.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following: • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the <code>position_</code> prefix. For example, to use <code>position_jitter()</code>, give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
show.legend	logical. Should this layer be included in the legends? <code>NA</code>, the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use <code>TRUE</code>. If <code>NA</code>, all levels are shown in legend, but unobserved levels are omitted.
inherit.aes	If <code>FALSE</code>, overrides the default aesthetics, rather than combining with them.
na.rm	Remove missing observations
max.num.points	maximum number of points to plot
increasing	<code>TRUE</code> (default) if <code>M</code> is positively associated with $\Pr(D = 1)$, if <code>FALSE</code>, assumes <code>M</code> is negatively associated with $\Pr(D = 1)$
...	Other arguments passed on to <code>layer()</code>'s <code>params</code> argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the <code>position</code> argument, or aesthetics that are required can <i>not</i> be passed through Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the params. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the geom part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the stat part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `...`. This can be one of the functions described as [key glyphs](#), to change the display of the layer in the legend.

Format

An object of class `StatRoc` (inherits from `Stat`, `ggproto`, `gg`) of length 6.

Aesthetics

`stat_roc` understands the following aesthetics (required aesthetics are in bold):

- **m** The continuous biomarker/predictor
- **d** The binary outcome, if not coded as 0/1, the smallest level in sort order is assumed to be 0, with a warning
- **alpha** Controls the label alpha, see also `linealpha` and `pointalpha`
- **color**
- **linetype**
- **size** Controls the line weight, see also `pointsize` and `labelsize`

Computed variables

false_positive_fraction estimate of false positive fraction

true_positive_fraction estimate of true positive fraction

cutoffs values of `m` at which estimates are calculated

Examples

```
D.ex <- rbinom(50, 1, .5)
rocdata <- data.frame(D = c(D.ex, D.ex),
  M = c(rnorm(50, mean = D.ex, sd = .4), rnorm(50, mean = D.ex, sd = 1)),
  Z = c(rep("A", 50), rep("B", 50)))

ggplot(rocdata, aes(m = M, d = D)) + stat_roc()
```

StatRocci

*Calculate confidence regions for the empirical ROC curve***Description**

Confidence intervals for TPF and FPF are calculated using the exact method of Clopper and Pearson (1934) each at the level $1 - \sqrt{1 - \alpha}$. Based on result 2.4 from Pepe (2003), the cross-product of these intervals yields a $1 - \alpha$

Usage

```
StatRocci

stat_rocci(
  mapping = NULL,
  data = NULL,
  geom = "rocci",
  position = "identity",
  show.legend = NA,
  inherit.aes = TRUE,
  ci.at = NULL,
  sig.level = 0.05,
  na.rm = TRUE,
  ...
)
```

Arguments

- | | |
|---------|---|
| mapping | Set of aesthetic mappings created by aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. |
| data | <p>The data to be displayed in this layer. There are three options:</p> <p>If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to ggplot().</p> <p>A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created.</p> <p>A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).</p> |
| geom | <p>The geometric object to use to display the data for this layer. When using a <code>stat_*()</code> function to construct a layer, the <code>geom</code> argument can be used to override the default coupling between stats and geoms. The <code>geom</code> argument accepts the following:</p> <ul style="list-style-type: none"> • A <code>Geom</code> ggproto subclass, for example <code>GeomPoint</code>. |

	• A string naming the geom. To give the geom as a string, strip the function name of the <code>geom_</code> prefix. For example, to use <code>geom_point()</code>, give the geom as "point". • For more information and other ways to specify the geom, see the layer geom documentation.
<code>position</code>	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The <code>position</code> argument accepts the following: • The result of calling a position function, such as <code>position_jitter()</code>. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the <code>position_</code> prefix. For example, to use <code>position_jitter()</code>, give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
<code>show.legend</code>	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
<code>inherit.aes</code>	If FALSE, overrides the default aesthetics, rather than combining with them.
<code>ci.at</code>	Vector of cutoffs at which to display confidence regions. If NULL, will automatically choose 3 evenly spaced points to display the regions
<code>sig.level</code>	Significance level for the confidence regions
<code>na.rm</code>	Remove missing observations
<code>...</code>	Other arguments passed on to <code>layer()</code>'s <code>params</code> argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the <code>position</code> argument, or aesthetics that are required can <i>not</i> be passed through ... Unknown arguments that are not part of the 4 categories below are ignored. • Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, <code>colour = "red"</code> or <code>linewidth = 3</code>. The geom's documentation has an Aesthetics section that lists the available options. The 'required' aesthetics cannot be passed on to the <code>params</code>. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data. • When constructing a layer using a <code>stat_*()</code> function, the <code>...</code> argument can be used to pass on parameters to the geom part of the layer. An example of this is <code>stat_density(geom = "area", outline.type = "both")</code>. The geom's documentation lists which parameters it can accept. • Inversely, when constructing a layer using a <code>geom_*()</code> function, the <code>...</code> argument can be used to pass on parameters to the stat part of the layer. An example of this is <code>geom_area(stat = "density", adjust = 0.5)</code>. The stat's documentation lists which parameters it can accept.

- The `key_glyph` argument of `layer()` may also be passed on through `...`. This can be one of the functions described as `key glyphs`, to change the display of the layer in the legend.

Format

An object of class `StatRocci` (inherits from `Stat`, `ggproto`, `gg`) of length 6.

Aesthetics

`stat_rocci` understands the following aesthetics (required aesthetics are in bold):

- **m** The continuous biomarker/predictor
- **d** The binary outcome, if not coded as 0/1, the smallest level in sort order is assumed to be 0, with a warning
- `alpha`
- `color`
- `fill`
- `linetype`
- `size`

Computed variables

FPF estimate of false positive fraction

TPF estimate of true positive fraction

cutoffs values of `m` at which estimates are calculated

FPFL lower bound of confidence region for FPF

FPFU upper bound of confidence region for FPF

TPFL lower bound of confidence region for TPF

TPFU upper bound of confidence region for TPF

References

- Clopper, C. J., and Egon S. Pearson. "The use of confidence or fiducial limits illustrated in the case of the binomial." *Biometrika* (1934): 404-413.
- Pepe, M.S. "The Statistical Evaluation of Medical Tests for Classification and Prediction." Oxford (2003).

Examples

```
D.ex <- rbinom(50, 1, .5)
rocdata <- data.frame(D = c(D.ex, D.ex),
  M = c(rnorm(50, mean = D.ex, sd = .4), rnorm(50, mean = D.ex, sd = 1)),
  Z = c(rep("A", 50), rep("B", 50)))

ggplot(rocdata, aes(m = M, d = D)) + geom_roc() + stat_rocci()
```

```
ggplot(rocdata, aes(m = M, d = D)) + geom_roc() +
  stat_rocci(ci.at = quantile(rocdata$M, c(.1, .3, .5, .7, .9)))
```

style_roc

Add guides and annotations to a ROC plot

Description

Adds a diagonal guideline, minor grid lines, and optionally direct labels to ggplot objects containing a `geom_roc` layer.

Usage

```
style_roc(
  major.breaks = c(0, 0.1, 0.25, 0.5, 0.75, 0.9, 1),
  minor.breaks = c(seq(0, 0.1, by = 0.01), seq(0.9, 1, by = 0.01)),
  guide = TRUE,
  guide_col = NULL,
  xlab = "False positive fraction",
  ylab = "True positive fraction",
  theme = theme_bw
)
```

Arguments

<code>major.breaks</code>	vector of breakpoints for the major gridlines and axes
<code>minor.breaks</code>	vector of breakpoints for the minor gridlines and axes
<code>guide</code>	logical, if TRUE draws diagonal guideline
<code>guide_col</code>	character, if guide is TRUE sets colour of diagonal guideline. Takes precedence over colour set by a theme.
<code>xlab</code>	X-axis label
<code>ylab</code>	Y-axis label
<code>theme</code>	Theme function compatible with ggplot2

Examples

```
D.ex <- rbinom(50, 1, .5)
fakedata <- data.frame(M1 = rnorm(50, mean = D.ex),
  D = D.ex)
ggplot(fakedata, aes(m = M1, d = D)) + geom_roc() + style_roc()
ggplot(fakedata, aes(m = M1, d = D)) + geom_roc() + style_roc(xlab = "1 - Specificity")
ggplot(fakedata, aes(m = M1, d = D)) + geom_roc() + style_roc(theme = theme_grey)
```

`verify_d`*Check that D is suitable for using as binary disease status*

Description

Checks for two classes and gives a warning message indicating which level is assumed to be 0/1. Throws an error if more than two levels appear in D.

Usage

```
verify_d(D)
```

Arguments

D Vector that will be checked for 2-class labels

Value

A vector the same length as D that takes values 0, indicating no disease or 1 indicating disease.

Examples

```
verify_d(c(1, 0, 1))  
## Not run:  
verify_d(c(TRUE, FALSE, TRUE)) #warning  
verify_d(c("Dead", "Alive", "Dead")) #warning  
verify_d(c("Disease", "Healthy", "Missing")) #error  
  
## End(Not run)
```

Index

- * **Internal**
 - getD3, [12](#)
 - roc_key, [17](#)
- * **datasets**
 - geom_rocci, [9](#)
 - GeomRoc, [6](#)
 - StatRoc, [18](#)
 - StatRocci, [21](#)
- aes(), [6](#), [10](#), [18](#), [21](#)
- annotate, [4](#)
- arrow, [7](#)
- calc_auc, [3](#)
- calculate_multi_roc, [2](#)
- calculate_roc, [3](#), [14](#)
- direct_label, [4](#), [9](#), [11](#)
- export_interactive_roc, [5](#), [9](#), [11](#), [14](#), [16](#)
- fortify(), [7](#), [10](#), [19](#), [21](#)
- geom_roc, [2–5](#), [11–13](#), [15](#)
- geom_roc (GeomRoc), [6](#)
- geom_rocci, [5](#), [9](#), [9](#), [15](#)
- GeomRoc, [6](#)
- GeomRocci (geom_rocci), [9](#)
- getD3, [12](#)
- ggplot(), [7](#), [10](#), [19](#), [21](#)
- ggroc, [12](#), [16](#), [17](#)
- key glyphs, [8](#), [11](#), [20](#), [23](#)
- layer geom, [19](#), [22](#)
- layer position, [7](#), [10](#), [19](#), [22](#)
- layer(), [8](#), [10](#), [11](#), [19](#), [20](#), [22](#), [23](#)
- melt_roc, [13](#), [15](#)
- multi_ggroc, [14](#), [16](#), [17](#)
- plot_interactive_roc, [14](#), [16](#)
- plot_journal_roc, [14](#), [16](#)
- plotROC, [15](#)
- plotROC-package (plotROC), [15](#)
- roc_key, [17](#)
- shiny_plotROC, [18](#)
- stat_roc, [8](#)
- stat_roc (StatRoc), [18](#)
- stat_rocci (StatRocci), [21](#)
- StatRoc, [18](#)
- StatRocci, [21](#)
- style_roc, [5](#), [9](#), [11](#), [16](#), [24](#)
- verify_d, [25](#)