

# Package ‘SPARRAfairness’

November 7, 2024

**Title** Analysis of Differential Behaviour of SPARRA Score Across Demographic Groups

**Version** 0.0.0.2

**Maintainer** James Liley <james.liley@durham.ac.uk>

**Description** The SPARRA risk score (Scottish Patients At Risk of admission and Re-Admission) estimates yearly risk of emergency hospital admission using electronic health records on a monthly basis for most of the Scottish population. This package implements a suite of functions used to analyse the behaviour and performance of the score, focusing particularly on differential performance over demographically-defined groups. It includes useful utility functions to plot receiver-operator-characteristic, precision-recall and calibration curves, draw stock human figures, estimate counterfactual quantities without the need to re-compute risk scores, to simulate a semi-realistic dataset.

**License** GPL (>= 3)

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.5.0), stats, graphics, grDevices, matrixStats, ranger

**Imports** mvtnorm, cvAUC, ggplot2, ggrepel, patchwork, scales

**RoxygenNote** 7.3.1

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Ioanna Thoma [aut] (<<https://orcid.org/0000-0001-6928-2198>>),  
Catalina Vallejos [ctb] (<<https://orcid.org/0000-0003-3638-1960>>),  
Louis Aslett [ctb] (<<https://orcid.org/0000-0003-2211-233X>>),  
Jill Ireland [ctb] (<<https://orcid.org/0009-0009-5324-6630>>),  
Simon Rogers [ctb] (<<https://orcid.org/0000-0003-3578-4477>>),  
James Liley [cre, aut] (<<https://orcid.org/0000-0002-0049-8238>>)

**Repository** CRAN

**Date/Publication** 2024-11-07 10:30:02 UTC

## Contents

ab	2
adjusted_fdr	3
adjusted_for	4
all_data	5
build_diff	5
cal_2panel	6
counterfactual_yhat	7
dat2mat	10
decomposition_matrix	11
demographic_parity	12
drawperson	12
drawprop	14
for_breakdown	15
getcal	16
getprc	17
getroc	18
groupmetric_2panel	19
group_fairness	20
integral	21
logistic	21
logit	22
phs_colours	22
plot.sparraCAL	23
plot.sparraPRC	24
plot.sparraROC	24
plot_decomp	25
prc_2panel	26
roc_2panel	27
sim_pop_data	28
<b>Index</b>	<b>30</b>

---

ab	<i>ab() Shorthand to draw a red x-y line</i>
----	--

---

### Description

ab() Shorthand to draw a red x-y line

### Usage

ab(...)

### Arguments

...            passed to abline()

**Value**

No return value, draws a figure

---

adjusted_fdr	<i>adjusted_fdr</i>
--------------	---------------------

---

**Description**

Estimates false discovery rate  $P(\text{target}=\text{FALSE}|\text{score}>\text{cutoff}, \text{group}=\text{g})$  'adjusted' for some category.

**Usage**

```
adjusted_fdr(
  scores,
  target,
  category,
  group1,
  group2,
  cutoffs = seq(min(scores, na.rm = TRUE), max(scores, na.rm = TRUE), length = 100),
  nboot = 100
)
```

**Arguments**

scores	vector of risk scores
target	vector of values of target (which risk score aims to predict)
category	vector of categories
group1	indices of group 1
group2	indices of group 2
cutoffs	score cutoffs at which to estimate metric (default 100 evenly-spaced)
nboot	number of bootstrap samples for standard error

**Details**

Namely, calculates

$$\sum ( P(\text{target}=\text{FALSE}|\text{score}>\text{cutoff}, \text{category}=\text{c}, \text{group}=\text{g}) P(\text{category}=\text{c}|\text{score}<\text{cutoff}) )$$

where the sum is over categories c.

**Value**

matrix of dimension  $\text{length}(\text{cutoffs}) \times 4$ , with  $(i, 2g-1)$ th entry the relevant fairness metric for group g at the  $i$ th cutoff value and  $(i, 2g)$ th entry the approximate standard error of the  $(i, 2g-1)$ th entry

**Examples**

```
# See vignette
```

---

adjusted_for	<i>adjusted_for</i>
--------------	---------------------

---

### Description

Estimates false omission rate  $P(\text{target}=\text{TRUE}|\text{score}\leq\text{cutoff},\text{group}=\text{g})$  'adjusted' for some category.

### Usage

```
adjusted_for(
  scores,
  target,
  category,
  group1,
  group2,
  cutoffs = seq(min(scores, na.rm = TRUE), max(scores, na.rm = TRUE), length = 100),
  nboot = 100
)
```

### Arguments

scores	vector of risk scores
target	vector of values of target (which risk score aims to predict)
category	vector of categories
group1	indices of group 1
group2	indices of group 2
cutoffs	score cutoffs at which to estimate metric (default 100 evenly-spaced)
nboot	number of bootstrap samples for standard error

### Details

Namely, calculates

$$\sum ( P(\text{target}=\text{TRUE}|\text{score}\leq\text{cutoff},\text{category}=\text{c},\text{group}=\text{g})P(\text{category}=\text{c}|\text{score}<\text{cutoff}) )$$

where the sum is over categories c.

### Value

matrix of dimension  $\text{length}(\text{cutoffs})\times 4$ , with  $(i,2g-1)$ th entry the relevant fairness metric for group g at the  $i$ th cutoff value and  $(i,2g)$ th entry the approximate standard error of the  $(i,2g-1)$ th entry

### Examples

```
# See vignette
```

---

all_data	<i>All data for fairness measures</i>
----------	---------------------------------------

---

**Description**

This object contains all data from analysis of fairness measures in SPARRA v3 and v4.

**Usage**

```
all_data
```

**Format**

An object of class `list` of length 1261.

---

build_diff	<i>build_diff Prepares a data frame for a ggplot object to compare differences using linear interpolation.</i>
------------	--

---

**Description**

`build_diff` Prepares a data frame for a `ggplot` object to compare differences using linear interpolation.

**Usage**

```
build_diff(df, xvar)
```

**Arguments**

df	data frame
xvar	name of variable to consider as 'x': interpolate over evenly spaced values of this variable.

**Value**

data frame using (common) interpolated x values rather than arbitrary x values

**Examples**

```
# Only used internally
```

---

cal_2panel	<i>cal_2panel</i> Draws calibration curves (with legend) with a second panel underneath showing predicted differences.
------------	--

---

### Description

cal\_2panel Draws calibration curves (with legend) with a second panel underneath showing predicted differences.

### Usage

```
cal_2panel(  
  cals,  
  labels,  
  col = 1:length(cals),  
  xy_col = phs_colours("phs-magenta"),  
  ci_col = col,  
  highlight = NULL,  
  yrange_lower = NULL,  
  legend_title = ""  
)
```

### Arguments

cals	list of calibration objects, output from getcal().
labels	labels to use in legend
col	line colours
xy_col	line colour for x-y line, defaults to phs-magenta
ci_col	colours to draw confidence intervals on lower panel; NA to not draw.
highlight	if non-null, highlight a particular value
yrange_lower	y range for lower plot. If NULL, generates automatically
legend_title	title for legend, defaults to nothing

### Value

Silently return ggplot object

### Examples

```
# See vignette
```

---

counterfactual\_yhat     *counterfactual\_yhat*

---

### Description

Estimation of counterfactual quantities by resampling.

### Usage

```
counterfactual_yhat(dat, X, x = NULL, G, g, gdash, excl = NULL, n = NULL)
```

### Arguments

dat	data frame containing variables in X, Yhat, G, excl. Variables in U are assumed to be $\text{colnames}(\text{dat}) \setminus (X \cup \text{Yhat} \cup G \cup \text{excl})$
X	set of variables and values on which to 'condition'; essentially we assume that any causal pathway from G to Yhat is through X
x	values of variables X on which to condition; can be null in which case we use marginal distribution of X
G	grouping variable, usually a sensitive attribute
g	conditioned value of g
gdash	counterfactual value of g
excl	variable names to exclude from U
n	number of samples; if NULL return all

### Details

Counterfactual fairness is with respect to the causal graph:

$G \leftarrow \text{U} \quad \text{U} \rightarrow \text{V} \rightarrow \text{X} \rightarrow \text{Yhat}$

where

- G=group (usually sensitive attribute);
- Yhat=outcome;
- X=set of variables through which G can act on Yhat,
- U=set of background variables;

We want the counterfactual  $Yhat_{g' \leftarrow G} | X = x, G = g$  (or alternatively  $(Yhat_{g' \leftarrow G} | G = g)$ ), using the term *Yhat* as it appears in the function, rather than  $\hat{Y}$ .

This can be interpreted as: the distribution of values of Yhat amongst individuals whose values of U are distributed as if G were  $g'$ .

Essentially, comparison of the counterfactual quantity above to the conditional  $(Yhat | G = g)$  isolates the difference in Yhat due to the effect of G on Yhat through X, removing any effect due to different distributions of U due to different values of G.

To estimate  $Y' = Yhat_{g' \rightarrow G} | G = g$ , we need to

1. Compute  $U' \sim (U|G = g)$
2. Compute the distribution  $X'$  as  $X' \sim (X|U \sim U', G = g')$
3. Sample  $Y'$  ( $Yhat|X \sim X', U \sim U'$ )

To estimate  $Y' = Yhat_{g' \rightarrow G}|X = x, G = g$ , we need to

1. Compute  $U' \sim (U|G = g, X = x)$
2. Compute the distribution  $X'$  as  $X' \sim (X|U \sim U', G = g')$
3. Sample  $Y' \sim (Yhat|X \sim X', U \sim U')$

This function approximates this sampling procedure as follows

1. Look at individuals with  $G = g$  (and optionally  $X = x$ )
2. Find the values of  $U$  for these individuals
3. Find a second set of individuals with the same values of  $U$  but for whom  $G = g'$
4. Return the indices of these individuals

The values of  $Yhat$  for these individuals constitute a sample from the desired counterfactual.

## Value

indices representing sample(s) from counterfactual  $Yhat(g' <- G) | X=x, G=g$

## Examples

```
set.seed(23173)
N=10000

# Background variables sampler
background_U=function(n) runif(n) # U~U(0,1)

# Structural equations
struct_G=function(u,n) rbinom(n,1,prob=u) # G|U=u ~ Bern(u)
struct_X=function(u,g,n) rbinom(n,1,prob=u*(0.5 + 0.5*g)) # X|U=u,G=g ~ Bern(u(1+g)/2)
struct_Yhat=function(u,x,n) (runif(n,0,x) + runif(n,0,u))/2 # Yhat|X,N ~ (U(0,X) + U(0,U))/2

# To see that the counterfactual 'isolates' the difference in Yhat due to the
# causal pathway from G to Yhat through X, change the definition of struct_G to
#
# struct_G=function(u,n) rbinom(n,1,prob=1/2) # G|U=u ~ Bern(1/2)
#
# so the posterior of U|G=g does not depend on g. Note that, with this definition, the
# counterfactual Yhat{G<-1}|G=1 coincides with the conditional Yhat|G=0, since
# the counterfactual G<-1 is equivalent to just conditioning on G=1.
#
# By contrast, if we change struct_G back to its original definition, but
# change the definition of struct_Yhat to
#
# struct_Yhat=function(u,x,n) (runif(n,0,1) + runif(n,0,u))/2 # Yhat|X,N ~ (U(0,1) + U(0,U))/2
#
# so Yhat depends on G only through the change in posterior of U from changing g,
```



```

# the counterfactual  $\hat{Y}_{G=1}$  coincides with the conditional  $\hat{Y}_{G=1}$ .

# Sample from complete causal model
U=background_U(N)
G=struct_G(U,N)
X=struct_X(U,G,N)
Yhat=struct_Yhat(U,X,N)
dat=data.frame(U,G,X,Yhat)

# True counterfactual  $\hat{Y}_{G \leftarrow 0}|G=1$ 
w1=which(dat$G==1)
n1=length(w1)
UG1=dat$U[w1] # This is  $U|G=1$ 
XG1=struct_X(UG1,rep(0,n1),n1)
YhatG1=struct_Yhat(UG1,XG1,n1)

# Estimated counterfactual  $\hat{Y}_{G \leftarrow 0}|G=1$ 
ind_G1=counterfactual_yhat(dat,X="X",G="G",g = 1, gdash = 0)
YhatG1_resample=dat$Yhat[ind_G1]

# True counterfactual  $\hat{Y}_{G \leftarrow 0}|G=1, X=1$ 
w11=which(dat$G==1 & dat$X==1)
n11=length(w11)
UG1X1=dat$U[w11] # This is  $U|G=1, X=1$ 
XG1X1=struct_X(UG1X1,rep(0,n11),n11)
YhatG1X1=struct_Yhat(UG1X1,XG1X1,n11)

# Estimated counterfactual  $\hat{Y}_{G \leftarrow 0}|G=1$ 
ind_G1X1=counterfactual_yhat(dat,X="X",G="G",g = 1, gdash = 0,x=1)
YhatG1X1_resample=dat$Yhat[ind_G1X1]

# Compare CDFs
x=seq(0,1,length=1000)
oldpar = par(mfrow=c(1,2))

plot(0,type="n",xlim=c(0,1),ylim=c(0,1),xlab="Value",
     ylab=expression(paste("Prop. ",hat('Y')," < x")))
lines(x,ecdf(dat$Yhat)(x),col="black") # Unconditional CDF of  $\hat{Y}$ 
lines(x,ecdf(dat$Yhat[which(dat$G==1)])(x),col="red") #  $\hat{Y}_{G=1}$ 
lines(x,ecdf(dat$Yhat[which(dat$G==0)])(x),col="blue") #  $\hat{Y}_{G=0}$ 

# True counterfactual  $\hat{Y}_{G \leftarrow 0}|G=1$ 
lines(x,ecdf(YhatG1)(x),col="blue",lty=2)

# Estimated counterfactual  $\hat{Y}_{G \leftarrow 0}|G=1$ 
lines(x,ecdf(YhatG1_resample)(x),col="blue",lty=3)

```

```

legend("bottomright",
      c(expression(paste(hat('Y'))),
        expression(paste(hat('Y'), "|G=1")),
        expression(paste(hat('Y'), "|G=0")),
        expression(paste(hat(Y)[G %<-% 0], "|G=1 (true)")),
        expression(paste(hat(Y)[G %<-% 0], "|G=1 (est.)"))),
      col=c("black", "red", "blue", "blue", "blue"),
      lty=c(1, 1, 1, 2, 3),
      cex=0.5)

plot(0, type="n", xlim=c(0, 1), ylim=c(0, 1), xlab="Value",
     ylab=expression(paste("Prop. ", hat('Y'), " < x")))
lines(x, ecdf(dat$Yhat[which(dat$X==1)])(x), col="black") # CDF of Yhat|X=1
lines(x, ecdf(dat$Yhat[which(dat$G==1 & dat$X==1)])(x), col="red") # Yhat|G=1, X=1
lines(x, ecdf(dat$Yhat[which(dat$G==0 & dat$X==1)])(x), col="blue") # Yhat|G=0, X=1

# True counterfactual Yhat{G <- 0}|G=1, X=1
lines(x, ecdf(YhatG1X1)(x), col="blue", lty=2)

# Estimated counterfactual Yhat{G <- 0}|G=1, X=1
lines(x, ecdf(YhatG1X1_resample)(x), col="blue", lty=3)

legend("bottomright",
      c(expression(paste(hat('Y|X=1'))),
        expression(paste(hat('Y'), "|G=1, X=1")),
        expression(paste(hat('Y'), "|G=0, X=1")),
        expression(paste(hat(Y)[G %<-% 0], "|G=1, X=1 (true)")),
        expression(paste(hat(Y)[G %<-% 0], "|G=1, X=1 (est.)"))),
      col=c("black", "red", "blue", "blue", "blue"),
      lty=c(1, 1, 1, 2, 3),
      cex=0.5)

# In both plots, the estimated counterfactual CDF closely matches the CDF of the
# true counterfactual.

# Restore parameters
par(oldpar)

```

---

dat2mat

*dat2mat*


---

### Description

Generates matrices for decomposition of admission type which can be used in `plot_decomp`

### Usage

```
dat2mat(dat, score, group1, group2, nquant = 20, cats = unique(dat$reason))
```

**Arguments**

dat	data frame with population data, such as output from sim_pop_data. Must include a column reason
score	risk scores corresponding to dat
group1	indices for group 1
group2	indices for group 2
nquant	number of quantiles of code to use; default 20
cats	vector of strings giving names of admission categories; default the unique values in dat\$reason. Can include NAs.

**Details**

Generates two matrices with the following specifications: Each matrix corresponds to one group. Columns are named with the admission types to be plotted. Any admission types including the string 'Died' are counted as deaths. If the matrix has N rows, these are interpreted as corresponding to N score quantiles. The (i,j)th entry of the matrix is the number of people admitted for reason i with a score greater than or equal to  $(j-1)/N$  and less than  $(j/N)$  who are in that group.

**Value**

list with two objects matrix1 and matrix2 giving output matrices

**Examples**

```
# See vignette
```

---

decomposition\_matrix *Decomposition matrix*

---

**Description**

Matrix giving frequency of admission types for various groups at various score thresholds. Row names are of the form vX\_Y\_qZ, where X is version (3 or 4), Y is cohort (e.g., all, over 65, island postcode) and Z is quantile (1-20) of score. Column names are cause of admission or cause of death.

**Usage**

```
decomposition_matrix
```

**Format**

An object of class data.frame with 520 rows and 41 columns.

---

demographic\_parity      *demographic\_parity*

---

### Description

Estimates demographic parity for a risk score (essentially cumulative distribution function)

### Usage

```
demographic_parity(
  scores,
  group1,
  group2,
  cutoffs = seq(min(scores, na.rm = TRUE), max(scores, na.rm = TRUE), length = 100)
)
```

### Arguments

scores	vector of risk scores
group1	indices of group 1
group2	indices of group 2
cutoffs	score cutoffs at which to estimate DP (default 100 evenly-spaced)

### Value

matrix of dimension  $\text{length}(\text{cutoffs}) \times 4$ , with  $(i, 2g-1)$ th entry the proportion of scores in group  $g$  which are less than or equal to the  $i$ th cutoff value and  $(i, 2g)$ th entry the approximate standard error of the  $(i, 2g-1)$ th entry

### Examples

```
# See vignette
```

---

drawperson      *drawperson*

---

### Description

Draws a simple stock image of a person.

**Usage**

```

drawperson(
  xloc = 0,
  yloc = 0,
  scale = 1,
  headsize = 0.16,
  headangle = pi/8,
  headloc = 0.5,
  necklength = 0.1,
  shoulderwidth = 0.1,
  shouldersize = 0.05,
  armlength = 0.4,
  armangle = 7 * pi/8,
  armwidth = 0.08,
  leglength = 0.5,
  legangle = 9 * pi/10,
  legwidth = 0.15,
  torsolength = 0.4,
  ...
)

```

**Arguments**

xloc	x-axis offset from origin
yloc	y-axis offset from origin
scale	scale upwards from 1x1 box
headsize	head size
headangle	half angle of neck in terms of head
headloc	location of centre of head relative to origin with scale 1
necklength	neck length
shoulderwidth	shoulder width
shouldersize	size radius of arc for shoulder
armlength	arm length
armangle	angle of arm from horizontal
armwidth	width of arm
leglength	leg length
legangle	angle of leg from horizontal
legwidth	width of leg
torsolength	length of torso
...	other parameters passed to polygon()

**Details**

Draws a figure at a particular location. With defaults, has centre at origin and fits in 1x1 box.

Dimensions customisable

**Value**

invisibly returns co-ordinates

**Examples**

```
plot(0,xlim=c(-1,1),ylim=c(-1,1),type="n")
drawperson(0,0,1,col="yellow",border="red",lwd=3,lty=2)
```

---

drawprop

*drawprop*

---

**Description**

Illustrates a proportion as a set of people who are blue rather than red.

**Usage**

```
drawprop(prop, ci, nxy = 10, col1 = "maroon", col2 = "lightblue", ...)
```

**Arguments**

prop	the proportion to illustrate
ci	half the 95% CI width of the proportion.
nxy	illustrate on an n x n grid
col1	colour to put the 'in' proportion
col2	the other colour
...	passed to 'plot'

**Details**

Why anyone would want to think about a proportion this way is beyond the understanding of the authors, but the people have spoken.

**Value**

No return value, draws a figure

**Examples**

```
# See vignette
```

---

for_breakdown	<i>for_breakdown</i>
---------------	----------------------

---

### Description

For a given category (e.g., 'male', 'over 65') considers

1. all admissions for people in that category
2. all admissions for people in that category for which the SPARRA score was less than some threshold (e.g., false negatives)

### Usage

```
for_breakdown(
  decomp_table,
  group,
  threshold,
  inc_died = TRUE,
  ldiff = 0.005,
  ci = 0.95,
  xlimit = c(-0.05, 0.35),
  ylimit = c(-0.04, 0.04)
)
```

### Arguments

decomp_table	matrix for group; see specification in description
group	name of group
threshold	cutoff, rounded to nearest 0.05
inc_died	set to TRUE to include a second panel showing 'death' type admissions
ldiff	specifically label points this far from xy line
ci	set to a value <1 to draw confidence intervals at that value, or FALSE to not draw confidence intervals.
xlimit	limits for x axis; default c(-0.05,0.35)
ylimit	limits for y axis; default c(-0.04,0.04)

### Details

For each of these groups, we consider the breakdown of medical admission types. We then plot the frequency of admission types in group 1 against the difference in frequencies between group 1 and group 2 (group 2 minus group 1). An admission type which is relatively more common in group (1) indicates that, in the relevant category, the admission type tends to be associated with higher SPARRA scores (and is in a sense easier to predict). Such admission types will correspond to points below the line  $y=0$ . Admission types which are relatively more common in group 2 correspond to

those which are relatively harder to predict. These correspond to points above the line  $y=0$ . Since points are close together, only those greater than a certain distance from 0 are marked.

Takes as an argument a matrix in which The matrix shows only data for the group in question. Columns are named with the admission types to be plotted. Any admission types including the string 'Died' are counted as deaths. If the matrix has  $N$  rows, these are interpreted as corresponding to  $N$  score quantiles in increasing order. The  $(i,j)$ th entry of the matrix is the number of people admitted for reason  $i$  with a score greater than or equal to  $(j-1)/N$  and less than  $(j/N)$  who are in that group.

### Value

ggplot figure (invisible)

### Examples

```
# See vignette
```

---

getcal	getcal()
--------	----------

---

### Description

Produces a set of points for a calibration plot.

### Usage

```
getcal(
  y,
  ypred,
  n = 10,
  kernel = FALSE,
  kernel_sd = 0.05,
  alpha = 0.05,
  c0 = 0,
  c2 = 0.1
)
```

### Arguments

y	class labels, 0/1 or logical
ypred	predictions $\Pr(Y=1)$ , numeric vector
n	number of subintervals/points
kernel	set to TRUE to use kernel method
kernel_sd	kernel width for kernel method; see above
alpha	return a pointwise confidence envelope for conservative $1-\alpha$ confidence interval



- `c0` for computing maximum bias; assume true covariance function is of the form  $a_0 + a_1x + a_2x^2$ , with  $|a_0| < c_0$ ,  $|a_2| < c_2$  ( $c_1$  does not matter)
- `c2` for computing maximum bias; assume true covariance function is of the form  $a_0 + a_1x + a_2x^2$ , with  $|a_0| < c_0$ ,  $|a_2| < c_2$  ( $c_1$  does not matter)

**Details**

Uses either a binning method or a kernel method to determine height of points.

In both methods, considers  $n$  equally spaced subintervals of  $(0,1)$

**Value**

a list with components  $x$  (expected calibration),  $y$  (observed calibration),  $n$  (number of samples in bins, if relevant), lower/upper (confidence interval on  $y$ )

**Examples**

```
# See vignette
```

---

<code>getprc</code>	<code>getprc()</code>
---------------------	-----------------------

---

**Description**

Comprehensive plotting function for precision-recall curve. Also calculates AUPRC and standard error.

**Usage**

```
getprc(y, ypred, cv = NULL, res = 100)
```

**Arguments**

- `y` class labels, 0/1 or logical
- `ypred` predictions  $\Pr(Y=1)$ , numeric vector
- `cv` cross-validation fold assignments, if relevant. Changes estimate of standard error.
- `res` resolution. Returns this many equally-spaced points along the curve. Set `res` to null to return all points.

**Details**

Rather than returning points corresponding to every cutoff, only returns a representative sample of equally-spaced points along the curve.

Does not plot anything. Object can be plotted in a default way.

**Value**

list containing: ppv, ppv for res points in every cv fold; sens, sensitivity for res points in every cv fold; auc, areas under the curve for each fold and average (note length is 1 greater than number of CV folds); se, standard error for AUC in each fold and standard error for average auc (note length is 1 greater than number of CV folds)

**Examples**

```
# See vignette
```

---

getroc	<i>getroc()</i> Comprehensive plotting function for receiver-operator characteristic curve. Also calculates AUROC and standard error.
--------	---

---

**Description**

Rather than returning points corresponding to every cutoff, only returns a representative sample of equally-spaced points along the curve.

**Usage**

```
getroc(y, ypred, cv = NULL, res = 100)
```

**Arguments**

y	class labels, 0/1 or logical
ypred	predictions Pr(Y=1), numeric vector
cv	cross-validation fold assignments, if relevant. Changes estimate of standard error.
res	resolution. Returns this many equally-spaced points along the curve. Set res to null to return all points.

**Details**

SE of AUROC with no CV structure is from Hanley and McNeil 1982. SE of AUROC with CV folds is from LeDell et al 2012

Does not plot anything. Object can be plotted in a default way.

**Value**

list containing: spec, specificity for res points in every cv fold; sens, sensitivity for res points in every cv fold; auc, areas under the curve for each fold and average (note length is 1 greater than number of CV folds); se, standard error for AUC in each fold and standard error for average auc (note length is 1 greater than number of CV folds)

**Examples**

```
# See vignette
```

---

groupmetric_2panel	<i>groupmetric_2panel</i> Draws plots of a group fairness metric with a second panel underneath
--------------------	---

---

**Description**

groupmetric\_2panel Draws plots of a group fairness metric with a second panel underneath

**Usage**

```
groupmetric_2panel(
  objs,
  labels = names(objs),
  col = 1:length(objs),
  yrange = NULL,
  ci_col = col,
  highlight = NULL,
  logscale = FALSE,
  lpos = c(1, 0),
  yrange_lower = NULL,
  legend_title = ""
)
```

**Arguments**

objs	list of fairness objects. Each should contain sub-objects 'x', 'y' and 'ci', which specify x and y values and half-widths of confidence intervals around y.
labels	labels to use in legend
col	line colours
yrange	limit of y axis; defaults to 0,1
ci_col	confidence envelope colours. These will be transparent.
highlight	if non-null, draw a point at a particular cutoff
logscale	if TRUE, draw with log-scale.
lpos	legend position; as accepted by ggplot legend.position
yrange_lower	y range for lower plot. If NULL, generates automatically
legend_title	title for legend, defaults to nothing

**Value**

Silently return ggplot object

**Examples**

```
# See vignette
```

---

group_fairness	group_fairness
----------------	----------------

---

### Description

Estimates group fairness metric according to a specification vector of the form

### Usage

```
group_fairness(
  specs,
  scores,
  target,
  group1,
  group2,
  cutoffs = seq(min(scores, na.rm = TRUE), max(scores, na.rm = TRUE), length = 100)
)
```

### Arguments

specs	specification vector; see description
scores	vector of risk scores
target	vector of values of target (which risk score aims to predict)
group1	indices of group 1
group2	indices of group 2
cutoffs	score cutoffs at which to estimate metric (default 100 evenly-spaced)

### Details

$c(A1, B1, C1, A2, B2, C2)$

encoding a probability

$P(A1, B1, C1 | A2, B2, C2)$

where

$A1/A2$  are events coded by 1: 'score >= cutoff'; 0: 'score < cutoff' and NA: 1/TRUE  $B1/B2$  are events coded by 1: 'target = TRUE'; 0: 'target = FALSE' and NA: 1/TRUE  $C1/C2$  are events coded by 1: 'group = g'; and NA: 1/TRUE

For example,  $specs = c(NA, 1, NA, 0, NA, 1)$  would encode false omission rate:

$P(\text{target} = \text{TRUE} | \text{score} < \text{cutoff}, \text{group} = g)$

### Value

matrix of dimension  $\text{length}(\text{cutoffs}) \times 4$ , with  $(i, 2g-1)$ th entry the relevant fairness metric for group  $g$  at the  $i$ th cutoff value and  $(i, 2g)$ th entry the approximate standard error of the  $(i, 2g-1)$ th entry

**Examples**

```
# See vignette
```

---

integral	<i>integral()</i> Quick form for trapezoidal integration over range of x
----------	--

---

**Description**

integral() Quick form for trapezoidal integration over range of x

**Usage**

```
integral(x, y = NULL)
```

**Arguments**

x	x co-ordinates, or nx2 matrix of points
y	y co-ordinates

**Value**

trapezoidal estimate of integral of the xth value of y over range of x.

---

logistic	<i>Logistic</i>
----------	-----------------

---

**Description**

Logistic function:  $1/(1+\exp(-x))$

**Usage**

```
logistic(x)
```

**Arguments**

x	argument
---	----------

**Value**

value of logistic(x)

**Examples**

```
# Plot
x=seq(-5,5,length=1000)
plot(x,logistic(x),type="l")
```

---

logit	<i>Logit</i>
-------	--------------

---

**Description**

Logit function:  $-\log((1/x)-1)$

**Usage**

logit(x)

**Arguments**

x                    argument

**Value**

value of logit(x); na if x is outside (0,1)

**Examples**

```
# Plot
x=seq(0,1,length=100)
plot(x,logit(x),type="l")

# Logit and logistic are inverses
x=seq(-5,5,length=1000)
plot(x,logit(logistic(x)),type="l")
```

---

phs_colours	<i>phs_colours</i>
-------------	--------------------

---

**Description**

Copied from github, "Public-Health-Scotland/phsstyles". Public Health Scotland colour scheme. Internal function.

**Usage**

```
phs_colours(colourname = NULL, keep_names = FALSE)
```

**Arguments**

colourname        name of colour; usually something like phs-blue. If NULL returns all colours.  
keep\_names        keep names of colours in return list. Defaults to false.

**Value**

vector of colours, optionally with names.

---

plot.sparraCAL	<i>Plot function for class sparraCAL</i>
----------------	--

---

**Description**

Plot function for class sparraCAL

**Usage**

```
## S3 method for class 'sparraCAL'
plot(
  x,
  cols = rep(phis_colours("phis-blue"), dim(x$sens)[1]),
  add = FALSE,
  add_xy_line = TRUE,
  ...
)
```

**Arguments**

x	output from getcal()
cols	colour to draw lines
add	set to FALSE to add to existing plot
add_xy_line	set to TRUE to draw an X-Y reference line.
...	passed to lines()

**Value**

No return value, draws a figure

**Examples**

```
# See vignette
```

---

plot.sparraPRC      *Plot function for class above*

---

### Description

Plot function for class above

### Usage

```
## S3 method for class 'sparraPRC'
plot(
  x,
  addauc = FALSE,
  cols = rep(phs_colours("phs-blue"), dim(x$sens)[1]),
  ...
)
```

### Arguments

x	output from getprc()
addauc	set to TRUE to add text to the plot showing the (mean) AUC and SE.
cols	colour to draw lines
...	passed to plot()

### Value

No return value, draws a figure

### Examples

```
# See vignette
```

---

plot.sparraROC      *Plot function for class sparraROC*

---

### Description

Plot function for class sparraROC

### Usage

```
## S3 method for class 'sparraROC'
plot(
  x,
  addauc = FALSE,
  cols = rep(phs_colours("phs-blue"), dim(x$sens)[1]),
  ...
)
```



**Arguments**

x	output from getroc()
addauc	set to TRUE to add text to the plot showing the (mean) AUC and SE.
cols	colour to draw lines
...	passed to plot()

**Value**

No return value, draws a figure

**Examples**

```
# See vignette
```

---

plot_decomp	<i>plot_decomp</i>
-------------	--------------------

---

**Description**

Plots a bar graph of decomposition of FORP by cause of admission

**Usage**

```
plot_decomp(decomp1, decomp2, threshold, labels, inc_died = TRUE)
```

**Arguments**

decomp1	matrix for first group; see specification in description
decomp2	matrix for second group; see specification in description
threshold	score threshold to plot (between 0 and 1)
labels	labels for group 1 and group 2
inc_died	set to TRUE to include a second panel showing 'death' type admissions

**Details**

Takes two matrices as input with the following specifications: Each matrix corresponds to one group. Columns are named with the admission types to be plotted. Any admission types including the string 'Died' are counted as deaths. If the matrix has N rows, these are interpreted as corresponding to N score quantiles in increasing order. The (i,j)th entry of the matrix is the number of people admitted for reason i with a score greater than or equal to (j-1)/N and less than (j/N) who are in that group.

**Value**

Silently return ggplot object

**Examples**

```
# See vignette
```

---

prc_2panel	<i>prc_2panel</i> Draws a PRC curve (with legend) with a second panel underneath showing precision difference.
------------	--

---

### Description

`prc_2panel` Draws a PRC curve (with legend) with a second panel underneath showing precision difference.

### Usage

```
prc_2panel(
  prcs,
  labels = names(prcs),
  col = 1:length(prcs),
  highlight = NULL,
  yrange_lower = NULL,
  legend_title = ""
)
```

### Arguments

<code>prcs</code>	list of <code>sparraPRC</code> objects.
<code>labels</code>	labels to use in legend
<code>col</code>	line colours
<code>highlight</code>	if non-null, draw a point at a particular cutoff
<code>yrange_lower</code>	y range for lower plot. If <code>NULL</code> , generates automatically
<code>legend_title</code>	title for legend, defaults to nothing

### Value

Silently return `ggplot` object

### Examples

```
# See vignette
```

---

roc_2panel	<i>roc_2panel</i> Draws a ROC curve (with legend) with a second panel underneath showing sensitivity difference.
------------	--

---

### Description

roc\_2panel Draws a ROC curve (with legend) with a second panel underneath showing sensitivity difference.

### Usage

```
roc_2panel(  
  rocs,  
  labels = names(rocs),  
  col = 1:length(rocs),  
  xy_col = phs_colours("phs-magenta"),  
  highlight = NULL,  
  yrange_lower = NULL,  
  legend_title = ""  
)
```

### Arguments

rocs	list of sparraROC objects
labels	labels to use in legend; default to names of rocs.
col	line colours
xy_col	line colour for x-y line, defaults to red
highlight	if non-null, add a point at this cutoff
yrange_lower	y range for lower plot. If NULL, generates automatically
legend_title	title for legend, defaults to nothing

### Value

Invisibly returns plot as ggplot object

### Examples

```
# See vignette
```

---

sim_pop_data	<i>sim_pop_data</i>
--------------	---------------------

---

## Description

Simulates population data with a reasonably realistic joint distribution

## Usage

```
sim_pop_data(
  npop,
  coef_adjust = 4,
  offset = 1,
  vcor = NULL,
  coefs = c(2, 1, 0, 5, 3, 0, 0),
  seed = 12345,
  incl_id = TRUE,
  incl_reason = TRUE
)
```

## Arguments

npop	population size
coef_adjust	inverse scale for all (true) coefficients (default 4): lower means that hospital admissions are more predictable from covariates.
offset	offset for logistic model (default 1): higher means a lower overall prevalence of admission
vcor	a valid 5x5 correlation matrix (default NULL), giving correlation between variables. If 'NULL', values roughly represents realistic data.
coefs	coefficients of age, male sex, non-white ethnicity, number of previous admissions, and deprivation decile on hospital admissions, Default (2,1,0,5,3). Divided through by coef_adjust.
seed	random seed (default 12345)
incl_id	include an ID column (default TRUE)
incl_reason	include a column indicating reason for admission.

## Details

Simulates data for a range of people for the variables

- Age (age)
- Sex (sexM; 1 if male)
- Race/ethnicity (raceNW: 1 if non-white ethnicity)
- Number of previous hospital admissions (PrevAdm)

- Deprivation decile (SIMD: 1 most deprived, 10 least deprived. NOTE - opposite to English IMD)
- Urban-rural residence status (urban\_rural: 1 for rural)
- Mainland-island residence status (mainland\_island: 1 for island)
- Hospital admission (target: 1/TRUE if admitted to hospital in year following prediction date)

Can optionally add an ID column.

Optionally includes an admission reason for samples with target=1. These admission reasons roughly correspond to the first letters of ICD10 categories, and can either correspond to an admission or death. Admission reasons are simulated with a non-constant multinomial distribution which varies across age/sex/ethnicity/urban-rural/mainland-island/PrevAdm values in a randomly- chosen way. The distributions of admission reasons are *not* however chosen to reflect real distributions, nor are systematic changes in commonality of admission types across categories intended to appear realistic.

### Value

data frame with realistic values.

### Examples

```
# Simulate data
dat=sim_pop_data(10000)
cor(dat[,1:7])

# See vignette
```

# Index

- \* **data**
  - [all\\_data, 5](#)
  - [decomposition\\_matrix, 11](#)
- \* **fairness**
  - [all\\_data, 5](#)
  - [decomposition\\_matrix, 11](#)
- \* **sparra**
  - [all\\_data, 5](#)
  - [decomposition\\_matrix, 11](#)

[ab, 2](#)  
[adjusted\\_fdr, 3](#)  
[adjusted\\_for, 4](#)  
[all\\_data, 5](#)

[build\\_diff, 5](#)

[cal\\_2panel, 6](#)  
[counterfactual\\_yhat, 7](#)

[dat2mat, 10](#)  
[decomposition\\_matrix, 11](#)  
[demographic\\_parity, 12](#)  
[drawperson, 12](#)  
[drawprop, 14](#)

[for\\_breakdown, 15](#)

[getcal, 16](#)  
[getprc, 17](#)  
[getroc, 18](#)  
[group\\_fairness, 20](#)  
[groupmetric\\_2panel, 19](#)

[integral, 21](#)

[logistic, 21](#)  
[logit, 22](#)

[phs\\_colours, 22](#)  
[plot.sparraCAL, 23](#)  
[plot.sparraPRC, 24](#)  
[plot.sparraROC, 24](#)  
[plot\\_decomp, 25](#)  
[prc\\_2panel, 26](#)  
[roc\\_2panel, 27](#)  
[sim\\_pop\\_data, 28](#)