Package ‘verification’

July 15, 2015

Version 1.42

Date 2015-07-10

Title Weather Forecast Verification Utilities

Author NCAR - Research Applications Laboratory

Maintainer Eric Gilleland <ericg@ucar.edu>

Depends R (>= 2.10), methods, fields, boot, CircStats, MASS, dtw

Imports graphics, stats

Description Utilities for verifying discrete, continuous and probabilistic forecasts, and forecasts expressed as parametric distributions are included.

License GPL (>= 2)

LazyData yes

Repository CRAN

Date/Publication 2015-07-15 01:30:51

NeedsCompilation no

R topics documented:

`attribute` *Attribute plot*

Description

An attribute plot illustrates the reliability, resolution and uncertainty of a forecast with respect to the observation. The frequency of binned forecast probabilities are plotted against proportions of binned observations. A perfect forecast would be indicated by a line plotted along the 1:1 line. Uncertainty is described as the vertical distance between this point and the 1:1 line. The relative frequency for each forecast value is displayed in parenthesis.

Usage

```
## Default S3 method:
attribute(x, obar.i,  prob.y = NULL, obar = NULL,
    class = "none", main = NULL, CI = FALSE, n.boot = 100, alpha = 0.05,
    tck = 0.01, freq = TRUE, pred = NULL, obs = NULL, thres = thres,
    bins = FALSE, ...)

## S3 method for class 'prob.bin'
attribute(x, ...)
```

Arguments

`x`	A vector of forecast probabilities or a “prob.bin” class object produced by the `verify` function.
`obar.i`	A vector of observed relative frequency of forecast bins.

prob.y	Relative frequency of forecasts of forecast bins.
obar	Climatological or sample mean of observed events.
class	Class of object. If prob.bin, the function will use the data to estimate confidence intervals.
main	Plot title.
CI	Confidence Intervals. This is only an option if the data is accessible by using the verify command first. Calculated by bootstrapping the observations and prediction, then calculating PODy and PODn values.
n.boot	Number of bootstrap samples.
alpha	Confidence interval. By default = 0.05
tck	Tick width on confidence interval whiskers.
freq	Should the frequecies be plotted. Default = TRUE
pred	Required to create confidence intervals
obs	Required to create confidence intervals
thres	thresholds used to create bins for plotting confidence intervals.
bins	Should probabilities be binned or treated as unique predictions?
...	Graphical parameters

Note

Points and bins are plotted at the mid-point of bins. This can create distorted graphs if forecasts are created at irregular intervals.

Author(s)

Matt Pocernich

References

Hsu, W. R., and A. H. Murphy, 1986: The attributes diagram: A geometrical framework for assessing the quality of probability forecasts. *Int. J. Forecasting* **2**, 285–293.

Wilks, D. S. (2005) *Statistical Methods in the Atmospheric Sciences* Chapter 7, San Diego: Academic Press.

See Also

verify reliability.plot

Examples

```
## Data from Wilks, table 7.3 page 246.
 y.i   <- c(0,0.05, seq(0.1, 1, 0.1))
 obar.i <- c(0.006, 0.019, 0.059, 0.15, 0.277, 0.377, 0.511,
             0.587, 0.723, 0.779, 0.934, 0.933)
 prob.y<- c(0.4112, 0.0671, 0.1833, 0.0986, 0.0616, 0.0366,
            0.0303,  0.0275, 0.245, 0.022, 0.017, 0.203)
```

```
 obar<- 0.162

attribute(y.i, obar.i, prob.y, obar, main = "Sample Attribute Plot")

## Function will work with a ``prob.bin'' class objects as well.
## Note this is a random forecast.
obs<- round(runif(100))
pred<- runif(100)

A<- verify(obs, pred, frcst.type = "prob", obs.type = "binary")
attribute(A, main = "Alternative plot", xlab = "Alternate x label" )
## to add a line from another model
obs<- round(runif(100))
pred<- runif(100)

B<- verify(obs, pred, frcst.type = "prob", obs.type = "binary")
lines.attrib(B, col = "green")


## Same with confidence intervals
attribute(A, main = "Alternative plot", xlab = "Alternate x label", CI =
TRUE)

#### add lines to plot
data(pop)
d <- pop.convert()
## internal function used to
## make binary observations for
## the pop figure.

### note the use of bins = FALSE
mod24 <- verify(d$obs_rain, d$p24_rain,
    bins = FALSE)

mod48 <- verify(d$obs_rain, d$p48_rain,
    bins = FALSE)
plot(mod24, freq = FALSE)

lines.attrib(mod48, col = "green",
    lwd = 2, type = "b")
```

`brier` *Brier Score*

Description

Calculates verification statistics for probabilistic forecasts of binary events.

Usage

```
brier(obs, pred, baseline, thresholds = seq(0,1,0.1), bins = TRUE, ... )
```

Arguments

`obs`	Vector of binary observations
`pred`	Vector of probablistic predictions [0,1]
`baseline`	Vector of climatological (no - skill) forecasts. If this is null, a sample climatology will be calculated.
`thresholds`	Values used to bin the forecasts. By default the bins are {[0,0.1), [0.1, 0.2), ...} .
`bins`	If TRUE, thresholds define bins into which the probablistic forecasts are entered and assigned the midpoint as a forecast. Otherwise, each unique forecast is considered as a seperate forecast. For example, set bins to FALSE when dealing with a finite number of probabilities generated by an ensemble forecast.
`...`	Optional arguments

Value

`baseline.tf`	Logical indicator of whether climatology was provided.
`bs`	Brier score
`bs.baseline`	Brier Score for climatology
`ss`	Skill score
`bs.reliability`	Reliability portion of Brier score.
`bs.resolution`	Resolution component of Brier score.
`bs.uncert`	Uncertainty component of Brier score.
`y.i`	Forecast bins – described as the center value of the bins.
`obar.i`	Observation bins – described as the center value of the bins.
`prob.y`	Proportion of time using each forecast
`obar`	Forecast based on climatology or average sample observations.
`check`	Reliability - resolution + uncertainty should equal brier score.

Note

This function is used within `verify`.

Author(s)

Matt Pocernich

References

Wilks, D. S. (1995) *Statistical Methods in the Atmospheric Sciences* Chapter 7, San Diego: Academic Press.

Examples

```
#  probabilistic/ binary example

pred<- runif(100)
obs<- round(runif(100))
brier(obs, pred)
```

`check.func` *check loss function*

Description

Calculates the check loss function.

Usage

```
check.func(u, p)
```

Arguments

`u`	Value to be evaluated
`p`	Probability level [0,1]

Details

The check loss is calculated as $\rho_p(u) = (abs(u) + (2 * p - 1) * u)/2$.

Value

The check loss for value u and probability level p.

Note

This function is used within `quantileScore`.

Author(s)

Sabrina Bentzien

Examples

```
## The function is currently defined as
function (u, p)
{
    rho <- (abs(u) + (2 * p - 1) * u) * 0.5
  }
```

conditional.quantile *Conditional Quantile Plot*

Description

This function creates a conditional quantile plot as shown in Murphy, et al (1989) and Wilks (1995).

Usage

```
conditional.quantile(pred, obs, bins = NULL, thrs = c(10, 20), main, ... )
```

Arguments

pred	Forecasted value. *([n,1] vector, n = No. of forecasts)*
obs	Observed value.*([n,1] vector, n = No. of observations)*
bins	Bins for forecast and observed values. A minimum number of values are required to calculate meaningful statistics. So for variables that are continuous, such as temperature, it is frequently convenient to bin these values. *([m,1] vector, m = No. of bins)*
thrs	The minimum number of values in a bin required to calculate the 25th and 75th quantiles and the 10th and 90th percentiles respectively. The median values will always be displayed. *([2,1] vector)*
main	Plot title
...	Plotting options.

Value

This function produces a conditional.quantile plot. The `y` axis represents the observed values, while the `x` axis represents the forecasted values. The histogram along the bottom axis illustrates the frequency of each forecast.

Note

In the example below, the median line extends beyond the range of the quartile or 10th and 90th percentile lines. This is because there are not enough points in each bin to calculate these quartile values. That is, there are fewer than the limits set in the `thrs` input.

Author(s)

Matt Pocernich

References

Murphy, A. H., B. G. Brown and Y. Chen. (1989) Diagnostic Verification of Temperature Forecasts. *Weather and Forecasting*, **4**, 485–501.

Examples

```
set.seed(10)
m<- seq(10, 25, length = 1000)
frcst <- round(rnorm(1000, mean = m, sd = 2) )
obs<- round(rnorm(1000, mean = m, sd = 2 ))
bins <- seq(0, 30,1)
thrs<- c( 10, 20) # number of obs needed for a statistic to be printed #1,4 quartile, 2,3 quartiles

conditional.quantile(frcst, obs, bins, thrs, main = "Sample Conditional Quantile Plot")
#### Or plots a ``cont.cont'' class object.

obs<- rnorm(100)
pred<- rnorm(100)
baseline <- rnorm(100, sd = 0.5)

A<- verify(obs, pred, baseline = baseline,  frcst.type = "cont", obs.type = "cont")
 plot(A)
```

---

crps   *Continuous Ranked Probability Score*

Description

Calculates the crps for a forecast made in terms of a normal probability distribution and an observation expressed in terms of a continuous variable.

Usage

```
crps(obs, pred, ...)
```

Arguments

`obs`	A vector of observations.
`pred`	A vector or matrix of the mean and standard deviation of a normal distribution. If the vector has a length of 2, it is assumed that these values represent the mean and standard deviation of the normal distribution that will be used for all forecasts.
`...`	Optional arguments

Value

`crps`	Continous ranked probability scores
`CRPS`	Mean of crps
`ign`	Ignorance score
`IGN`	Mean of the ignorance score

Note

This function is used within verify.

Author(s)

Matt Pocernich

References

Gneiting, T., Westveld, A., Raferty, A. and Goldman, T, 2004: *Calibrated Probabilistic Forecasting Using Ensemble Model Output Statistics and Minimum CRPS Estimation.* Technical Report no. 449, Department of Statistics, University of Washington. [Available online at http://www.stat.washington.edu/www/research/reports/]

Examples

```
#  probabilistic/ binary example


x <- runif(100) ## simulated observation.
crps(x, c(0,1))

## simulated forecast in which mean and sd differs for each forecast.
frcs<- data.frame( runif(100, -2, 2), runif(100, 1, 3 ) )
crps(x, frcs)
```

---

crpsDecompostion *Decompostion of Continuous Ranked Probability Score*

Description

The CRPS measures the distance between the predicted and the observed cumulative density functions (CDFs) of scalar variables. Furthermore, the crpsDecomposition function provides the reliability and resolution terms obtained by the CRPS decomposition proposed by Hersbach. The alpha, beta matrices and Heavisides vectors of outliers calculated in the CRPS decomposition are also returned. To speed up calculation time, these matrices/vectors can then be used to recalculate the CRPS's in a bootstrap by using the crpsFromAlphaBeta function.

Usage

```
crpsDecomposition(obs, eps)
crpsFromAlphaBeta(alpha,beta,heaviside0,heavisideN)
```

Arguments

`obs`	Vector of observations
`eps`	Matrix of ensemble forecast. Each column represent a member.
`alpha`	Matrix of alpha (returned from crpsDecomposition)
`beta`	Vector of beta (returned from crpsDecomposition)
`heaviside0`	Vector of Heaviside for outlier i=0 (returned from crpsDecomposition)
`heavisideN`	Vector of Heaviside for outlier i=N (returned from crpsDecomposition)

Value

`CRPS`	CRPS score
`CRPSpot`	The potential CRPS (Resolution - Uncertainty)
`Reli`	The Reliability term of the CRPS
`alpha`	Matrix (Nobservation rows x Nmember +1 columns) of alpha used in the CRPS decomposition.
`beta`	Matrix (Nobservation rows x Nmember +1 columns) of beta used in the CRPS decomposition.
`heaviside0`	Vector (Nobservation length) of Heaviside for outlier i=0 used in the CRPS decomposition
`heavisideN`	Vector (Nobservation length) of Heaviside for outlier i=N used in the CRPS decomposition

Author(s)

Ronald Frenette <Ronald.Frenette@ec.gc.ca>

References

G. Candille, P. L. Houtekamer, and G. Pellerin: Verification of an Ensemble Prediction System against Observations, *Monthly Weather Review*,**135**, pp. 2688-2699

Hershcach, Hans, 2000. Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems. *Weather and Forecasting*, **15**, (5) pp. 559-570.

Examples

```
data(precip.ensemble)
x <- precip.ensemble[seq(5,5170,5),]

#Observations are in the column
obs<-x[,3]

#Forecast values of ensemble are in the column 4 to 54
eps<-x[,4:54]

#CRPS calculation
c<-crpsDecomposition(obs,eps)
```

```
#CRPS with alpha and beta
#Resampling indices
nObs<-length(obs)
i<-sample(seq(nObs),nObs,replace=TRUE)
crps2<-crpsFromAlphaBeta(c$alpha[i,],c$beta[i,],c$heaviside0[i],c$heavisideN[i])
```

---

disc.dat *Discrimination plot dataset.*

Description

This dataset is used to illustrate the `discrimination.plot` function.

Usage

```
data(disc.dat)
```

---

discrimination.plot *Discrimination plot*

Description

This function creates a plot of discrimination plots (overlay histograms). In the context of verification, this is often used to compare the distribution of event and no-event forecasts. This may be useful in comparing any set of observations. By default, boxplots of groups appear as upper marginal plots. These may be surpressed.

Usage

```
discrimination.plot(group.id, value, breaks = 11, main =
"Discrimination Plot", xlim = NULL, ylim = NULL,  legend =
FALSE, leg.txt = paste("Model", sort(unique(group.id)) ),   marginal = TRUE, cols =
seq(2, length(unique(group.id)) + 1), xlab = "Forecast",  ... )
```

Arguments

group.id	A vector identifying groups. A histogram is created for each unique value.
value	A vector of values corresponding to the group.id vector used to create the histograms
breaks	Number of breaks in the x-axis of the histogram. The range of values is taken to be the range of prediction values.
main	Title for plot.
xlim	Range of histogram - x axis - main plot coordinates.

`ylim`	Range of histogram - y axis - main plot coordinates.
`legend`	Should there be a legend? Default = FALSE
`leg.txt`	Legend text. If FALSE or if a marginal plot is created, no legend is added.
`cols`	A vector showing the colors to be used in the histograms and in the marginal boxplots
`marginal`	Should a boxplots be placed in the top margin? Defaults to TRUE
`xlab`	Label of the x-axis on the main plot.
`...`	Additional plotting options.

Author(s)

Matt Pocernich

Examples

```
 #  A sample forecast.

data(disc.dat)
discrimination.plot(disc.dat$group.id, disc.dat$frcst, main = "Default  Plot")

discrimination.plot(disc.dat$group.id, disc.dat$frcst, main = "New Labels", cex = 1.2,
leg.txt = c("Low", "Med", "High" ) )

discrimination.plot(disc.dat$group.id, disc.dat$frcst, main = "Without Marginal Plots ",
marginal = FALSE)
```

`fss` *Fractional Skill Score*

Description

Calculates the fractional skill score for spatial forecasts and spatial observations.

Usage

```
fss(obs, pred,w = 0, FUN = mean, ...)
```

Arguments

`obs`	A matrix of binomial observed values.
`pred`	A matrix of binomial forecasted values
`w`	Box width. When w = 0, each pixel is considered alone. w = 2 creates a box with a length of 5 units.
`FUN`	Function to be applied to each subgrid. By default, the mean of each box is used to calculate the fraction of each subgrid.
`...`	Optional arguments

Value

Return the fraction skill score.

Note

This function contain several loops and consequently is not particularly fast.

Author(s)

Matt Pocernich

References

Roberts, N.M and H.W. Lean: 2008: Scale-Selective Verification of Rainfall Accumulations from High-Resolution Forecasts of Convective Events. *Monthly Weather Review* **136**, 78-97.

Examples

```
grid<- list( x= seq( 0,5,,100), y= seq(0,5,,100))
obj<-Exp.image.cov( grid=grid, theta=.5, setup=TRUE)
look<- sim.rf( obj)
look[look < 0] <- 0

look2 <- sim.rf( obj)
look2[look2<0] <- 0

fss(look, look2, w=5)


## Not run:
#  The following example replicates Figure 4 in Roberts and Lean (2008).
####      examples

LAG <- c(1,3,11,21)
box.radius <- seq(0,24,2)

OUT <- matrix(NA, nrow = length(box.radius), ncol = length(LAG) )

for(w in 1:4){
```

```
FRCS <- OBS <- matrix(0, nrow = 100, ncol = 100)

obs.id       <- 50
OBS[,obs.id]  <- 1

FRCS[, obs.id + LAG[w]] <- 1

for(i in 1:length(box.radius)){

OUT[i, w] <- fss(obs = OBS, pred = FRCS, w = box.radius[i] )

} ### close i
} ### close w

b <- mean(OBS) ## base rate

fss.uniform  <- 0.5 + b/2
fss.random   <- b

matplot(OUT, ylim = c(0,1), ylab = "FSS", xlab = "grid squares",
type = "b", lty = 1, axes = FALSE , lwd = 2)

abline(h = c(fss.uniform, fss.random), lty = 2)
axis(2)
box()
axis(1, at = 1:length(box.radius), lab = 2*box.radius + 1)
grid()

legend("bottomright", legend = LAG, col = 1:4, pch = as.character(1:4),
 title = "Spatial Lag", inset = 0.02, lwd = 2 )

## End(Not run)
```

`leps` *Linear Error in Probability Space (LEPS)*

Description

Calculates the linear error in probability spaces. This is the mean absolute difference between the forecast cumulative distribution value (cdf) and the observation. This function creates the empirical cdf function for the observations using the sample population. Linear interpretation is used to estimate the cdf values between observation values. Therefore; this may produce awkward results with small datasets.

Usage

```
leps(x, pred, plot = TRUE, ... )
```

Arguments

`x`	A vector of observations or a verification object with "cont.cont" properties.
`pred`	A vector of predictions.
`plot`	Logical to generate a plot or not.
`...`	Additional plotting options.

Value

If assigned to an object, the following values are reported.

`leps.0`	Negatively oriented score on the [0,1] scale, where 0 is a perfect score.
`leps.1`	Positively oriented score proposed by Potts.

Author(s)

Matt Pocernich

References

DeQue, Michel. (2003) "Continuous Variables" **Chapter 5**, *Forecast Verification: A Practitioner's Guide in Atmospheric Science.*

Potts, J. M., Folland, C.K., Jolliffe, I.T. and Secton, D. (1996) "Revised 'LEPS' scores fore assessing climate model simulations and long-range forecasts." *J. Climate*, **9**, pp. 34-54.

Examples

```
 obs <- rnorm(100, mean = 1, sd = sqrt(50))
 pred<-  rnorm(100, mean = 10, sd = sqrt(500))

 leps(obs, pred, main = "Sample Plot")
## values approximated

OBS <- c(2.7, 2.9, 3.2, 3.3, 3.4, 3.4, 3.5, 3.8, 4, 4.2, 4.4, 4.4, 4.6,
5.8, 6.4)
PRED <- c(2.05, 3.6, 3.05, 4.5, 3.5, 3.0, 3.9, 3.2, 2.4, 5.3, 2.5, 2.8,
3.2, 2.8, 7.5)

a <- leps(OBS, PRED)
a
```

lines.verify — *Add lines to ROC or attribute diagrams*

Description

Add lines to attribute and verification diagrams from verify.prob.bin objects.

Usage

```
       ## S3 method for class 'roc'
lines(x,binormal = FALSE, ...)
        ## S3 method for class 'attrib'
lines(x,...)
```

Arguments

`x`	An object created by the verify function with the prob.bin class
`binormal`	Logical value indicating whether the lines to be added to a ROC plot are empirical or a binormal fit.
`...`	Optional arguments for the lines function. These include color, line weight (ltw) and line stype (lty)

Note

This will soon be replaced the a lines command constructed using S4 class properites.

Author(s)

Matt Pocernich

See Also

verify

measurement.error — *Skill score with measurement error.*

Description

Skill score that incorporates measurement error. This function allows the user to incorporate measurement error in an observation in a skill score.

Usage

```
measurement.error( obs, frcs = NULL, theta = 0.5, CI =
          FALSE, t = 1, u = 0, h = NULL, ...)
```

Arguments

`obs`	Information about a forecast and observation can be done in one of two ways. First, the results of a contingency table can be entered as a vector containing c(n11, n10, n01, n00), where n11 are the number of correctly predicted events and n01 is the number of incorrectly predicted non-events. Actual forecasts and observations can be used. In this case, obs is a vector of binary outcomes [0,1].
`frcs`	If obs is entered as a contingency table, this argument is null. If obs is a vector of outcomes, this column is a vector of probabilistic forecasts.
`theta`	Loss value (cost) of making a incorrect forecast by a non-event. Defaults to 0.5.
`CI`	Calculate confidence intervals for skill score.
`t`	Probability of forecasting an event, when an event occurs. A perfect value is 1.
`u`	Probability of forecasting that no event will occur, when and event occurs. A perfect value is 0.
`h`	Threshold for converting a probabilistic forecast into a binary forecast. By default, this value is NULL and the theta is used as this threshold.
`...`	Optional arguments.

Value

`z`	Error code
`k`	Skill score
`G`	Likelihood ratio statistic
`p`	p-value for the null hypothesis that the forecast contains skill.
`theta`	Loss value. Loss associated with an incorrect forecast of a non-event.
`ciLO`	Lower confidence interval
`ciHI`	Upper confidence interval

Author(s)

Matt Pocernich (R - code)

W.M Briggs <wib2004(at)med.cornell.edu> (Method questions)

References

W.M. Briggs, 2004. *Incorporating Cost in the Skill Score* Technical Report, wm-briggs.com/public/skillocst.pdf.

W.M. Briggs and D. Ruppert, 2004. *Assessing the skill of yes/no forecasts. Submitting to Biometrics.*

J.P. Finley, 1884. Tornado forecasts. *Amer. Meteor. J.* 85-88. (Tornado data used in example.)

Examples

```
DAT<- data.frame( obs = round(runif(50)), frcs = runif(50))

A<-   measurement.error(DAT$obs, DAT$frcs, CI = TRUE)
A
### Finley Data

measurement.error(c(28, 23, 72, 2680)) ## assuming perfect observation,
```

`multi.cont` *Multiple Contingency Table Statistics*

Description

Provides a variety of statistics for a data summarized in a contingency table. This will work for a 2 by 2 table, but is more useful for tables of greater dimensions.

Usage

```
multi.cont(DAT, baseline = NULL)
```

Arguments

`DAT`	A contingency table in the form of a matrix. It is assumed that columns represent observation, rows represent forecasts.
`baseline`	A vector indicating the baseline probabilities of each category. By default, it the baseline or naive forecasts is based on teh

Value

`pc`	Percent correct - events along the diagonal.
`bias`	Bias
`ts`	Threat score a.k.a. Critical success index (CSI)
`hss`	Heidke Skill Score
`pss`	Peirce Skill Score
`gs`	Gerrity Score
`pc2`	Percent correct by category (vector)
`h`	Hit Rate by category (vector)
`false.alarm.ratio`	False alarm ratio by category (vector)

Note

Some verification statistics for a contingency table assume that the forecasts and observations are ordered, while others do not. An example of an ordered or ordinal forecast is "low, medium and high". An example of an unordered or nominal forecast is "snow, rain, hail, and none." If the forecasts are ordered, it is possible to account for forecasts which are close to the the observed value. For example, the Gerrity score takes this closeness into account. The Pierce Skill Score does not.

For ordered forecast, it is assumed that the columns and rows of the input matrix are ordered sequentially.

When multiple values are returned, as in the case of pc2, h, f and false.alarm.ratio, these values are conditioned on that category having occurred. For example, in the example included in Jolliffe, given that a below average temperature was observed, the forecast had a bias of 2.3 and had a 0.47 chance of being detected.

Author(s)

Matt Pocernich

References

Gerrity, J.P. Jr (1992). A note on Gandin and Murphy's equitable skill score. *Mon. Weather Rev.*, **120**, 2707-2712.

Jolliffe, I.T. and D.B. Stephenson (2003). Forecast verification: a practitioner's guide in atmospheric science. John Wiley and Sons. See chapter 4 concerning categorical events, written by R. E. Livezey.

See Also

`binary.table`

Examples

```
DAT<- matrix(c(7,4,4,14,9,8,14,16,24), nrow = 3) # from p. 80 - Jolliffe
multi.cont(DAT)

DAT<- matrix(c(3,8,7,8,13,14,4,18,25), ncol = 3) ## Jolliffe JJA
multi.cont(DAT)

DAT<- matrix(c(50,47,54,91,2364,205,71,170,3288), ncol = 3) # Wilks p. 245
multi.cont(DAT)

DAT<- matrix(c(28, 23, 72, 2680 ), ncol = 2) ## Finley
multi.cont(DAT)
## Finnish clouds
DAT<- matrix(c(65, 10, 21, 29,17,48, 18, 10, 128), nrow = 3, ncol = 3, byrow = TRUE)
multi.cont(DAT)
 ### alternatively, the verify function and summary can be used.

 mod <- verify(DAT, frcst.type = "cat", obs.type = "cat")
```

```
summary(mod)
```

observation.error *Observation Error*

Description

Quantifies observation error through use of a “Gold Standard” of observations.

Usage

```
observation.error(obs, gold.standard = NULL, ...)
```

Arguments

obs	Observation made by method to be quantified. This information can be entered two ways. If obs is a vector of length 4, it is assumed that is contains the values c(n11, n10, n01, n00), where n11 are the number of correctly predicted events and n01 is the number of incorrectly predicted non-events.
gold.standard	The gold standard. This is considered a higher quality observation (coded {0, 1 }).
...	Optional arguments.

Value

t	Probability of forecasting an event, when an event occurs. A perfect value is 1.
u	Probability of forecasting that no event will occur, when and event occurs. A perfect value is 0.

Note

This function is used to calculate values for the measurement.error function.

Author(s)

Matt Pocernich

See Also

measurement.error

Examples

```
obs <- round(runif(100))
gold <- round(runif(100) )
observation.error(obs, gold)

## Pirep gold standard

observation.error(c(43,10,17,4) )
```

---

performance.diagram *Performance Diagram*

Description

Creates plot displaying multiple skill scores on a single plot

Usage

```
performance.diagram(...)
```

Arguments

...	Optional plotting parameters.

Note

Currently this function just produces the base plot. Points summarizing model performance can be added using the points function.

Author(s)

Matt Pocernich

References

Roebber, P.J. (2009). Visualizing Multiple Measures of Forecast Quality, *Weather and Forecasting*. **24**, pp - 601 - 608.

Examples

```
performance.diagram(main = "Sample Plot")
RB1 <- matrix(c(95, 55, 42, 141), ncol = 2)
## at point
pts     <- table.stats(RB1)
boot.pts <- table.stats.boot(RB1, R = 100   )
## add confidence intervals
segments(x0=1-pts$FAR, y0=boot.pts["up","pod"],
    x1=1-pts$FAR, y1=boot.pts["dw", "pod"], col=2, lwd=2)
```

```
segments(x0=1-boot.pts["up","far"], y0=pts$POD,
    x1=1-boot.pts["dw","far"], y1=pts$POD, col=2, lwd=2)
points(1 - pts$FAR, pts$POD, col = 2, cex = 2)
```

---

pop *Probability of precipitation (pop) data.*

Description

These datasets are used to illustrate several functions including `value` and `roc.plot`.

These forecasts are summaries of 24-hour probability of precipitation forecasts that were made by the Finnish Meteorological Institute (FMI) during 2003, for daily precipitation in the city of Tampere in south central Finland. Light precipitation is considered rainfall greater than .2 mm. Rainfall accumulation is considered values greater than 4.4 mm. Rows of data either missing forecasts or observations have been removed.

This data has been kindly provided by Dr. Pertti Nurmi of the Finnish Meteorological Institute. http://www.cawcr.gov.au/projects/verification/POP3/POP3.html

Usage

```
data(pop)
```

---

precip.ensemble *An ensemble of precipitation forecasts*

Description

This is an example of an ensemble of precipitation forecasts. The data set contains forecast for 517 days (3 monsoon seasons) at lead times of 1 to 10 days. Observations and forecasts are in millimeters.

predcomp.test *Time Series Prediction Comparison Test*

Description

Forecast prediction comparison test for two competing forecasts against an observation.

Usage

```
predcomp.test(x, xhat1, xhat2, alternative = c("two.sided", "less", "greater"),
    lossfun = "losserr", lossfun.args = NULL, test = c("DM", "HG"), ...)

losserr(x, xhat, method = c("abserr", "sqerr", "simple", "power",
    "corrskill", "dtw"), scale = 1, p = 1, dtw.interr = c("abserr",
    "sqerr", "simple", "power"), ...)

exponentialACV(x, y, ...)

## S3 method for class 'predcomp.test'
summary(object, ...)
```

Arguments

`x,xhat1,xhat2,xhat`
: numeric vectors giving the verification data and each competing forecast model output (1 and 2). For `losserr`, `xhat` is a numeric giving a single forecast model output (i.e., by default the function is called internally by `predcomp.test` once for `xhat1` and once for `xhat2`). For `exponentialACV`, see argument `y` below.

`y`
: `x` for `exponentialACV` is a numeric giving the separation distance, and `y` a numeric giving the autocovariance values.

`object`
: list object of class "predcomp.test" as returned by `predcomp.test`.

`alternative`
: character string stating which type of hypothesis test to conduct.

`lossfun`
: character string naming the loss function to call. The default, `losserr`, calls one of several methods depending on its `method` argument. Any function that takes `x` and `xhat` numeric vectors as arguments and returns a numeric vector of the same length can be used.

`lossfun.args`
: List providing additional arguments to `lossfun`.

`test`
: character string stating whether to run the Diebold-Mariano type of test or the Hering-Genton modification of it (i.e., use a parametric autocovariance function).

`method,dtw.interr`
: character string stating which type of loss (or skill) function to use. In the case of `dtw.interr`, this is the loss function for the intensity part of the error only.

scale numeric giving a value by which to scale the loss function. In the case of "dtw", this is only applied to the intensity part of the loss function, and can be used to scale the influence of the intensity vs. temporal lag errors. See Details section for more.

p numeric only used by the "power" loss function.

... For predcomp.test, these are any additional arguments to the acf function besides x, type and plot, which may not be passed.

For losserr, these are any additional arguments to dtw except for x, y, and step.pattern, which may not be passed.

For exponentialACV these are any optional arguments to nls except for formula and data. If start is not passed, then reasonable starting values are calculated and passed in for this argument.

For the summary method function, these are not used.

Details

This function performs the analyses described in Gilleland and Roux (2014); although note that while working on another manuscript (Gilleland and Hering, in preparation), a better optimization routine has replaced the one used in said paper, which has been thoroughly tested to yield good size and power under a variety of temporal dependence structures, as well as having far fewer situations where a fit cannot be found. Namely, the Diebold Mariano test for competing forecast performance, the Hering and Genton (2011) modification of this test, as well as the dynamic time warping extension.

The Diebold-Mariano test was proposed in Diebold and Mariano (1995) for obtaining hypothesis tests to compare the forecast accuracy of two competing forecasts against a common verification series. The null hypothesis is that they have the same accuracy. The test statistic is of the form

S = dbar/sqrt(2*pi*se_d(0)/N),

where d is the loss differential, d = e1 - e2 (e1 = loss(x, xhat1) and e2 = loss(x, xhat2)), dbar is its sample mean, and se_d(0) is the standard error for d, which must be estimated, and N is the length of the series investigated. Let V = 2*pi*se_d(0), then V is estimated by

V = sum(gamma(tau)),

where the summation is over tau = -(k - 1) to (k - 1) for temporal lags k, and gamma are the empirical autocovariances.

Hering and Genton (2011) propose a modification that employs fitting a parameteric covariance model in determining the standard error for the test statistic (they also propose a spatial extension, see, e.g., spatMLD from **SpatialVx**).

In either case, asymptotic results suggest that S ~ N(0,1), and the hypothesis test is conducted subsequently.

Discrete time warping can be applied (see examples below) in order to obtain a loss function based on location (in time) and intensity errors similar to the spatial version in Gilleland (2013).

The loss functions supplied by losserr include:

abserr: Absolute error loss, defined by abs((xhat - x)/scale),

sqerr: Square error loss, defined by ((xhat - x)/scale)^2,

simple: Simple loss, defined by (xhat - x)/scale, `power`: Power loss, defined by ((xhat - x)/scale)^p (same as `sqerr` if p=2),

`corrskill`: Correlation skill defined by scale * (x - mean(x)) * (xhat - mean(xhat)),

`dtw`: Discrete time warp loss defined by: d1 + d2, where d1 is the absolute distance (ignoring direction) of warp movement, and d2 is one of the above loss functions (except for `corrskill`) applied to the resulting intensity errors after warping the series.

The `exponential` function takes numeric vector arguments `x` and `y` and estimates the parameters, `c(sigma, theta)`, that optimize

y = sigma^2*exp(-3*x/theta)

Value

`predcomp.test` returns a list object of class c("predcomp.test", "htest") with components:

`call`	the calling string
`method`	character string giving the full name of the method (Diebold-Mariano or Hering-Genton) used.
`fitmodel`	character naming the function used to fit the parametric model to the autocovariances or "none".
`fitmodel.args`	If fitmodel is used, then this will be a list of any arguments passed in for it.
`loss.function`	character string naming the loss function called.
`statistic`	numeric giving the value of the statistic.
`alternative`	character string naming which type of hypothesis test was used (i.e., two-sided or one of the one-sided possibilities).
`p.value`	numeric giving the p-value for the test.
`data.name`	character vector naming the verification and competing forecast series applied to the test.

`losserr` returns a numeric vector of loss values.

`exponentialACV` returns a list object of class "nls" as returned by `nls`.

Author(s)

Eric Gilleland

References

Diebold, F. X. and Mariano, R. S. (1995) Comparing predictive accuracy. *Journal of Business and Economic Statistics*, **13**, 253–263.

Gilleland, E. (2013) Testing competing precipitation forecasts accurately and efficiently: The spatial prediction comparison test. *Mon. Wea. Rev.*, **141** (1), 340–355, http://dx.doi.org/10.1175/MWR-D-12-00155.1.

Gilleland, E. and Roux, G. (2014) A New Approach to Testing Forecast Predictive Accuracy. Accepted to *Meteorol. Appl.* Available at: http://onlinelibrary.wiley.com/doi/10.1002/met.1485/abstract

Hering, A. S. and Genton, M. G. (2011) Comparing spatial predictions. *Technometrics*, **53**, (4), 414–425, doi:10.1198/TECH.2011.10136.

See Also

print.htest, nls, dtw, acf

Examples

```
z0 <- arima.sim(list(order=c(2,0,0), ar=c(0.8,-0.2)), n=1000)
z1 <- c(z0[10:1000], z0[1:9]) + rnorm(1000, sd=0.5)
z2 <- arima.sim(list(order=c(3,0,1), ar=c(0.7,0,-0.1), ma=0.1), n=1000) +
    abs(rnorm(1000, mean=1.25))

test <- predcomp.test(z0, z1, z2)
summary(test)

test2 <- predcomp.test(z0, z1, z2, test = "HG" )
summary(test2)

## Not run:
test2 <- predcomp.test(z0, z1, z2, test = "HG" )
summary(test2)

test2.2 <- predcomp.test(z0, z1, z2, alternative="less")
summary(test2.2)

test3 <- predcomp.test(z0, z1, z2, lossfun.args=list(method="dtw") )
summary(test3)

test3.2 <- predcomp.test(z0, z1, z2, alternative="less",
    lossfun.args=list(method="dtw"), test = "HG" )
summary(test3.2)

test4 <- predcomp.test(z0, z1, z2, lossfun.args = list(method="corrskill"), test = "HG" )
summary(test4)

test5 <- predcomp.test(z0, z1, z2, lossfun.args = list(method="dtw", dtw.interr="sqerr"),
    test = "HG" )
summary(test5)

test5.2 <- predcomp.test(z0, z1, z2, alternative="less",
    lossfun.args=list(method="dtw", dtw.interr="sqerr"), test = "HG" )
summary(test5.2)

## End(Not run)
```

---

prob.frcs.dat *Probablisitic Forecast Dataset.*

Description

This data set is used as an example of data used by the `roc.plot` function. The first column contains a probablisitic forecast for aviation icing. The second column contains a logical variable indicating whether or not icing was observed.

Usage

```
data(prob.frcs.dat)
```

References

PROBABILITY FORECASTS OF IN-FLIGHT ICING CONDITIONS Barbara G. Brown, Ben C. Bernstein, Frank McDonough and Tiffany A. O. Bernstein, 8th Conference on Aviation, Range, and Aerospace Meteorology, Dallas TX, 10-15 January 1999.

`probcont2disc` *Converts continuous probability values into binned discrete probability forecasts.*

Description

Converts continuous probability values into binned discrete probability forecasts. This is useful in calculated Brier type scores for values with continuous probabilities. Each probability is assigned the value of the midpoint.

Usage

```
probcont2disc(x, bins = seq(0,1,0.1) )
```

Arguments

`x`	A vector of probabilities
`bins`	Bins. Defaults to 0 to 1 by 0.1 .

Value

A vector of discrete probabilities. E

Note

This function is used within `brier`.

Author(s)

Matt Pocernich

Examples

```
#  probabilistic/ binary example

set.seed(1)
x <- runif(10) ## simulated probabilities.

probcont2disc(x)
probcont2disc(x, bins = seq(0,1,0.25) )

## probcont2disc(4, bins = seq(0,1,0.3)) ## gets error
```

`qrel.plot` *Quantile Reliability Plot*

Description

The quantile reliability plot gives detailed insights into the performance of quantile forecasts. The conditional observed quantiles are plotted against the discretized quantile forecasts. For calibrated forecasts (i.e., reliability), the points should lie on a diagonal line. The interpretation concerning over or under forecasting of a quantile reliability diagram is analogous to the interpretation of a reliability diagram for probability forecasts of dichotomous events (see for example Wilks (2006), pp. 287 - 290).

Usage

```
qrel.plot(A, ...)
```

Arguments

`A`	A "quantile" class object from `verify`
`...`	optional arguments.

Note

This function is based on `reliabiliy.plot` by Matt Pocernich.

Author(s)

Sabrina Bentzien

References

Bentzien and Friederichs (2013), Decomposition and graphical portrayal of the quantile score, submitted to QJRMS.

See Also

quantileScore, reliability.plot

Examples

```
data(precip.ensemble)

#Observations are in column 3
obs <- precip.ensemble[,3]

#Forecast values of ensemble are in columns 4 to 54
eps <- precip.ensemble[,4:54]

#Quantile forecasts from ensemble
p <- 0.9
qf <- apply(eps,1,quantile,prob=p,type=8)

#generate equally populated binnng intervals
breaks <- quantile(qf,seq(0,1,length.out=11))

qs <- quantileScore(obs,qf,p,breaks)
qrel.plot(qs)
```

---

quantile2disc — *Convert Continuous Forecast Values to Discrete Forecast Values.*

Description

Converts continuous forecast values into discrete forecast values. This is necessary in calculating the quantile score decomposition. Discrete forecasts are defined by the mean value of forecasts within a bin specified by the bin vector (bin boundaries).

Usage

```
quantile2disc(x, bins)
```

Arguments

x	A vector of forecast values
bins	Vector with bin boundaries

Value

new	New vector x (continuous forecast values replaced with disretized forecast values)
mids	Vector of discrete forecast values

Note

This function is used within `quantileScore`.

Author(s)

Sabrina Bentzien

Examples

```
x <- rnorm(100)
bins <- quantile(x,seq(0,1,length.out=11))

newx <- quantile2disc(x,bins)
```

---

quantileScore *Quantile Score*

Description

Calculates verification statistics for quantile forecasts.

Usage

```
quantileScore(obs, pred, p, breaks, ...)
```

Arguments

obs	Vector of observations
pred	Vector of quantile forecasts
p	Probability level of quantile forecasts [0,1].
breaks	Values used to bin the forecasts
...	Optional arguments

Details

This function calculates the quantile score and its decomposition into reliability, resolution, and uncertainty. Note that a careful binning (discretization of forecast values) is necessary to obtain good estimates of reliability and resolution (see Bentzien and Friederichs (2013) for more details).

Value

qs.orig	Quantile score for original data
qs	Quantile score for binned data
qs.baseline	Quantile score for climatology
ss	Quantile skill score
qs.reliability	Reliability part of the quantile score
qs.resolution	Resolution part of the quantile score
qs.uncert	Uncertainty part of the quantile score
y.i	Discretized forecast values – defined as the mean value of forecasts in each bin
obar.i	Conditional observed quantiles
prob.y	Number of forecast-observation pairs in each bin
obar	Climatology – unconditional sample quantile of observations
breaks	Values used to bin the forecasts
check	Difference between original quantile score and quantile score decomposition

Note

This function is used within verify.

Author(s)

Sabrina Bentzien

References

Bentzien, S. and Friederichs, P. (2013) Decomposition and graphical portrayal of the quantile score. Submitted to *QJRMS*.

See Also

check.func, qrel.plot

Examples

```
data(precip.ensemble)

#Observations are in column 3
obs <- precip.ensemble[,3]

#Forecast values of ensemble are in columns 4 to 54
eps <- precip.ensemble[,4:54]

#Quantile forecasts from ensemble
p <- 0.9
qf <- apply(eps,1,quantile,prob=p,type=8)
```

```
#generate equally populated binnng intervals
breaks <- quantile(qf,seq(0,1,length.out=11))

qs <- quantileScore(obs,qf,p,breaks)
## Not run:  qrel.plot(qs)
```

`rcrv` *Reduced centered random variable*

Description

The RCRV provides information on the reliability of an ensemble system in terms of the bias and the dispersion. A perfectly reliable system as no bias and a dispersion equal to 1. The observational error is taken into account

Usage

```
rcrv(obs,epsMean,epsVariance,obsError)
```

Arguments

`obs`	A vector of observations
`epsMean`	A vector of the means of the ensemble
`epsVariance`	A vector of the variances of the ensemble
`obsError`	Observational error

Value

`bias`	The weighted bias between the ensemble and the observation. A value equal to 0 indicates no bias. A positive (negative) value indicates a positive (negative) bias
`disp`	The dispersion of the ensemble. A value equal to 1 indicates no dispersion. A value greater (smaller) then 1 indicates underdispersion (overdispersion)
`y`	Vector of y. Mean of y equals bias and standard deviation of y equals dispersion
`obsError`	Observational error (passed to function)

Author(s)

Ronald Frenette <Ronald.Frenette@ec.gc.ca>

References

G. Candille, C. P. L. Houtekamer, and G. Pellerin: Verification of an Ensemble Prediction System against Observations, *Monthly Weather Review*,**135**, pp. 2688-2699

Examples

```
data(precip.ensemble)
#Observations are in the column
obs<-precip.ensemble[,3]

#Forecast values of ensemble are in the column 4 to 54
eps<-precip.ensemble[,4:54]

#Means and variances of the ensemble
mean<-apply(eps,1,mean)
var<-apply(eps,1,var)

#observation error of 0.5mm
sig0 <- 0.5

rcrv(obs,mean,var,sig0)
```

`reliability.plot` *Reliability Plot*

Description

A reliability plot is a simple form of an attribute diagram that depicts the performance of a probabilistic forecast for a binary event. In this diagram, the forecast probability is plotted against the observed relative frequency. Ideally, this value should be near to each other and so points falling on the 1:1 line are desirable. For added information, if one or two forecasts are being verified, sharpness diagrams are presented in the corners of the plot. Ideally, these histograms should be relatively flat, indicating that each bin of probabilities is use an appropriate amount of times.

Usage

```
   ## Default S3 method:
reliability.plot(x, obar.i, prob.y, titl = NULL, legend.names = NULL, ... )
## S3 method for class 'verify'
reliability.plot(x, ...)
```

Arguments

`x`	Forecast probabilities.(y_i) or a “prob.bin” class object from `verify`.
`obar.i`	Observed relative frequency $\bar{o}_i$.
`prob.y`	Relative frequency of forecasts
`titl`	Title
`legend.names`	Names of each model that will appear in the legend.
`...`	Graphical parameters.

Details

This function works either by entering vectors or on a verify class object.

Note

If a single prob.bin class object is used, a reliability plot along with a sharpness diagram is displayed. If two forecasts are provided in the form of a matrix of predictions, two sharpness diagrams are provided. If more forecasts are provided, the sharpness diagrams are not displayed.

Author(s)

Matt Pocernich

References

Wilks, D. S. (1995) *Statistical Methods in the Atmospheric Sciences* Chapter 7, San Diego: Academic Press.

Examples

```
## Data from Wilks, table 7.3 page 246.
 y.i   <- c(0,0.05, seq(0.1, 1, 0.1))
 obar.i <- c(0.006, 0.019, 0.059, 0.15, 0.277, 0.377, 0.511,
    0.587, 0.723, 0.779, 0.934, 0.933)

 prob.y <- c(0.4112, 0.0671, 0.1833, 0.0986, 0.0616, 0.0366,
    0.0303,  0.0275, 0.245, 0.022, 0.017, 0.203)

 obar <- 0.162

reliability.plot(y.i, obar.i, prob.y, titl = "Test 1", legend.names =
c("Model A") )


## Function will work with a ``prob.bin'' class object as well.
## Note this is a very bad forecast.
obs<- round(runif(100))
pred<- runif(100)

A<- verify(obs, pred, frcst.type = "prob", obs.type = "binary")

reliability.plot(A, titl = "Alternative plot")
```

`roc.area` *Area under curve (AUC) calculation for Response Operating Characteristic curve.*

Description

This function calculates the area underneath a ROC curve following the process outlined in Mason and Graham (2002). The p-value produced is related to the Mann-Whitney U statistics. The p-value is calculated using the wilcox.test function which automatically handles ties and makes approximations for large values.

The p-value addresses the null hypothesis $H_o:$ The area under the ROC curve is 0.5 i.e. the forecast has no skill.

Usage

```
roc.area(obs, pred)
```

Arguments

`obs`	A binary observation (coded {0, 1 }).
`pred`	A probability prediction on the interval [0,1].

Value

`A`	Area under ROC curve, adjusted for ties in forecasts, if present
`n.total`	Total number of records
`n.events`	Number of events
`n.noevents`	Number of non-events
`p.value`	Unadjusted p-value

Note

This function is used internally in the `roc.plot` command to calculate areas.

Author(s)

Matt Pocernich

References

Mason, S. J. and Graham, N. E. (2002) Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation, *Q. J. R. Meteorol. Soc.* **128**, 2145–2166.

See Also

roc.plot, verify, wilcox.test

Examples

```
# Data used from Mason and Graham (2002).
a<- c(1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990,
 1991, 1992, 1993, 1994, 1995)
b<- c(0,0,0,1,1,1,0,1,1,0,0,0,0,1,1)
c<- c(.8, .8, 0, 1,1,.6, .4, .8, 0, 0, .2, 0, 0, 1,1)
d<- c(.928,.576, .008, .944, .832, .816, .136, .584, .032, .016, .28, .024, 0, .984, .952)

A<- data.frame(a,b,c, d)
names(A)<- c("year", "event", "p1", "p2")

## for model with ties
roc.area(A$event, A$p1)

## for model without ties
roc.area(A$event, A$p2)
```

---

roc.plot *Relative operating characteristic curve.*

Description

This function creates Receiver Operating Characteristic (ROC) plots for one or more models. A ROC curve plots the false alarm rate against the hit rate for a probablistic forecast for a range of thresholds. The area under the curve is viewed as a measure of a forecast's accuracy. A measure of 1 would indicate a perfect model. A measure of 0.5 would indicate a random forecast.

Usage

```
    ## Default S3 method:
roc.plot(x, pred, thresholds = NULL, binormal =
FALSE,   legend = FALSE, leg.text = NULL,  plot = "emp", CI = FALSE,
n.boot = 1000, alpha = 0.05, tck = 0.01, plot.thres = seq(0.1,
0.9, 0.1), show.thres = TRUE, main = "ROC Curve", xlab = "False Alarm Rate",
ylab = "Hit Rate", extra = FALSE,  ...)
## S3 method for class 'prob.bin'
roc.plot(x, ...)
```

Arguments

x A binary observation (coded {0, 1 }) or a verification object.

pred	A probability prediction on the interval [0,1]. If multiple models are compared, this may be a matrix where each column represents a different prediction.
thresholds	Thresholds may be provided. These thresholds will be used to calculate the hit rate (h) and false alarm rate (f). If thresholds is NULL, all unique thresholds are used as a threshold. Alternatively, if the number of bins is specified, thresholds will be calculated using the specified numbers of quantiles.
binormal	If TRUE, in addition to the empirical ROC curve, the binormal ROC curve will be calculated. To get a plot draw, plot must be either “binorm” or “both”.
legend	Binomial. Defaults to FALSE indicating whether a legend should be displayed.
leg.text	Character vector for legend. If NULL, models are labeled “Model A", “Model B",...
plot	Either “emp” (default), “binorm” or “both” to determine which plot is shown. If set to NULL, a plot is not created
CI	Confidence Intervals. Calculated by bootstrapping the observations and prediction, then calculating PODy and PODn values.
n.boot	Number of bootstrap samples.
alpha	Confidence interval. By default = 0.05
tck	Tick width on confidence interval whiskers.
plot.thres	By default, displays the threshold levels on the ROC diagrams. To surpress these values, set it equal to NULL. If confidence intervals (CI) is set to TRUE, levels specified here will determine where confidence interval boxes are placed.
show.thres	Show thresholds for points indicated by plot.thres. Defaults to TRUE.
main	Title for plot.
xlab, ylab	Plot axes labels. Defaults to “Hit Rate” and “False Alarm Rate”, for the y and x axes respectively.
extra	Extra text describing binormal and empirical lines.
...	Additional plotting options.

Value

If assigned to an object, the following values are reported.

plot.data	The data used to generate the ROC plots. This is a array. Column headers are thresholds, empirical hit and false alarm rates, and binormal hit and false alarm rates. Each model is depicted on an array indexed by the third dimension.
roc.vol	The areas under the ROC curves. By default,this is printed on the plots. Areas and p-values are calculated with and without adjustments for ties along with the p-value for the area. These values are calculated using roc.area. The fifth column contains the area under the binormal curve, if binormal is selected.
A.boot	If confidence intervals are calculated, the area under the ROC curve are returned.

Note

Other packages in R provide functions to create ROC diagrams and different diagnostics. The **ROCR** package provides excellent functions to generate ROC diagrams with lines coded by threshold. Large datasets are handled by a sampling routine and the user may plot a number of threshold dependent, contingency table scores. Arguably, this is a superior package with respect to ROC plotting.

There is not a minimum size required to create confidence limits or show thresholds. When there are few data points, it is possilbe to make some pretty unattractive graphs.

The roc.plot method can be used to summarize a "verify, prob.bin" class object created with the verify command. It is appropriate to use the roc plot for forecast which are not probabilities, but rather forecasts made on a continuous scale. The roc plot function can be used to summarize such forecasts but it is not possible to use the verify function to summarize such forecasts. An example is shown below.

Author(s)

Matt Pocernich

References

Mason, I. (1982) “A model for assessment of weather forecasts,” *Aust. Met. Mag* **30** (1982) 291-303.

Mason, S.J. and N.E. Graham. (2002) “Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation, ” *Q. J. R. Meteorol. Soc.* **128** pp. 2145-2166.

Swets, John A. (1996) *Signal Detection Theory and ROC Analysis in Psychology and Diagnostics*, Lawrence Erlbaum Associates, Inc.

See Also

pop and lines.roc

Examples

```
# Data from Mason and Graham article.

a<- c(0,0,0,1,1,1,0,1,1,0,0,0,0,1,1)
b<- c(.8, .8, 0, 1,1,.6, .4, .8, 0, 0, .2, 0, 0, 1,1)
c<- c(.928,.576, .008, .944, .832, .816, .136, .584, .032, .016, .28, .024, 0, .984, .952)

A<- data.frame(a,b,c)
names(A)<- c("event", "p1", "p2")

## for model with ties
roc.plot(A$event, A$p1)

## for model without ties
roc.plot(A$event, A$p2)
```

```
### show binormal curve fit.

roc.plot(A$event, A$p2, binormal = TRUE)
## Not run:
# icing forecast

data(prob.frcs.dat)
A <- verify(prob.frcs.dat$obs, prob.frcs.dat$frcst/100)
roc.plot(A, main = "AWG Forecast")


# plotting a ``prob.bin'' class object.
obs<- round(runif(100))
pred<- runif(100)

A<- verify(obs, pred, frcst.type = "prob", obs.type = "binary")

roc.plot(A, main = "Test 1", binormal = TRUE, plot = "both")

## show confidence intervals.  MAY BE SLOW
roc.plot(A, threshold = seq(0.1,0.9, 0.1), main = "Test 1", CI = TRUE,
alpha = 0.1)

###   example from forecast verification website.
data(pop)
d <- pop.convert() ## internal function used to make binary observations for the pop figure.
### note the use of bins = FALSE !!
 mod24 <- verify(d$obs_norain, d$p24_norain, bins = FALSE)

 mod48 <- verify(d$obs_norain, d$p48_norain, bins = FALSE)

roc.plot(mod24, plot.thres = NULL)
lines.roc(mod48, col = 2, lwd = 2)
leg.txt <- c("24 hour forecast", "48 hour forecast")
legend( 0.6, 0.4, leg.txt, col = c(1,2), lwd = 2)

## End(Not run)
```

---

rps *Ranked Probability Score*

Description

Calculates the ranked probability score (rps) and ranked probability skill score (rpss) for probabilistic forecasts of ordered events.

Usage

```
rps(obs, pred, baseline=NULL)
```

Arguments

`obs`	A vector of observed outcomes. These values correspond to columns of prediction probabilities.
`pred`	A matrix of probabilities for each outcome occurring. Each column represents a category of prediction.
`baseline`	If NULL (default) the probability based on the sample data of each event to occur. Alternatively, a vector the same length of the as the number categories can be entered.

Value

`rps`	Ranked probability scores
`rpss`	Ranked probability skill score. Uses baseline or sample climatology as a references score.
`rps.clim`	Ranked probability score for baseline forecast.

Note

Perhaps the format of the data is best understood in the context of an example. Consider a probability of precipitation forecast of "none", "light" or "heavy". This could be [0.5, 0.3, 0.2]. If heavy rain occurred, the observed value would be 3, indicating event summarized in the third column occurred.

The RPS value is scaled to a [0,1] interval by dividing by (number of categories -1 . There is a discrepancy in the way this is explained in Wilks (2005) and the WWRF web page.

Author(s)

Matt Pocernich

References

WWRP/WGNE Joint Working Group on Verification - Forecast Verification - Issues, Methods and FAQ http://www.cawcr.gov.au/projects/verification/verif_web_page.html#RPS

Wilks, D. S. (2005) *Statistical Methods in the Atmospheric Sciences* Chapter 7, San Diego: Academic Press.

See Also

`crps`

Examples

```
###  Example from Wilks, note without a baseline and only one
### forecast, the rpss and ss are not too meaningfull.


rps( obs = c(1), pred = matrix(c(0.2, 0.5, 0.3), nrow = 1))
```

`table.stats` *Verification statistics for a 2 by 2 Contingency Table*

Description

Provides a variety of statistics for a data summarized in a 2 by 2 contingency table.

Usage

```
table.stats(obs, pred, fudge = 0.01, silent = FALSE)
```

Arguments

`obs`	Either a vector of contingency table counts, a vector of binary observations, or a 2 by 2 matrix in the form of a contingency table. (See note below.)
`pred`	Either null or a vector of binary forecasts.
`fudge`	A numeric fudge factor to be added to each cell of the contingency table in order to avoid division by zero.
`silent`	Should warning statements be surpressed.

Value

`tab.out`	Contingency table
`TS`	Threat score a.k.a. Critical success index (CSI)
`TS.se`	Standard Error for TS
`POD`	Hit Rate aka probability of detection
`POD.se`	Standard Error for POD
`M`	Miss rate
`F`	False Alarm RATE
`F.se`	Standard Error for F
`FAR`	False Alarm RATIO
`FAR.se`	Standard Error for FAR
`HSS`	Heidke Skill Score
`HSS.se`	Standard Error for HSS
`PSS`	Peirce Skill Score
`PSS.se`	Standard Error for PSS
`KSS`	Kuiper's Skill Score
`PC`	Percent correct - events along the diagonal.
`PC.se`	Standard Error for PC
`BIAS`	Bias
`ETS`	Equitable Threat Score

`ETS.se`	Standard Error for ETS
`theta`	Odds Ratio
`log.theta`	Log Odds Ratio
`LOR.se`	Standard Error for Log Odds Ratio
`n.h`	Degrees of freedom for log.theta
`orss`	Odds ratio skill score, aka Yules's Q
`ORSS.se`	Standard Error for Odds ratio skill score
`eds`	Extreme Dependency Score
`esd.se`	Standard Error for EDS
`seds`	Symmetric Extreme Dependency Score
`seds.se`	Standard Error for Symmetric Extreme Dependency Score
`EDI`	Extreme Dependency Index
`EDI.se`	Standard Error for EDI
`SEDI`	Symmetric EDI
`SEDI.se`	Standard Error for SEDI

Note

Initially, table.stats was an internal function used by verify for binary events and multi.cont for categorical events. But occassionally, it is nice to use it directly.

Author(s)

Matt Pocernich

References

Jolliffe, I.T. and D.B. Stephenson (2003). Forecast verification: a practitioner's guide in atmospheric science. John Wiley and Sons. See chapter 3 concerning categorical events.

Stephenson, D.B. (2000). "Use of 'Odds Ratio for Diagnosing Forecast Skill." *Weather and Forecasting* **15** 221-232.

Hogan, R.J., O'Connor E.J. and Illingworth, 2009. "Verification of cloud-fraction forecasts." *Q.J.R. Meteorol. Soc.* **135**, 1494-1511.

See Also

`verify` and `multi.cont`

Examples

```
DAT<- matrix(c(28, 23, 72, 2680 ), ncol = 2) ## Finley
table.stats(DAT)
```

`table.stats.boot` *Percentile bootstrap for 2 by 2 table*

Description

Performs a bootstrap on data from a 2 by 2 contingency table returning verification statistics. Potentially useful in creating error bars for performance diagrams.

Usage

```
table.stats.boot(CT, R = 100, alpha = 0.05, fudge = 0.01)
```

Arguments

`CT`	Two by two contingency table. Columns summarize observed values. Rows summarize forecasted values.
`R`	Number of resamples
`alpha`	Confidence intervals.
`fudge`	A numeric fudge factor to be added to each cell of the contingency table in order to avoid division by zero.

Value

2 row matrix with upper and lower intervals for bias, pod, far, ets.

Author(s)

Matt Pocernich

See Also

`table.stats`

Examples

```
### example from Roebber.
RB1 <- matrix(c(95, 55, 42, 141), ncol = 2)
table.stats.boot(RB1, R = 1000   )
```

value *Forecast Value Function*

Description

Calculates the economic value of a forecast based on a cost/loss ratio.

Usage

```
value(obs, pred= NULL, baseline = NULL, cl = seq(0.05, 0.95, 0.05),
    plot = TRUE, all = FALSE, thresholds = seq(0.05, 0.95, 0.05),
    ylim = c(-0.05, 1), xlim = c(0,1), ...)
```

Arguments

obs	A vector of binary observations or a contingency table summary of values in the form c(n11, n01, n10, n00) where in nab a = obs, b = forecast.
pred	A vector of probabilistic predictions.
baseline	Baseline or naive forecast. Typically climatology.
cl	Cost loss ratio. The relative value of being unprepared and taking a loss to that of un-necessarily preparing. For example, cl = 0.1 indicates it would cost \$ 1 to prevent a \$10 loss. This defaults to the sequence 0.05 to 0.95 by 0.05.
plot	Should a plot be created? Default is TRUE
all	In the case of probabilistic forecasts, should value curves for each thresholds be displayed.
thresholds	Thresholds considered for a probabilistic forecast.
ylim, xlim	Plotting options.
...	Options to be passed into the plotting function.

Value

If assigned to an object, the following values are reported.

vmax	Maximum value
V	Vector of values for each cl value
F	Conditional false alarm rate.
H	Conditional hit rate
cl	Vector of cost loss ratios.
s	Base rate

Author(s)

Matt Pocernich

References

Jolliffe, Ian and David B. Stephensen (2003) *Forecast Verification: A Practioner's Guide in Atmospheric Science*, Chapter 8. Wiley

Examples

```
## value as a contingency table
## Finley tornado data
obs<- c(28, 72, 23, 2680)
value(obs)
aa <- value(obs)
aa$Vmax # max value

## probabilistic forecast example
 obs  <- round(runif(100) )
 pred <-  runif(100)

value(obs, pred, main = "Sample Plot",
             thresholds = seq(0.02, 0.98, 0.02) )
##########
data(pop)
d <- pop.convert()

value(obs = d$obs_rain, pred = d$p24_rain, all = TRUE)
```

`verify` *Verification function*

Description

Based on the type of inputs, this function calculates a range of verification statistics and skill scores. Additionally, it creates a verify class object that can be used in further analysis or with other methods such as plot and summary.

Usage

```
verify(obs, pred, p = NULL, baseline = NULL,
    frcst.type = "prob", obs.type = "binary",
    thresholds = seq(0,1,0.1), show = TRUE, bins = TRUE,
    fudge = 0.01, ...)
```

Arguments

`obs`	The values with which the verifications are verified. May be a vector of length 4 if the forecast and predictions are binary data summarized in a contingency table. In this case, the value are entered in the order of c(n11, n01, n10, n00). If obs is a matrix, it is assumed to be a contingency table with observed values summarized in the columns and forecasted values summarized in the rows.
`pred`	Prediction of event. The prediction may be in the form of the a point prediction or the probability of a forecast. Let pred = NULL if obs is a contingency table.
`p`	the probability level of the quantile forecast, any value between 0 and 1.
`baseline`	In meteorology, climatology is the baseline that represents the no-skill forecast. In other fields this field would differ. This field is used to calculate certain skill scores. If left NULL, these statistics are calculated using sample climatology. If this is not NULL, the mean of these values is used as the baseline forecast. This interpretation is not appropriate for all applications. For example, if a baseline forecast is different for each forecast this will not work appropriately.
`frcst.type`	Forecast type. One of "prob", "binary", "norm.dist", "cat" or "cont", or "quantile". Defaults to "prob". "norm.dist" is used when the forecast is in the form of a normal distribution. See crps for more details.
`obs.type`	Observation type. Either "binary", "cat" or "cont". Defaults to "binary"
`thresholds`	Thresholds to be considered for point forecasts of continuous events.
`show`	Binary; if TRUE (the default), print warning message
`bins`	Binary; if TRUE (default), the probabilistic forecasts are placed in bins defined by the sequence defined in threshold and assigned the midpoint value.
`fudge`	A numeric fudge factor to be added to each cell of the contingency table in order to avoid division by zero.
`...`	Additional options.

Details

See Wilks (2006) and the WMO Joint WWRP/WGNE Working Group web site on verification for more details about these verification statistics. See Stephenson et al. (2008) and Ferro and Stephenson (2011) for more on the extreme dependence scores and indices. For information on confidence intervals for these scores, see Gilleland (2010).

Value

An object of the verify class. Depending on the type of data used, the following information may be returned. The following notation is used to describe which values are produced for which type of forecast/observations. (BB = binary/binary, PB = probablistic/binary, CC = continuous/continuous, CTCT = categorical/categorical)

`BS`	Brier Score (PB)
`BSS`	Brier Skill Score(PB)
`SS`	Skill Score (BB)
`hit.rate`	Hit rate, aka PODy, \$h\$ (PB, CTCT)

`false.alarm.rate`	False alarm rate, PODn, f (PB, CTCT)
`TS`	Threat Score or Critical Success Index (CSI)(BB, CTCT)
`ETS`	Equitable Threat Score (BB, CTCT)
`BIAS`	Bias (BB, CTCT)
`PC`	Percent correct or hit rate (BB, CTCT)
`Cont.Table`	Contingency Table (BB)
`HSS`	Heidke Skill Score(BB, CTCT)
`KSS`	Kuniper Skill Score (BB)
`PSS`	Pierce Skill Score (CTCT)
`GS`	Gerrity Score (CTCT)
`ME`	Mean error (CC)
`MSE`	Mean-squared error (CC)
`MAE`	Mean absolute error (CC)
`theta`	Odds Ratio (BB)
`log.theta`	Log Odds Ratio
`n.h`	Degrees of freedom for log.theta (BB)
`orss`	Odds ratio skill score, aka Yules's Q (BB)
`eds`	Extreme Dependency Score (BB)
`eds.se`	Standard Error for Extreme Dependence Score (BB)
`seds`	Symmetric Extreme Dependency Score (BB)
`seds.se`	Standard Error for Symmetric Extreme Dependency Score (BB)
`EDI`	Extremal Dependence Index (BB)
`EDI.se`	Standard Error for Extremal Dependence Index (BB)
`SEDI`	Symmetric Extremal Dependence Index (BB)
`SEDI.se`	Standard Error for Symmetric Extremal Dependence Index (BB)

Note

There are other packages in R and Bioconductor which are usefull for verification tasks. This includes the **ROCR**, **ROC**, package and the **limma** package (in the Bioconductor repository.) Written by people in different fields, each provides tools for verification from different perspectives.

For the categorical forecast and verification, the Gerrity score only makes sense for forecast that have order, or are basically ordinal. It is assumed that the forecasts are listed in order. For example, if the rows of a contigency table were summarized as "medium, low, high", the Gerrity score will be incorrectly summarized.

As of version 1.37, the intensity scale (IS) verification funcitons have been removed from this package. Please use **SpatialVx** for this functionality.

Author(s)

Matt Pocernich

References

Ferro, C. A. T. and D. B. Stephenson, 2011. Extremal dependence indices: Improved verification measures for deterministic forecasts of rare binary events. *Wea. Forecasting*, **26**, 699 - 713.

Gilleland, E., 2010. Confidence intervals for forecast verification. NCAR Technical Note NCAR/TN-479+STR, 71pp. Available at: http://nldr.library.ucar.edu/collections/technotes/asset-000-000-000-846.pdf

Stephenson, D. B., B. Casati, C. A. T. Ferro, and C. A. Wilson, 2008. The extreme dependency score: A non-vanishing measure for forecasts of rare events. *Meteor. Appl.*, **15**, 41 - 50.

Wilks, D. S., 2006. *Statistical Methods in the Atmospheric Sciences* , San Diego: Academic Press., 627 pp. (2nd Editiion).

WMO Joint WWRP/WGNE Working Group on Verification Website

http://www.cawcr.gov.au/projects/verification/

See Also

`table.stats`

Examples

```
# binary/binary example
obs<- round(runif(100))
pred<- round(runif(100))

# binary/binary example
# Finley tornado data.

obs<- c(28, 72, 23, 2680)
A<- verify(obs, pred = NULL, frcst.type = "binary", obs.type = "binary")

summary(A)

# categorical/categorical example
# creates a simulated 5 category forecast and observation.
obs <- round(runif(100, 1,5) )
pred <- round(runif(100, 1,5) )

A<- verify(obs, pred, frcst.type = "cat", obs.type = "cat" )
summary(A)

#  probabilistic/ binary example

pred<- runif(100)
A<- verify(obs, pred, frcst.type = "prob", obs.type = "binary")
summary(A)

# continuous/ continuous example
obs<- rnorm(100)
pred<- rnorm(100)
baseline <- rnorm(100, sd = 0.5)
```

```
A<- verify(obs, pred, baseline = baseline,  frcst.type = "cont", obs.type = "cont")
summary(A)
```

Index