Package ‘scoring’

February 12, 2018

Title Proper Scoring Rules

Version 0.6

Date 2018-02-11

Author Ed Merkle

Maintainer Ed Merkle <merklee@missouri.edu>

Description Evaluating probabilistic forecasts via proper scoring rules. scoring implements the beta, power, and pseudospherical families of proper scoring rules, along with ordered versions of the latter two families. Included among these families are popular rules like the Brier (quadratic) score, logarithmic score, and spherical score. For two-alternative forecasts, also includes functionality for plotting scores that one would obtain under specific scoring rules.

Suggests testthat

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-12 04:05:24 UTC

R topics documented:

`brierscore` *Calculate Brier Scores And Decompositions*

Description

Calculate Brier scores, average Brier scores by a grouping variable, and Brier score decompositions for two-alternative forecasts.

Usage

```
brierscore(object, data, group = NULL, decomp = FALSE, bounds = NULL,
           reverse = FALSE, wt = NULL, decompControl = list())
```

Arguments

`object`	an object of class "formula", of the form `outcome ~ forecast`. See `calcscore()` documentation for further details.
`data`	an optional data frame or list containing the variables in the formula. If not found in `data`, the variables are taken from the environment from which `calcscore` is called.
`group`	the name of a grouping variable within `data`, which is used to calculate average Brier score by group.
`decomp`	if `TRUE`, Brier score decompositions are calculated.
`bounds`	a vector of length 2 corresponding to the desired minimum and maximum Brier score, respectively.
`reverse`	if `FALSE` (default), smaller scores imply better forecasts. If `TRUE`, larger scores imply better forecasts.
`wt`	a vector of weights for computing a weighted Brier score. If `NULL`, the weights are set to 1/n, where n is the number of forecasts (this corresponds to a simple average Brier score).
`decompControl`	a list of additional settings for the Brier decomposition. See options below.

Details

If `decomp=TRUE` or `group` is supplied, the function returns a list (see value section). Otherwise, the function returns a numeric vector containing the Brier score associated with each forecast.

Some `decompControl` arguments are specifically designed for forecasting tournaments and may not be useful in other situations. Possible arguments for `decompControl` include:

- **wt** A vector of weights, for performing a weighted Brier decomposition (could also use the simple `wt` argument).
- **qid** A vector of question ids, for use with the `qtype` argument.
- **bin** If `TRUE` (default), forecasts are binned prior to decomposition. If `FALSE`, the original forecasts are maintained.

qtype A data frame with columns `qid`, `ord`, `squo`. For each unique question id in the `qid` argument above, this describes whether or not the question is ordinal (1=yes,0=no) and whether or not the question has a "status quo" interpretation (1=yes,0=no).

scale Should Brier components be rescaled, such that 1 is always best and 0 is always worst? Defaults to `FALSE`.

roundto To what value should forecasts be rounded (necessary for Murphy decomposition)? Defaults to .1, meaning that forecasts are rounded to the nearest .1.

binstyle Method for ensuring that each forecast sums to 1. If equal to 1 (default), the smallest forecast is one minus the sum of the other forecasts. If equal to 2, the forecast furthest from its rounded value is one minus the sum of other forecasts.

resamples Desired number of Brier resamples (useful for questions with inconsistent alternatives). Defaults to 0; see Merkle & Hartman reference for more detail.

Value

Depending on input arguments, `brierscore` may return an object of class `numeric` containing raw Brier scores. It may also return a list containing the objects below.

`rawscores`	an object of class `numeric` containing raw Brier scores for each forecast.
`brieravg`	an object of class `numeric` containing average Brier scores for each unique value of `group`. If `wt` was supplied, this is a weighted sum. Otherwise, it is a simple average (equal weights summing to 1).
`decomp`	an object of class `matrix` containing Brier score decompositions and mean Brier scores for each unique value of `group`.

Author(s)

Ed Merkle

References

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review, 78*, 1-3.

Merkle, E. C. & Hartman, R. (2018). Weighted Brier score decompositions for topically heterogenous forecasting tournaments. *Working paper*.

Murphy, A. H. (1973). A new vector partition of the probability score. *Journal of Applied Meteorology, 12*, 595-600.

Yates, J. F. (1982). External correspondence: Decompositions of the mean probability score. *Organizational Behavior and Human Performance, 30*, 132-156.

Young, R. M. B. (2010). Decomposition of the Brier score for weighted forecast-verification pairs. *Quarterly Journal of the Royal Meteorological Society, 136*, 1364-1370.

See Also

`calcscore`

Examples

```
data("WorldEvents")
## Raw Brier scores
brier1 <- brierscore(answer ~ forecast, data=WorldEvents)
## Raw Brier scores plus group means and decompositions
brier2 <- brierscore(answer ~ forecast, data=WorldEvents,
                     group="forecaster", decomp=TRUE)
## Obtain Brier scores via calcscore
brier3 <- calcscore(answer ~ forecast, data=WorldEvents,
                    param=2, fam="pow")
all.equal(brier1, brier3)
```

---

calcscore *Calculate Scores Under A Specific Rule*

Description

Given parameters of a scoring rule family, calculate scores for probabilistic forecasts and associated outcomes.

Usage

```
## S3 method for class 'formula'
calcscore(object, fam="pow", param, data, bounds=NULL,
          reverse=FALSE, ordered=FALSE, ...)

## Default S3 method:
calcscore(object, outcome, fam="pow",
          param=c(2,rep(1/max(2,NCOL(forecast)),max(2,NCOL(forecast)))),
          bounds=NULL, reverse=FALSE, ordered=FALSE, ...)
```

Arguments

`object`	an object of class "formula", of the form `outcome ~ forecast` (see details). Alternatively, a matrix of forecasts, with observations in rows and forecast alternatives in columns. For two-alternative forecasts, this can be a vector reflecting forecasts for one alternative.
`outcome`	a vector of outcomes (used if `object` is a matrix). For each row of the `forecast` matrix, `outcome` should contain an entry reflecting the column number associated with the event that occurred.
`fam`	scoring rule family. `pow` (default) is the power family, `beta` is the beta family, `sph` is the pseudospherical family.
`param`	for family `beta`, a numeric vector of length 2 containing the scoring rule family parameters. For other families, a numeric vector containing first the family parameter gamma and optionally `NCOL(forecast)` baseline parameters (see details). Alternatively, a matrix may be supplied containing unique family parameters for each forecast row.

`data`	an optional data frame or list containing the variables in the formula. If not found in `data`, the variables are taken from the environment from which `calcscore` is called.
`bounds`	a vector of length 2 corresponding to the desired minimum value and maximum value of the scoring rule, respectively. Entries of `NA` imply that the minimum and/or maximum bound will not be modified from the natural, family-implied bounds.
`reverse`	if `FALSE` (default), smaller scores imply better forecasts. If `TRUE`, larger scores imply better forecasts.
`ordered`	if `FALSE` (default), forecast alternatives have no ordering. If `TRUE`, forecast alternatives have the ordering implied by `forecast`. The resulting scoring rule is sensitive to this ordering (see details).
`...`	Additional arguments.

Details

The formula is of the form `outcome ~ forecast`, where `forecast` describes the column(s) containing forecasts associated with the possible outcomes. Multiple columns are separated by `+`. `outcome` is always a vector describing the outcome associated with each forecast. It should be coded 1, 2, ..., reflecting the column associated with the outcome (see examples).

For events with only two alternatives, one can take a shortcut and supply only forecasts associated with a single outcome (if baseline parameters are specified for families `pow` and `sph`, the parameter for only that outcome should also be supplied). In this case, the `outcome` vector should contain zeros and ones, where 'one' means that the forecasted alternative occurred.

If `ordered=TRUE`, an "ordered" scoring rule is obtained using the strategy proposed by Jose, Nau, & Winkler (2009). These ordered rules are only useful when the number of forecasted alternatives is greater than two (i.e., when one uses family `pow` or `sph`).

If baseline parameters are not supplied for families `pow` or `sph`, then the parameters are taken to be equal across all alternatives (though the natural scaling of the scoring rule differs depending on whether or not one explicitly supplies equal baseline parameters).

If desired, a unique scoring rule can be applied to each row of the forecast matrix: the `param` argument can be supplied as a matrix.

When the `bounds` argument is supplied, the code attempts to scale the scores so that the maximum score is `bounds[2]` and the minimum score is `bounds[1]`. This scaling cannot be accomplished when the scoring rule allows scores of infinity (the log score is the most common case here). If `reverse=TRUE`, the bounds are applied after the reversal (so that the supplied lower bound reflects the worst score and upper bound reflects the best score).

Value

`calcscore` returns a numeric vector that has length equal to `length(outcome)`, containing scores under the selected scoring rule.

Note

The beta family was originally proposed by Buja et al.\ (2005); the power and pseudospherical families with baseline are described by Jose et al.\ (2009). A discussion of choosing specific rules from these families is provided by Merkle and Steyvers (2013).

Some notable special cases of these families are:

Beta family: Log score when parameters are (0,0); Brier score when parameters are (1,1).

Power family with baseline parameters all equal (to 1/(number of alternatives)): The family approaches the log score as gamma goes to 1 (but the family is undefined for gamma=1). The Brier score is obtained for gamma=2.

Pseudospherical family with baseline parameters all equal: The family approaches the log score as gamma goes to 1 (but the family is undefined for gamma=1). The spherical score is obtained for gamma=2.

Author(s)

Ed Merkle

References

Buja, A., Stuetzle, W., & Shen, Y. (2005). Loss functions for binary class probability estimation and classification: Structure and applications. (Obtained from http://stat.wharton.upenn.edu/~buja/PAPERS/)

Jose, V. R. R., Nau, R. F., & Winkler, R. L. (2008). Scoring rules, generalized entropy, and utility maximization. *Operations Research, 56*, 1146–1157.

Jose, V. R. R., Nau, R. F., & Winkler, R. L. (2009). Sensitivity to distance and baseline distributions in forecast evaluation. *Management Science, 55*, 582–590.

Merkle, E. C. & Steyvers, M. (in press). Choosing a strictly proper scoring rule. *Decision Analysis*.

See Also

plotscore

Examples

```
## Brier scores for two alternatives, with bounds of 0 and 1
data("WorldEvents")
scores <- calcscore(answer ~ forecast, fam="beta",
                    param=c(1,1), data=WorldEvents,
                    bounds=c(0,1))

## Calculate Brier scores manually
scores.man <- with(WorldEvents, (forecast - answer)^2)

## Comparison
all.equal(scores, scores.man)

## Average Brier score for each forecaster
with(WorldEvents, tapply(scores, forecaster, mean))
```

```
## Brier scores for 3 alternatives, with bounds of 0 and 1
data("WeatherProbs")
scores2 <- calcscore(tcat ~ tblw + tnrm + tabv, fam="pow",
                     param=2, data=WeatherProbs,
                     bounds=c(0,1))

## Spherical scores for 3 alternatives, reversed so 0 is worst and
## 1 is best
scores3 <- calcscore(tcat ~ tblw + tnrm + tabv, fam="sph",
                     param=2, data=WeatherProbs,
                     bounds=c(0,1), reverse=TRUE)

## Replicate Jose, Nau, & Winkler, 2009, Figure 1
r2 <- seq(0, .6, .05)
r <- cbind(.4, r2, .6 - r2)
j <- rep(1, length(r2))

## Panel 1
quad <- calcscore(j ~ r, fam="pow", param=2, bounds=c(-1,1), reverse=TRUE)
quadbase <- calcscore(j ~ r, fam="pow", param=c(2,.3,.6,.1), reverse=TRUE)
rankquad <- calcscore(j ~ r, fam="pow", param=2, ordered=TRUE, reverse=TRUE)
rankquadbase <- calcscore(j ~ r, fam="pow", param=c(2,.3,.6,.1), ordered=TRUE,
                          reverse=TRUE)
plot(r2, quad, ylim=c(-2,1), type="l", ylab="Quadratic scores")
lines(r2, quadbase, lty=2)
lines(r2, rankquad, type="o", pch=22)
lines(r2, rankquadbase, type="o", pch=2)

## Panel 2
sph <- calcscore(j ~ r, fam="sph", param=2, reverse=TRUE, bounds=c(-1.75,1))
sphbase <- calcscore(j ~ r, fam="sph", param=c(2,.3,.6,.1), reverse=TRUE)
ranksph <- calcscore(j ~ r, fam="sph", param=2, ordered=TRUE, reverse=TRUE)
ranksphbase <- calcscore(j ~ r, fam="sph", param=c(2,.3,.6,.1), ordered=TRUE,
                         reverse=TRUE)
plot(r2, sph, ylim=c(-1,.6), type="l", ylab="Spherical scores")
lines(r2, sphbase, lty=2)
lines(r2, ranksph, type="o", pch=22)
lines(r2, ranksphbase, type="o", pch=2)

## Panel 3
lg <- calcscore(j ~ r, fam="pow", param=1.001, reverse=TRUE)
lgbase <- calcscore(j ~ r, fam="pow", param=c(1.001,.3,.6,.1), reverse=TRUE)
ranklg <- calcscore(j ~ r, fam="pow", param=1.001, ordered=TRUE, reverse=TRUE)
ranklgbase <- calcscore(j ~ r, fam="pow", param=c(1.001,.3,.6,.1),
                        ordered=TRUE, reverse=TRUE)
plot(r2, lg, ylim=c(-2,1), type="l", ylab="Log scores")
lines(r2, lgbase, lty=2)
lines(r2, ranklg, type="o", pch=22)
lines(r2, ranklgbase, type="o", pch=2)
```

logscore — *Calculate Logarithmic Scores*

Description

Calculate logarithmic scores and average logarithmic scores by a grouping variable.

Usage

```
logscore(object, data, group = NULL, reverse = FALSE)
```

Arguments

`object`	an object of class "formula", of the form `outcome ~ forecast`. See `calcscore()` documentation for further details.
`data`	an optional data frame or list containing the variables in the formula. If not found in `data`, the variables are taken from the environment from which `calcscore` is called.
`group`	the name of a grouping variable within `data`, which is used to calculate average log score by group.
`reverse`	if `FALSE` (default), smaller scores imply better forecasts. If `TRUE`, larger scores imply better forecasts.

Details

If `group` is supplied, the function returns a list (see value section). Otherwise, the function returns a numeric vector containing the log score associated with each forecast.

The argument bounds is not available because the upper bound of the logarithmic score is infinity. If one wants a bounded rule that approximates the logarithmic rule, try using `calcscore()` with `fam="pow"` and `param=1.001`.

Value

Depending on input arguments, `logscore` may return an object of class `numeric` containing raw logarithmic scores. It may also return a list containing the objects below.

`rawscores`	an object of class `numeric` containing raw log scores for each forecast.
`mnlog`	an object of class `numeric` containing mean log scores for each unique value of `group`.

Author(s)

Ed Merkle

References

Toda, M. (1963). Measurement of subjective probability distributions. ESD-TDR-63-407, Decision Sciences Laboratory, L. G. Hanscom Field, Bedford, Mass.

Shuford, E. H., Albert, A., & Massengill, H. E. (1966). Admissible probability measurement procedures. *Psychometrika, 31*, 125-145.

See Also

calcscore

Examples

```
data("WorldEvents")
## Raw log scores (0 best, infinity worst)
log1 <- logscore(answer ~ forecast, data=WorldEvents)
## Raw log scores (0 best, -infinity worst)
log1 <- logscore(answer ~ forecast, data=WorldEvents,
                 reverse = TRUE)
## Raw log scores plus group means
log2 <- logscore(answer ~ forecast, data=WorldEvents,
                 group="forecaster")
```

---

plotscore — *Plot a Two-Alternative Scoring Rule*

Description

Given parameters for a two-alternative scoring rule, plot the hypothetical scores that would be obtained for each forecast/outcome combination.

Usage

```
plotscore(param = c(2, 0.5), fam = "pow", bounds, reverse = FALSE,
          legend = TRUE, ...)
```

Arguments

`param`	Numeric vector of length 2, containing the parameters for `fam`. For family `beta`, these are the parameters commonly denoted alpha and beta. For families `pow` and `sph`, these correspond to the family parameter gamma and the baseline parameter associated with the focal outcome, respectively.
`fam`	scoring rule family. `pow` (default) is the power family, `beta` is the beta family, `sph` is the pseudospherical family.
`bounds`	Lower and upper bounds supplied to `calcscore`.
`reverse`	`reverse` argument supplied to `calcscore`.
`legend`	Should a legend be displayed? Defaults to `TRUE`
`...`	Other arguments to `plot()`

Details

For more information on the scoring rule families and the bounds and `reverse` arguments, see the details of `calcscore()`.

Value

Returns the result of a `plot()` call that graphs the scoring rule.

Author(s)

Ed Merkle

References

Buja, A., Stuetzle, W., & Shen, Y. (2005). Loss functions for binary class probability estimation and classification: Structure and applications. (Obtained from http://stat.wharton.upenn.edu/~buja/PAPERS/)

Jose, V. R. R., Nau, R. F., & Winkler, R. L. (2008). Scoring rules, generalized entropy, and utility maximization. *Operations Research, 56*, 1146–1157.

Jose, V. R. R., Nau, R. F., & Winkler, R. L. (2009). Sensitivity to distance and baseline distributions in forecast evaluation. *Management Science, 55*, 582–590.

Merkle, E. C. & Steyvers, M. (in press). Choosing a strictly proper scoring rule. *Decision Analysis*.

See Also

calcscore

Examples

```
## Plot Brier score from power family with natural bounds
plotscore(c(2,.5), fam="pow")

## Plot Brier score from beta family with bounds of 0 and 1
plotscore(c(1,1), fam="beta", bounds=c(0,1))

## Plot log score
plotscore(c(0,0), fam="beta")

## Score from pseudospherical family with
## baseline of .3 and (0,1) bounds
plotscore(c(3, .3), fam="sph", bounds=c(0,1))
```

`sphscore` *Calculate Spherical Scores*

Description

Calculate spherical scores and average spherical scores by a grouping variable.

Usage

```
sphscore(object, data, group = NULL, bounds = NULL, reverse = FALSE)
```

Arguments

`object`	an object of class "formula", of the form `outcome ~ forecast`. See `calcscore()` documentation for further details.
`data`	an optional data frame or list containing the variables in the formula. If not found in `data`, the variables are taken from the environment from which `calcscore` is called.
`group`	the name of a grouping variable within `data`, which is used to calculate average spherical score by group.
`bounds`	a vector of length 2 corresponding to the desired minimum and maximum spherical score, respectively.
`reverse`	if `FALSE` (default), smaller scores imply better forecasts. If `TRUE`, larger scores imply better forecasts.

Details

If `group` is supplied, the function returns a list (see value section). Otherwise, the function returns a numeric vector containing the spherical score associated with each forecast.

Value

Depending on input arguments, `sphscore` may return an object of class `numeric` containing raw spherical scores. It may also return a list containing the objects below.

`rawscores`	an object of class `numeric` containing raw spherical scores for each forecast.
`mnsph`	an object of class `numeric` containing mean spherical scores for each unique value of `group`.

Author(s)

Ed Merkle

References

Toda, M. (1963). Measurement of subjective probability distributions. ESD-TDR-63-407, Decision Sciences Laboratory, L. G. Hanscom Field, Bedford, Mass.

Shuford, E. H., Albert, A., & Massengill, H. E. (1966). Admissible probability measurement procedures. *Psychometrika, 31*, 125-145.

See Also

calcscore

Examples

```
data("WorldEvents")
## Raw spherical scores
sph1 <- sphscore(answer ~ forecast, data=WorldEvents)
## Raw spherical scores plus group means
sph2 <- sphscore(answer ~ forecast, data=WorldEvents,
                 group="forecaster")
```

WeatherProbs *Three-category weather forecasts*

Description

Probabilistic forecasts from the U.S. National Oceanic and Atmospheric Administration, concerning below/near/above average temperatures and below/near/above median precipitation.

Usage

```
data("WeatherProbs")
```

Format

A data frame with 8976 observations on the following 11 variables.

`stn` Station World Meteorological Organization (WMO) number

`made` Forecast issuance date

`valid` Center of forecast valid period

`tblw` Probability of below normal temperatures

`tnrm` Probability of near normal temperatures

`tabv` Probability of above normal temperatures

`tcat` Realized temperature category (1=below, 2=near, 3=above)

`pblw` Probability of below median precipitation

`pnrm` Probability of near median precipitation

`pabv` Probability of above median precipitation

`pcat` Realized precipitation category (1=below, 2=near, 3=above)

Details

The forecasts are valid for a period of 6 to 10 days from the date that the forecast was made. The forecasts were supplied every weekday during April, 2009, and they specifically predict the average temperature or total precipitation for the entire valid period.

Source

Data were obtained from http://www.cpc.ncep.noaa.gov/products/archives/short_range/ (see URL in references).

References

See http://www.cpc.ncep.noaa.gov/products/archives/short_range/README.6-10day.txt for more details on the data.

For an application of similar data (different dates, same source), see:

Wilks, D. S. (in press). The calibration simplex: A generalization of the reliability diagram for 3-category probability forecasts. *Weather and Forecasting*.

Examples

```
data("WeatherProbs")

## Brier score for temperature forecasts
## (Warning arises because some forecast rows don't sum to 1.)
res <- calcscore(tcat ~ tblw + tnrm + tabv, data=WeatherProbs,
                 bounds=c(0,1))

## Ordered Brier score for temperature forecasts
res2 <- calcscore(tcat ~ tblw + tnrm + tabv, data=WeatherProbs,
                  bounds=c(0,1), ordered=TRUE)

## Spherical score for temperature forecasts
res3 <- calcscore(tcat ~ tblw + tnrm + tabv, data=WeatherProbs,
                  fam="sph", bounds=c(0,1))

## Average scores by station
avgbrier <- with(WeatherProbs, tapply(res, stn, mean))
avgobrier <- with(WeatherProbs, tapply(res2, stn, mean))
avgsph <- with(WeatherProbs, tapply(res3, stn, mean))

## Conclusions vary across Brier and ordinal Brier scores
plot(avgbrier, avgobrier, pch=20, xlab="Brier", ylab="Ordinal Brier")
```

`WorldEvents` *Forecasts of world events*

Description

Probabilistic forecasts of three world events, provided by seven MTurkers.

Usage

```
data("WorldEvents")
```

Format

A data frame with forecasts of three world events provided by seven Mechanical Turk users.

`forecaster` Forecaster ID

`item` Item ID (see details)

`answer` Item resolution (0/1)

`forecast` Forecast associated with outcome 1

Details

The three forecasted items were:

1. The UK will leave the European Union before the end of 2012.
2. Before Jan 1, 2013, Apple will announce it has sold more than 10 million iPad minis.
3. Japan's nuclear plant in Tsuruga will remain idle between June 1 and December 31, 2012.

For each item, `outcome=1` implies that the item text did occur and `outcome=0` implies that the item text did not occur. Forecasts were provided on Dec 20, 2012.

Source

Unpublished data provided by Ed Merkle.

Examples

```
data("WorldEvents")

## Average forecast for each item
with(WorldEvents, tapply(forecast, item, mean))

## Brier scores
bs <- calcscore(answer ~ forecast, data = WorldEvents, bounds=c(0,1))
```

Index