Package 'isodisregAFSD'

August 30, 2023

Title Computes IDR for distributional input

Version 1.0

Description What the package does (one paragraph).

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

LinkingTo Rcpp

Imports Rcpp

URL https://github.com/evwalz/isodisregAFSD

BugReports https://github.com/evwalz/isodisregAFSD/issues

Suggests knitr,
rmarkdown

VignetteBuilder knitr

R topics documented:

isodisregAFSD-package *Isotonic distributional Regression (IDR) for distributional data*

Description

Isotonic distributional Regression (IDR) is a nonparametric method to estimate conditional distributions under monotonicity constraints.

How does it work?

Provide some more detailes

bscore *Brier score for forecast probability of threshold exceedance*

Description

Computes the Brier score of forecast probabilities for exceeding given thresholds.

Usage

```
bscore(predictions, thresholds, y)
```

Arguments

predictions	either an object of class idrafsd (output of predict.idrcal), or a data.frame of numeric variables. In the latter case, the CDF is computed using the empirical distribution of the variables in predictions.
thresholds	numeric vector of thresholds at which the CDF will be evaluated.
y	a numeric vector of obervations of the same length as the number of predictions, or of length 1. In the latter case, y will be used for all predictions.

Details

The Brier score for the event of exceeding a given threshold z is defined as

$$(1\{y > z\} - P(y > z))^2$$

where *y* is the observation and *P(y > z)* the forecast probability for exceeding the threshold z.

Value

A matrix of the Brier scores for the desired thresholds, one column per threshold.

References

Gneiting, T. and Raftery, A. E. (2007), 'Strictly proper scoring rules, prediction, and estimation', Journal of the American Statistical Association 102(477), 359-378

See Also

predict.idrcal, cdf

Examples

```
data("rain")

## Postprocess HRES forecast using data of 3 years

X <- rain[1:(3 * 365), "HRES", drop = FALSE]
y <- rain[1:(3 * 365), "obs"]

fit <- idr(y = y, X = X)

## Compute Brier score for postprocessed probability of precipitation
## forecast using data of the next 2 years (out-of-sample predictions)

data <- rain[(3 * 365 + 1):(5 * 365), "HRES", drop = FALSE]
obs <- rain[(3 * 365 + 1):(5 * 365), "obs"]
predictions <- predict(fit, data = data)
score <- bscore(predictions, thresholds = 0, y = obs)

mean(score)
```

---

cdf *Cumulative distribution function (CDF) of IDR or raw forecasts*

Description

Evaluate the the cumulative distribution function (CDF) of IDR predictions or of unprocessed forecasts in a data.frame.

Usage

```
cdf(predictions, thresholds)

## S3 method for class 'idrafsd'
cdf(predictions, thresholds)

## S3 method for class 'data.frame'
cdf(predictions, thresholds)
```

Arguments

predictions	either an object of class idrafsd (output of predict.idrcal), or a data.frame of numeric variables. In the latter case, the CDF is computed using the empirical distribution of the variables in predictions.
thresholds	numeric vector of thresholds at which the CDF will be evaluated.

Details

The CDFs are considered as piecewise constant stepfunctions: If x are the points where the IDR fitted CDF (or the empirical distribution of the forecasts) has jumps and p the corresponding CDF values, then for x[i] <= x < x[i + 1], the CDF at x is p[i].

Value

A matrix of probabilities giving the evaluated CDFs at the given thresholds, one column for each threshold.

See Also

predict.idrcal qpred, bscore

Examples

```
## Example from IDR package: Adapt to idrafsd
data("rain")

## Postprocess HRES forecast using data of 3 years

X <- rain[1:(3 * 365), "HRES", drop = FALSE]
y <- rain[1:(3 * 365), "obs"]

fit <- idr(y = y, X = X)

## Compute probability of precipitation given that the HRES forecast is
## 0 mm, 0.5 mm or 1 mm

predictions <- predict(fit, data = data.frame(HRES = c(0, 0.5, 1)))
1 - cdf(predictions, thresholds = 0)
```

---

crps *Continuous ranked probability score (CRPS)*

Description

Computes the CRPS of IDR or raw forecasts.

Usage

```
crps(predictions, y)

## S3 method for class 'idrafsd'
crps(predictions, y)

## S3 method for class 'data.frame'
crps(predictions, y)
```

Arguments

predictions	either an object of class idrafsd (output of predict.idrcal), or a data.frame of numeric variables. In the latter case, the CRPS is computed using the empirical distribution of the variables in predictions.
y	a numeric vector of obervations of the same length as the number of predictions, or of length 1. In the latter case, y will be used for all predictions.

Details

This function uses adapted code taken from the function `crps_edf` of the **scoringRules** package.

Value

A vector of CRPS values.

References

Jordan A., Krueger F., Lerch S. (2018). "Evaluating Probabilistic Forecasts with scoringRules." Journal of Statistical Software. Forthcoming.

Gneiting, T. and Raftery, A. E. (2007), 'Strictly proper scoring rules, prediction, and estimation', Journal of the American Statistical Association 102(477), 359-378

See Also

`predict.idrcal`

Examples

```
data("rain")

## Postprocess HRES forecast using data of 3 years

X <- rain[1:(3 * 365), "HRES", drop = FALSE]
y <- rain[1:(3 * 365), "obs"]

fit <- idr(y = y, X = X)

## Compute CRPS of postprocessed HRES forecast using data of the next 2 years
## (out-of-sample predictions)

data <- rain[(3 * 365 + 1):(5 * 365), "HRES", drop = FALSE]
obs <- rain[(3 * 365 + 1):(5 * 365), "obs"]
predictions <- predict(fit, data = data)
idrCrps <- crps(predictions, y = obs)

## Compare this to CRPS of the raw ensemble of all forecasts (high resolution,
## control and 50 perturbed ensemble forecasts)

rawData <- rain[(3 * 365 + 1):(5 * 365), c("HRES", "CTR", paste0("P", 1:50))]
rawCrps <- crps(rawData, y = obs)

c("idr_HRES" = mean(idrCrps), "raw_all" = mean(rawCrps))
```

---

idrafsd *Fit IDR to training data*

Description

Fits isotonic distributional regression (IDR) to a training dataset with respect to the AFSD order.

Usage

```
idr(y, X, eps, pars = osqpSettings(verbose = FALSE, eps_abs =
  1e-5, eps_rel = 1e-5, max_iter = 10000L), progress = TRUE)
```

Arguments

y	numeric vector (the response variable).
X	data frame of numeric or ordered factor variables (the regression covariates).
eps	a parameter in (0,0.5) to specify the AFSD order.
pars	parameters for quadratic programming optimization (only relevant if X has more than one column), set using osqpSettings.
progress	display progressbar (TRUE, FALSE or 1, 0)?

Details

This function computes the isotonic distributional regression (IDR) of a response y on covariates that consists of ensemble forecasts and are restored in a data-frame X. IDR estimates the cumulative distribution function (CDF) of y conditional on X by monotone regression, assuming that y is more likely to take higher values, as X increases. Formally, IDR assumes that the conditional CDF $F_{y|X=x}(z)$ at each fixed threshold z decreases, as x increases, or equivalently, that the exceedance probabilities for any threshold z $P(y > z|X = x)$ increase with x.

The conditional CDFs are estimated at each threshold in `unique(y)`. This is the set where the CDFs may have jumps. If `X` contains more than one variable, the CDFs are estimated by calling `solve_osqp` from the package **osqp** `length(unique(y))` times. This might take a while if the training dataset is large.

Value

An object of class `"idrfit"` containing the following components:

X	data frame of all distinct covariate combinations used for the fit (note the different order!).
y	list of all observed responses in the training data for given covariate combinations in X.
cdf	matrix containing the estimated CDFs, one CDF per row, evaluated at thresholds (see next point). The CDF in the ith row corredponds to the estimated conditional distribution of the response given the covariates values in X[i,].
thresholds	the thresholds at which the CDFs in cdf are evaluated. The entries in cdf[,j] are the conditional CDFs evaluated at thresholds[j].
diagnostic	list giving a bound on the precision of the CDF estimation (the maximal downwards-step in the CDF that has been detected) and the fraction of CDF estimations that were stopped at the iteration limit max_iter. Decrease the parameters eps_abs and/or eps_rel or increase max_iter in pars to improve the precision. See osqpSettings for more optimization parameters.
indices	the row indices of the covariates in X in the original training dataset (used for in-sample predictions with predict.idrfit).
constraints	(in multivariate IDR, NULL otherwise) matrices giving the order constraints for optimization. Used in predict.idrfit.

Note

The function `idr` is only intended for fitting IDR model for a training dataset and storing the results for further processing, but not for prediction or evaluation, which is done using the output of `predict.idrfit`.

See Also

The S3 method `predict.idrfit` for predictions based on an IDR fit.

---

pit *Probability integral transform (PIT)*

Description

Computes the probability integral transform (PIT) of IDR or raw forecasts.

Usage

```
pit(predictions, y, randomize = TRUE, seed = NULL)

## S3 method for class 'idrafsd'
pit(predictions, y, randomize = TRUE, seed = NULL)

## S3 method for class 'data.frame'
pit(predictions, y, randomize = TRUE, seed = NULL)
```

Arguments

`predictions`	either an object of class `idrafsd` (output of `predict.idrcal`), or a `data.frame` of numeric variables. In the latter case, the PIT is computed using the empirical distribution of the variables in `predictions`.
`y`	a numeric vector of obervations of the same length as the number of predictions.
`randomize`	PIT values should be randomized at discontinuity points of the predictive CDF (e.g. at zero for precipitation forecasts). Set `randomize = TRUE` to randomize.
`seed`	argument to `set.seed` for random number generation (if `randomize` is `TRUE`).

Value

Vector of PIT values.

References

Gneiting, T., Balabdaoui, F. and Raftery, A. E. (2007), 'Probabilistic forecasts, calibration and sharpness', Journal of the Royal Statistical Society: Series B (Statistical Methodology) 69(2), 243-268.

See Also

`predict.idrcal`

Examples

```
data("rain")
require("graphics")

## Postprocess HRES forecast using data of 4 years

X <- rain[1:(4 * 365), "HRES", drop = FALSE]
y <- rain[1:(4 * 365), "obs"]

fit <- idr(y = y, X = X)

## Assess calibration of the postprocessed HRES forecast using data of next 4
## years and compare to calibration of the raw ensemble

data <- rain[(4 * 365 + 1):(8 * 365), "HRES", drop = FALSE]
obs <- rain[(4 * 365 + 1):(8 * 365), "obs"]
predictions <- predict(fit, data = data)
idrPit <- pit(predictions, obs, seed = 123)

rawData <- rain[(4 * 365 + 1):(8 * 365), c("HRES", "CTR", paste0("P", 1:50))]
rawPit <- pit(rawData, obs, seed = 123)

hist(idrPit, xlab = "Probability Integral Transform",
  ylab = "Density", freq = FALSE, main = "Postprocessed HRES")
hist(rawPit, xlab = "Probability Integral Transform",
  ylab = "Density", freq = FALSE, main = "Raw ensemble")
```

`plot.idrafsd` *Plot IDR predictions*

Description

Plot an IDR predictive CDF.

Usage

```
## S3 method for class 'idrafsd'
plot(
  x,
  index = 1,
  bounds = TRUE,
  col.cdf = "black",
  col.bounds = "blue",
  lty.cdf = 1,
  lty.bounds = 3,
  xlab = "Threshold",
  ylab = "CDF",
  main = "IDR predictive CDF",
  ...
)
```

Arguments

x	object of class idrafsd (output of predict.idrcal).
index	index of the prediction in x for which a plot is desired.
bounds	whether the bounds should be plotted or not (see predict.idrfit for details about the meaning of the bounds).
col.cdf	color of the predictive CDF.
col.bounds	color of the bounds.
lty.cdf	linetype of the predictive CDF.
lty.bounds	linetype of the CDF bounds.
xlab	label for x axis.
ylab	label for y axis.
main	main title.
...	further arguments to plot.stepfun or plot.

Value

The data based on which the plot is drawn (returned invisible).

See Also

predict.idrcal

Examples

```
data("rain")
require("graphics")

## Postprocess HRES and CTR forecast using data of 2 years

X <- rain[1:(2 * 365), c("HRES", "CTR"), drop = FALSE]
y <- rain[1:(2 * 365), "obs"]

## Fit IDR and plot the predictive CDF when the HRES forecast is 1 mm and
## CTR is 0 mm

fit <- idr(y = y, X = X)
pred <- predict(fit, data = data.frame(HRES = 1, CTR = 0))
plot(pred)
```

---

predict.idrcal *Predict method for IDR fits*

Description

Prediction based on IDR model fit.

Usage

```
## S3 method for class 'idrcal'
predict(
  object,
  data = NULL,
  grid = NULL,
  digits = 3,
  interpolation = "linear",
  ...
)
```

Arguments

object	IDR fit (object of class "idrfit").
data	optional data.frame containing variables with which to predict. In-sample predictions are returned if this is omitted.
digits	number of decimal places for the predictive CDF.
...	included for generic function consistency.

Details

If the variables x = data[j,] for which predictions are desired are already contained in the training dataset X for the fit, predict.idrfit returns the corresponding in-sample prediction. Otherwise monotonicity is used to derive upper and lower bounds for the predictive CDF, and the predictive CDF is a pointwise average of these bounds.

If the lower and the upper bound on the predictive cdf are far apart (or trivial, i.e. constant 0 or constant 1), this indicates that the prediction based on x is uncertain because either the training dataset is too small or only few similar variable combinations as in x have been observed in the training data. However, *the bounds on the predictive CDF are not prediction intervals and should not be interpreted as such. They only indicate the uncertainty of out-of-sample predictions for which the variables are not contained in the training data.*

If the new variables x are greater than all X[i,] in the selected order(s), the lower bound on the cdf is trivial (constant 0) and the upper bound is taken as predictive cdf. The upper bound on the cdf is trivial (constant 1) if x is smaller than all X[i,]. If x is not comparable to any row of X in the given order, a prediction based on the training data is not possible. In that case, the default forecast is the empirical distribution of y in the training data.

Value

A list of predictions. Each prediction is a data.frame containing the following variables:

points	the points where the predictive CDF has jumps.
cdf	the estimated CDF evaluated at the points.
lower, upper	(only for out-of-sample predictions) bounds for the estimated CDF, see 'Details' above.

The output has the attribute incomparables, which gives the indices of all predictions for which the climatological forecast is returned because the forecast variables are not comparable to the training data.

See Also

idr to fit IDR to training data.

cdf, qpred to evaluate the CDF or quantile function of IDR predictions.

bscore, qscore, crps, pit to compute Brier scores, quantile scores, the CRPS and the PIT of IDR predictions.

plot to plot IDR predictive CDFs.

---

qpred *Quantile function of IDR or raw forecasts*

Description

Evaluate the the quantile function of IDR predictions or of unprocessed forecasts in a data.frame.

Usage

```
qpred(predictions, quantiles)

## S3 method for class 'idrafsd'
qpred(predictions, quantiles)

## S3 method for class 'data.frame'
qpred(predictions, quantiles)
```

Arguments

predictions	either an object of class idrafsd (output of predict.idrcal), or a data.frame of numeric variables. In the latter case, quantiles are computed using the empirical distribution of the variables in predictions.
quantiles	numeric vector of desired quantiles.

Details

The quantiles are defined as lower quantiles, that is,

$$q(u) = inf(x : cdf(x) >= u).$$

Value

A matrix of forecasts for the desired quantiles, one column per quantile.

See Also

predict.idrcal, cdf, qscore

Examples

```
data("rain")

## Postprocess HRES forecast using data of 3 years

X <- rain[1:(3 * 365), "HRES", drop = FALSE]
y <- rain[1:(3 * 365), "obs"]

fit <- idr(y = y, X = X)

## Compute 95%-quantile forecast given that the HRES forecast is
## 2.5 mm, 5 mm or 10 mm

predictions <- predict(fit, data = data.frame(HRES = c(2.5, 5, 10)))
qpred(predictions, quantiles = 0.95)
```

`qscore` *Quantile scores for IDR or raw forecasts*

Description

Computes quantile scores of IDR quantile predictions or of quantile predictions from raw forecasts in a `data.frame`.

Usage

```
qscore(predictions, quantiles, y)
```

Arguments

`predictions`	either an object of class `idrafsd` (output of `predict.idrcal`), or a `data.frame` of numeric variables. In the latter case, quantiles are computed using the empirical distribution of the variables in `predictions`.
`quantiles`	numeric vector of desired quantiles.
`y`	a numeric vector of obervations of the same length as the number of predictions, or of length 1. In the latter case, `y` will be used for all predictions.

Details

The quantile score of a forecast x for the u-quantile is defined as

$$2(1x > y - u)(x - y),$$

where y is the observation. For $u = 1/2$, this equals the mean absolute error of the median forecast.

Value

A matrix of the quantile scores for the desired quantiles, one column per quantile.

References

Gneiting, T. and Raftery, A. E. (2007), 'Strictly proper scoring rules, prediction, and estimation', Journal of the American Statistical Association 102(477), 359-378

See Also

predict.idrcal, qpred

Examples

```
data("rain")

## Postprocess HRES forecast using data of 3 years

X <- rain[1:(3 * 365), "HRES", drop = FALSE]
y <- rain[1:(3 * 365), "obs"]

fit <- idrafsd(y = y, X = X)

## Compute mean absolute error of the median postprocessed forecast using
## data of the next 2 years (out-of-sample predictions) and compare to raw
## HRES forecast

data <- rain[(3 * 365 + 1):(5 * 365), "HRES", drop = FALSE]
obs <- rain[(3 * 365 + 1):(5 * 365), "obs"]

predictions <- predict(fit, data = data)
idrMAE <- mean(qscore(predictions, 0.5, obs))
rawMAE <- mean(qscore(data, 0.5, obs))

c("idr" = idrMAE, "raw" = rawMAE)
```

Index