Package ‘rfinterval’

July 18, 2019

Type Package
Title Predictive Inference for Random Forests
Version 1.0.0
Date 2019-07-14
Maintainer Haozhe Zhang <haozhe.stat@gmail.com>
Description An integrated package for constructing random forest prediction intervals using a fast implementation package ‘ranger’. This package can apply the following three methods described in Haozhe Zhang, Joshua Zimmerman, Dan Nettleton, and Daniel J. Nordman (2019) <doi:10.1080/00031305.2019.1585288>: the out-of-bag prediction interval, the split conformal method, and the quantile regression forest.
License GPL-3
Imports ranger, MASS
Depends R (>= 3.1)
URL http://github.com/haozhestat/rfinterval
BugReports http://github.com/haozhestat/rfinterval/issues
Suggests testthat
LazyData true
Encoding UTF-8
RoxygenNote 6.1.1
Language en-US
NeedsCompilation no
Author Haozhe Zhang [aut, cre] (<https://orcid.org/0000-0002-7771-4808>)
Repository CRAN
Date/Publication 2019-07-18 16:40:04 UTC

R topics documented:

BeijingPM25 ................................................................. 2
rfinterval ................................................................. 3
sim_data ................................................................. 4
BeijingPM25

Beijing PM2.5 Air Pollution Data

Description

This hourly data set contains the PM2.5 data of US Embassy in Beijing. Meanwhile, meteorological data from Beijing Capital International Airport are also included.

Usage

BeijingPM25

Format

A data frame with 8661 rows and 11 variables:

- **pm2.5** PM2.5 concentration (ug/m^3)
- **month** month of observation
- **day** day of observation
- **hour** hour of observation
- **DEWP** dew point
- **TEMP** temperature
- **PRES** air pressure
- **cbwd** combined wind direction
- **Iws** cumulated wind speed
- **Is** cumulated hours of snow
- **Ir** cumulated hours of rain

Source

rfinterval

Prediction Intervals for Random forests

Description

The rfinterval constructs prediction intervals for random forest predictions using a fast implementation package 'ranger'.

Usage

rfinterval(formula = NULL, train_data = NULL, test_data = NULL, method = c("oob", "split-conformal", "quantreg"), alpha = 0.1, symmetry = TRUE, seed = NULL, params_ranger = NULL)

Arguments

formula Object of class formula or character describing the model to fit. Interaction terms supported only for numerical variables.
train_data Training data of class data.frame, matrix, or dgCMatrix (Matrix).
test_data Test data of class data.frame, matrix, or dgCMatrix (Matrix).
method Method for constructing prediction interval. If method = "oob", compute the out-of-bag prediction intervals; if method = "split-conformal", compute the split conformal prediction interval; if method = "quantreg", use quantile regression forest to compute prediction intervals.
alpha Confidence level. alpha = 0.05 for the 95% prediction interval.
symmetry True if constructing symmetric out-of-bag prediction intervals, False otherwise. Only for method = "oob"
seed Seed (only for method = "split-conformal")
params_ranger List of further parameters that should be passed to ranger. See ranger for possible parameters.

Value

oob_interval Out-of-bag prediction intervals
sc_interval Split-conformal prediction intervals
quantreg_interval Quantile regression forest prediction intervals
alpha Confidence level for prediction intervals
testPred Random forest prediction for test set
train_data Training data
test_data Test data
References


Examples

```r
train_data <- sim_data(n = 500, p = 8)
test_data <- sim_data(n = 500, p = 8)
output <- rfinterval(y~., train_data = train_data, test_data = test_data,
method = c("oob", "split-conformal", "quantreg"),
symmetry = TRUE, alpha = 0.1)
y <- test_data$y
mean(output$oob_interval$lo < y & output$oob_interval$up > y)
mean(output$sc_interval$lo < y & output$sc_interval$up > y)
mean(output$quantreg_interval$lo < y & output$quantreg_interval$up > y)
```

**sim_data**  
Simulate data

**Description**

Simulate data for illustrate the performance of prediction intervals for random forests

**Usage**

```r
sim_data(n = 500, p = 10, rho = 0.6, predictor_dist = "correlated",
mean_function = "nonlinear-interaction",
error_dist = "homoscedastic")
```

**Arguments**

- `n`  
  Sample size

- `p`  
  Number of features

- `rho`  
  Correlation between predictors

- `predictor_dist`  
  Distribution of predictor: "uncorrelated", and "correlated"
sim_data

mean_function
   Mean function: "linear", "nonlinear", and "nonlinear-interaction"

error_dist
   Distribution of error: "homoscedastic", "heteroscedastic", and "heavy-tailed"

Value

   a data.frame of simulated data

Examples

   train_data <- sim_data(n = 500, p = 10)
   test_data <- sim_data(n = 500, p = 10)