Package: groupWQS (via r-universe)

October 24, 2026

Type Package

Title Grouped Weighted Quantile Sum Regression

Version 0.0.3

Author David Wheeler, Matthew Carli

Maintainer Matthew Carli <carlimm@mymail.vcu.edu>

Description Fits weighted quantile sum (WQS) regressions for one or more chemical groups with continuous or binary outcomes. Wheeler D, Czarnota J.(2016) <doi:10.1289/isee.2016.4698>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Depends R (>= 3.2.1)

Imports Rsolnp, glm2, stats, graphics, MASS, rjags

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Repository https://cran.r-universe.dev

Date/Publication 2020-06-27 17:10:02 UTC

RemoteUrl https://github.com/cran/groupWQS

RemoteRef HEAD

RemoteSha 74d4ebee1fcb20c966d4900a1ab3dc6f8834a94f

Contents

gwqs.fit 2
make.X 3
make.x.s 4
simdata 5
weight.plot 6
WQSdata 6

Index **8**

`gwqs.fit` *Grouped WQS Regression*

Description

This function fits a grouped weighted quantile sum (GWQS) regression model.

Usage

```
gwqs.fit(
  y,
  y.train = NULL,
  x,
  x.train = NULL,
  z = NULL,
  z.train = NULL,
  x.s,
  B = 100,
  n.quantiles = 4,
  pars = NULL,
  func,
  ineqLB = NULL,
  ineqUB = NULL,
  tol = 1e-06,
  delta = 1e-06
)
```

Arguments

`y`	A vector containing outcomes for validation.
`y.train`	A vector containing outcomes for training. If left as NULL the validation data will be used for training as well.
`x`	A matrix of component data for validation.
`x.train`	A matrix of component data for training. If left as NULL the validation data will be used for training as well.
`z`	A vector or matrix of covariates for validation.
`z.train`	A vector or matrix of covariates for training. If left as NULL the validation data will be used for training as well.
`x.s`	A vector of the number of components in each index.
`B`	The number of bootstrap samples, must be 1 or more.
`n.quantiles`	The number of quantiles to apply to data.
`pars`	A vector of initial values, listed in order: beta naught intercept and group index beta coefficients, individual chemical weight coefficients, and covariate coefficients.

func	The objective function to be used (must match outcome data type); currently only fun args "continuous" or "binary" are supported.
ineqLB	Vector of lower bounds for betas and weights, set to -2 by default.
ineqUB	Vector of upper bounds for betas and weights, set to 2 be default.
tol	Tolerance level for bootstrap convergence.
delta	Step size for bootstrap procedure.

Value

A list of 3 containing the GWQS estimate based on calculated weights, the GWQS model fit to validation data, and weight estimates

Examples

```
data("WQSdata")
group_list <- list(c("X1", "X2", "X3"), c("X4", "X7"), c("X5", "X6", "X9", "X8"))
x.s <- make.x.s(WQSdata, 3, group_list)
X <- make.X(WQSdata, 3, group_list)
Y <- WQSdata$y
results <- gwqs.fit(y = Y, x = X, x.s = x.s, B=1, func = "continuous")
```

---

make.X *Forms matrix of components*

Description

This function returns a matrix of component variables, X. The user can specify the desired chemicals and order by creating a list of string vectors, each vector containing the variable names of all desired elements of that group.

Usage

```
make.X(df, num.groups, groups)
```

Arguments

df	A dataframe containing named component variables
num.groups	An integer representing the number of component groups desired
groups	A list, each item in the list being a string vector of variable names for one component group

Value

A matrix of component variables

Examples

```
data("WQSdata")
group_list <- list(c("X1", "X2", "X3"), c("X4", "X7"), c("X5", "X6", "X9", "X8"))
X <- make.X(WQSdata, 3, group_list)
X
```

`make.x.s` *Forms component group ID vector of X*

Description

This function returns a vector which lets WQS.fit know the size and order of groups in X

Usage

```
make.x.s(df, num.groups, groups)
```

Arguments

`df`	A dataframe containing named component variables
`num.groups`	An integer representing the number of component groups desired
`groups`	A list, each item in the list being a string vector of variable names for one component group

Value

A vector of integers, each integer relating how many columns are in each group

Examples

```
data("WQSdata")
group_list <- list(c("X1", "X2", "X3"), c("X4", "X7"), c("X5", "X6", "X9", "X8"))
x.s <- make.x.s(WQSdata, 3, group_list)
x.s
```

`simdata` *Simulated data of chemical concentrations and one binary outcome variable*

Description

Data simulated to have .7 in-group correlation and .3 between-group correlation. There are three groups, the third being significantly correlated to the outcome variable

Usage

```
simdata
```

Format

A data frame with 1000 rows and 15 variables:

pcb_118 a numeric vector; part of group 1

pcb_138 a numeric vector; part of group 1

pcb_153 a numeric vector; part of group 1

pcb_180 a numeric vector; part of group 1

pcb_192 a numeric vector; part of group 1

as a numeric vector; part of group 2

cu a numeric vector; part of group 2

pb a numeric vector; part of group 2

sn a numeric vector; part of group 2

carbaryl a numeric vector; part of group 3

propoxur a numeric vector; part of group 3

methoxychlor a numeric vector; part of group 3

diazinon a numeric vector; part of group 3

chlorpyrifos a numeric vector; part of group 3

Y a numeric vector; the outcome variable

weight.plot *Generates Plots of weights by group*

Description

This function takes the object created by the wqs.fit function and a vector of group names and generates a random forest variable importance plot for each group. The weights in each group are listed in descending order.

Usage

```
weight.plot(fit.object, group.names)
```

Arguments

fit.object	The object that is returned by the wqs.fit function
group.names	A string vector containing the name of each group included in the GWQS regression. Will be used for plot titles.

Value

A plot for each group of the GWQS regression

Examples

```
data("WQSdata")
group_list <- list(c("X1", "X2", "X3"), c("X4", "X7"), c("X5", "X6", "X9", "X8"))
chem_groups <- c("PCBs", "Metals", "Insecticides")
x.s <- make.x.s(WQSdata, 3, group_list)
X <- make.X(WQSdata, 3, group_list)
Y <- WQSdata$y
results <- gwqs.fit(y = Y, x = X, x.s = x.s, B=1, func = "continuous")
weight.plot(results, chem_groups)
```

WQSdata *Simulated data of chemical concentrations and one continuous outcome variable*

Description

Correlation and concentration patterns were loosely based on NHL data.

Usage

```
WQSdata
```

Format

A data frame with 1000 rows and 10 variables:

X1 a numeric vector

X2 a numeric vector

X3 a numeric vector

X4 a numeric vector

X5 a numeric vector

X6 a numeric vector

X7 a numeric vector

X8 a numeric vector

X9 a numeric vector

y a numeric vector; the outcome variable

Index

∗ **datasets**
simdata, 5
WQSdata, 6

gwqs.fit, 2

make.X, 3
make.x.s, 4

simdata, 5

weight.plot, 6
WQSdata, 6