Package ‘crmReg’

September 23, 2020

Type Package
Title Cellwise Robust M-Regression and SPADIMO
Version 1.0.2
Description Method for fitting a cellwise robust linear M-regression model (CRM, Filzmoser et al. (2020) <DOI:10.1016/j.csda.2020.106944>) that yields both a map of cellwise outliers consistent with the linear model, and a vector of regression coefficients that is robust against vertical outliers and leverage points. As a by-product, the method yields an imputed data set that contains estimates of what the values in cellwise outliers would need to amount to if they had fit the model. The package also provides diagnostic tools for analyzing casewise and cellwise outliers using sparse directions of maximal outlyingness (SPADIMO, Debruyne et al. (2019) <DOI:10.1007/s11222-018-9831-5>.
Depends R (>= 3.5.0)
Imports FNN, ggplot2, gplots, pcaPP, plyr, robustbase, rrcov
License GPL (>= 2)
Author Peter Filzmoser [aut],
    Sebastiaan Hoppner [aut, cre],
    Irene Ortner [aut],
    Sven Sermeels [aut],
    Tim Verdonck [aut]
Maintainer Sebastiaan Hoppner <sebastiaan.hoppner@gmail.com>
LazyLoad yes
Encoding UTF-8
LazyData false
NeedsCompilation yes
RoxygenNote 6.1.1
Repository CRAN
Date/Publication 2020-09-23 08:40:11 UTC

R topics documented:

    crmReg-package .......................................................... 2
Description

Method for fitting a cellwise robust linear M-regression model (CRM, Filzmoser et al. (2020) <DOI:10.1016/j.csda.2020.106944>) that yields both a map of cellwise outliers consistent with the linear model, and a vector of regression coefficients that is robust against vertical outliers and leverage points. As a by-product, the method yields an imputed data set that contains estimates of what the values in cellwise outliers would need to amount to if they had fit the model. The package also provides diagnostic tools for analyzing casewise and cellwise outliers using sparse directions of maximal outlyingness (SPADIMO, Debruyne et al. (2019) <DOI:10.1007/s11222-018-9831-5>).

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>crmReg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0.1</td>
</tr>
<tr>
<td>Date:</td>
<td>2020-03-26</td>
</tr>
<tr>
<td>License:</td>
<td>GPL (&gt;=2)</td>
</tr>
</tbody>
</table>

The crmReg package provides the implementation of the Cellwise Robust M-regression (CRM) algorithm (Filzmoser et al., 2020) and the SPArse DIrections of Maximal Outlyingness (SPADIMO) algorithm (Debruyne et al., 2019). The package also includes a predict function for fitted CRM regression models, a function for creating heatmaps of cellwise outliers, and a data preprocessing function for centering and scaling the data as used by CRM.

Given an observation that has been detected as an outlier, SPADIMO (Debruyne et al., 2019) finds the subset of variables contributing most the outlier’s outlyingness. Here, the outlyingness of a data point is defined as its robust Mahalanobis distance. The relevant variables are found by checking the direction in which the observation is most outlying. SPADIMO estimates this direction of maximal outlyingness in a sparse manner. Thereby, the method helps to understand in which way an outlier lies out.

The SPADIMO algorithm allows us to introduce the cellwise robust M-regression (CRM) estimator (Filzmoser et al., 2020) as a linear regression estimator that intrinsically yields both a map of cellwise outliers consistent with the linear model, and a vector of regression coefficients that is robust against vertical outliers and leverage points. As a by-product, the method yields a weighted and im-
The package contains five main functions.

The function `spadimo` computes the sparse directions of maximal outlyings of a given observation and shows diagnostic plots for analyzing that observation.

The function `crm` fits a cellwise robust M-regression estimator. Besides a vector of regression coefficients, the function returns an imputed data set that contains estimates of what the values in cellwise outliers would need to amount to if they had fit the model. The output of `crm` is a list object of class "crm".

The function `predict.crm` obtains predictions from a fitted object of class "crm".

The function `cellwiseheatmap` makes a heatmap of cellwise outliers which are typically the result of a call to the `crm` function.

The function `daprpr` preprocesses the data by classical or robust centering and scaling.

**Author(s)**

Peter Filzmoser, Sebastiaan Hoppen, Irene Ortner, Sven Serneels, and Tim Verdonck

Maintainer: Sebastiaan Hoppen <sebastiaan.hoppner@kuleuven.be>

**References**


**See Also**

`crm`, `spadimo`, `predict.crm`, `cellwiseheatmap`, `daprpr`

**Examples**

```r
library(crmReg)
data(topgear)

# get case weights from a robust estimator (covMCD function in robustbase package):
MCD <- robustbase::covMcd(topgear, alpha = 0.5)

# SPADIMO with diagnostic plots:
# Example 1:
Peugeot <- spadimo(data = topgear,
                      weights = MCD$mcd.wt,
                      obs = which(rownames(topgear) == "Peugeot 107"))

# check the plots!
# individual variable names contributing most to Peugeot 107's outlyingness:
print(Peugeot$outlvars)
```
# sparse direction of maximal outlyingness with eta = Peugeot$eta:
print(Peugeot$a)

# default SPADIMO control parameters:
print(Peugeot$control)

# Example 2:
Bugatti <- spadimo(data = topgear,
    weights = MCD$mcd.wt,
    obs = which(rownames(topgear) == "Bugatti Veyron"),
    control = list(stopearly = TRUE, trace = TRUE, plot = TRUE))

# check the plots!
# individual variable names contributing most to Bugatti Veyron's outlyingness:
print(Bugatti$outlvars)

# sparse direction of maximal outlyingness with eta = Bugatti$eta:
print(Bugatti$a)

# fit Cellwise Robust M-regression:
crmfit <- crm(formula = MPG ~ ., data = topgear)

# estimated regression coefficients and detected casewise outliers:
print(crmfit$coefficients)
print(rownames(topgear)[which(crmfit$casewiseoutliers)])

# fitted response values (MPG) versus true response values:
plot(topgear$MPG, crmfit$fitted.values, xlab = "True MPG", ylab = "Fitted MPG")
abline(a = 0, b = 1)

# residuals:
plot(crmfit$residuals, ylab = "Residuals")
text(x = which(crmfit$residuals > 30), y = crmfit$residuals[which(crmfit$residuals > 30)],
    labels = rownames(topgear)[which(crmfit$residuals > 30)], pos = 2)

print(cbind.data.frame(car = rownames(topgear),
    MPG = topgear$MPG)[which(crmfit$residuals > 30), ])

# cellwise heatmap of casewise outliers:
cellwiseheatmap(cellwiseoutliers = crmfit$cellwiseoutliers[which(crmfit$casewiseoutliers), ],
    data = round(topgear[which(crmfit$casewiseoutliers), -7], 2),
    col.scale.factor = 1/4)

# check the plotted heatmap!

---

cellwiseheatmap  
Heatmap of cellwise outliers

description

Makes a heatmap of cellwise outliers.
Usage

cellwiseheatmap(cellwiseoutliers, data,  
  col = c("blue", "lightgray", "red"), col.scale.factor = 1,  
  notecol.outlier = "white", notecol.clean = "black", notecex = 1,  
  margins = c(9.5, 14), lhei = c(0.5, 15), lwid = c(0.1, 3.5),  
  sepcolor = "white", sepwidth = c(0.01, 0.01))

Arguments

cellwiseoutliers
  a matrix that indicates the cellwise outliers as the (scaled) difference between the  
  original data and imputed data, both scaled and centered. Typically the result of  
  a call to the crm function.
data
  the data as a data frame that is shown in the cells, including row and column  
  names.
col
  vector of colors used for downward outliers, clean cells and upward outliers  
  respectively (default is c("blue","lightgray","red")).
col.scale.factor
  numeric factor for scaling the colors of the cells (default is 1). Usually a value  
  between 0 and 1, e.g. 1/2, 1/3, etc.
notecol.outlier
  character string specifying the color for cellnote text of cellwise outliers (default  
  is "white").
notecol.clean
  character string specifying the color for cellnote text of clean cells (default is  
  "black").
notecex
  numeric scaling factor for cellnotes (default is 1).
margins
  numeric vector of length 2 containing the margins (see par(mar= *)) for column  
  and row names, respectively (default is c(9.5,14)).
lhei
  numeric vector of length 2 containing the row height (default is c(1,15)).
lwid
  numeric vector of length 2 containing the row width (default is c(0.7,3.5)).
sepcolor
  character string specifying the color between the cells (default is "white").
sepwidth
  vector of length 2 giving the width and height of the separator box drawn be-  
  tween the cells (default is c(0.01,0.01)).

Details

cellwiseheatmap plots a heatmap of cellwise outliers which are typically the result of a call to the  
crm function.

Author(s)

Peter Filzmoser, Sebastiaan Hoppen, Irene Ortner, Sven Serneels, and Tim Verdonck

References

Filzmoser, P., Hoppen, S., Ortner, I., Serneels, S., and Verdonck, T. (2020). Cellwise Robust M re-  
gression. Computational Statistics and Data Analysis, 147, 106944. DOI:10.1016/j.csda.2020.106944
crm

See Also

crm, spadimo, predict.crm, daprpr

Examples

library(crmReg)
data(topgear)

# fit Cellwise Robust M-regression:
crmfit <- crm(formula = MPG ~ ., data = topgear)

# cellwise heatmap of casewise outliers:
cellwiseheatmap(cellwiseoutliers = crmfit$cellwiseoutliers[which(crmfit$casewiseoutliers), ],
data = round(topgear[which(crmfit$casewiseoutliers), -7], 2),
col.scale.factor = 1/4)

# check the plotted heatmap!

---

crm

Cellwise Robust M-regression

Description

Fits a cellwise robust M-regression estimator. Besides a vector of regression coefficients, the function returns an imputed data set that contains estimates of what the values in cellwise outliers would need to amount to if they had fit the model.

Usage

crm(formula, data, maxiter = 100, tolerance = 0.01, outlyingness.factor = 1,
spadieta = seq(0.9, 0.1, -0.1), center = "median", scale = "qn",
regtype = "MM", alphaLTS = NULL, seed = NULL, verbose = TRUE)

Arguments

formula an lm-style formula object specifying which relationship to estimate.
data the data as a data frame.
maxiter maximum number of iterations (default is 100).
tolerance obtain optimal regression coefficients to within a certain tolerance (default is 0.01).
outlyingness.factor numeric value, larger or equal to 1 (default). Only cells are altered of cases for which the original outlyingness (before SPADIMO) is larger than outlyingness.factor * outlyingness AFTER SPADIMO. The larger this factor, the fewer cells are imputed.
spadieta the sparsity parameter to start internal outlying cell detection with, must be in the range [0.1] (default is seq(0.9, 0.1, -0.1)).
**center**  
how to center the data. A string that matches the R function to be used for centering (default is "median").

**scale**  
how to scale the data. Choices are "no" (no scaling) or a string matching the R function to be used for scaling (default is "qn").

**regtype**  
type of robust regression. Choices are "MM" (default) or "LTS".

**alphaLTS**  
parameter used by LTS regression. The percentage (roughly) of squared residuals whose sum will be minimized (default is .5).

**seed**  
initial seed for random generator, like .Random.seed (default is NULL).

**verbose**  
should output be shown during the process (default is TRUE).

**Details**

The cellwise robust M-regression (CRM) estimator (Filzmoser et al., 2020) is a linear regression estimator that intrinsically yields both a map of cellwise outliers consistent with the linear model, and a vector of regression coefficients that is robust against vertical outliers and leverage points. As a by-product, the method yields a weighted and imputed data set that contains estimates of what the values in cellwise outliers would need to amount to if they had fit the model. The CRM method consists of an iteratively reweighted least squares procedure where SPADIMO is applied at each iteration to detect the cells that contribute most to outlyingness. As such, CRM detects deviating data cells consistent with a linear model.

**Value**

.crm returns a list object of class "crm" containing the following elements:

- **coefficients**: a named vector of fitted coefficients.
- **fitted.values**: the fitted response values.
- **residuals**: the residuals, that is response minus fitted values.
- **weights**: the (case) weights of the residuals.
- **data.imputed**: the data as imputed by CRM.
- **casewiseoutliers**: a vector that indicates the casewise outliers with TRUE or FALSE.
- **cellwiseoutliers**: a matrix that indicates the cellwise outliers as the (scaled) difference between the original data and imputed data, both scaled and centered.
- **terms**: the terms object used.
- **call**: the matched call.
- **inputs**: the list of supplied input arguments.
- **numloops**: the number of iterations.
- **time**: the number of seconds passed to execute the CRM algorithm.

**Author(s)**

Peter Filzmoser, Sebastiaan Hoppner, Irene Ortner, Sven Serneels, and Tim Verdonck
References


See Also

spadimo.predict.crm, cellwiseheatmap, daprpr

Examples

```r
library(crmReg)
data(topgear)

# fit Cellwise Robust M-regression:
crmfit <- crm(formula = MPG ~ ., data = topgear)

# estimated regression coefficients and detected casewise outliers:
print(crmfit$coefficients)
print(rownames(topgear)[which(crmfit$casewiseoutliers)])

# fitted response values (MPG) versus true response values:
plot(topgear$MPG, crmfit$fitted.values, xlab = "True MPG", ylab = "Fitted MPG")
abline(a = 0, b = 1)

# residuals:
plot(crmfit$residuals, ylab = "Residuals")
text(x = which(crmfit$residuals > 30), y = crmfit$residuals[which(crmfit$residuals > 30)],
     labels = rownames(topgear)[which(crmfit$residuals > 30)], pos = 2)

print(cbind.data.frame(car = rownames(topgear),
                        MPG = topgear$MPG)[which(crmfit$residuals > 30),])

# cellwise heatmap of casewise outliers:
cellwiseheatmap(cellwiseoutliers = crmfit$cellwiseoutliers[which(crmfit$casewiseoutliers),],
                data = round(topgear[which(crmfit$casewiseoutliers), -7], 2),
                col.scale.factor = 1/4)

# check the plotted heatmap!
```

---

**daprpr**

*Data Preprocessing*

**Description**

Data preprocessing, classical and robust centering and scaling.

**Usage**

```r
daprpr(Data, center.type, scale.type)
```
Arguments

- **Data**
  - the data.
- **center.type**
  - type of centering as R function name (e.g. "mean", "median", "l1median").
- **scale.type**
  - type of scaling as R function name (e.g. "sd", "qn", "Sn", "scaleTau2").

Details

daprpr preprocesses the data by classical or robust centering and scaling. Given `center.type = "mean"` and `scale.type = "sd"`, function `daprpr` is equivalent to `scale(Data, center = TRUE, scale = TRUE).

Value

daprpr returns the scaled data with attributes "Center", "Scale" and "Type".

Author(s)

Sven Serneels

See Also

crm, spadimo, predict.crm, cellwiseheatmap

Examples

```r
library(crmReg)
data(topgear)

topgear_centered_scaled <- daprpr(topgear, center.type = "median", scale.type = "qn")

boxplot(topgear_centered_scaled)
attributes(topgear_centered_scaled)$Type
attributes(topgear_centered_scaled)$Center
attributes(topgear_centered_scaled)$Scale
```

**predict.crm**

*Predict method for CRM fits*

Description

Obtains predictions from a fitted `crm` object.

Usage

```r
## S3 method for class 'crm'
predict(object, newdata = NULL, ...)
```
Arguments

- object: a fitted object of class "crm".
- newdata: optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted coefficients are used.
- ...: further arguments passed to or from other methods.

Details

`predict.crm` produces predicted values, obtained by evaluating the fitted `crm` object on the data frame `newdata`.

Value

`predict.crm` returns a vector of predicted response values.

Author(s)

Peter Filzmoser, Sebastiaan Hoppner, Irene Ortner, Sven Serneels, and Tim Verdonck

References


See Also

`crm`, `spadimo`, `cellwiseheatmap`, `daprpr`

Examples

```r
library(crmReg)
data(topgear)

train <- topgear[1:200, ]
test <- topgear[201:245, ]
crmfit <- crm(formula = MPG ~ ., data = train, seed = 2020)
estimated_MPG_test <- predict(crmfit, newdata = test)

plot(test$MPG, estimated_MPG_test, xlab = "True MPG", ylab = "Estimated MPG")
abline(a = 0, b = 1)
```
Description

Computes the sparse directions of maximal outlyings of a given observation and shows diagnostic plots for analyzing that observation.

Usage

```
spadimo(data, weights, obs,
control = list(scaleFun = Qn, nlatent = 1, etas = NULL, csqcritv = 0.975,
        stopearly = FALSE, trace = FALSE, plot = TRUE))
```

Arguments

data the data as a data frame.
weights a numeric vector containing the case weights from a robust estimator.
obs the (integer) case number under consideration.
control a list of options that control details of the crm algorithm. The following options are available:
  • `scaleFun` function used for robust scaling the variables (e.g. Qn, mad, etc.).
  • `nlatent` integer number of latent variables for sparse PLS regression (via SNIPLS) (default is 1).
  • `etas` vector of decreasing sparsity parameters (default is NULL in which case etas = seq(0.9, 0.1, -0.05) if n > p, otherwise etas = seq(0.6, 0.1, -0.05)).
  • `csqcritv` probability level for internal chi-squared quantile (used when n > p) (default is 0.975).
  • `stopearly` if TRUE, method stops as soon as the reduced case is no longer outlying, else if FALSE (default) it loops through all values of eta.
  • `trace` should intermediate results be printed (default is FALSE).
  • `plot` should heatmaps and graph of the results be shown (default is TRUE).

Details

Given an observation that has been detected as an outlier, SPADIMO (Debruyne et al., 2019) finds the subset of variables contributing most the outlier’s outlyingness. Here, the outlyingness of a data point is defined as its robust Mahalanobis distance. The relevant variables are found by checking the
direction in which the observation is most outlying. SPADIMO estimates this direction of maximal outlyingness in a sparse manner. Thereby, the method helps to understand in which way an outlier lies out.

**Value**

`spadimo` returns a list containing the following elements:

- `outlvars`: vector containing individual variable names contributing most to `obs`’s outlyingness.
- `outlvarslist`: list of variables contributing to `obs`’s outlyingness for different values of `eta`.
- `a`: vector, the sparse direction of maximal outlyingness.
- `alist`: list of sparse directions of maximal outlyingness for different values of `eta`.
- `o.before`: outlyingness of original case (`n < p`) or PCA outlier flag (`n >= p`) before removing outlying variables.
- `o.after`: outlyingness of reduced case (`n > p`) or PCA outlier flag (`n >= p`) after removing outlying variables.
- `eta`: cutoff where `obs` is no longer outlying.
- `time`: time to execute the SPADIMO algorithm.
- `control`: a list with control parameters that are used.

**Author(s)**

Michiel Debruyne, Sebastiaan Hoppner, Sven Serneels, and Tim Verdonck

**References**


**See Also**

`crm`, `predict.crm`, `cellwiseheatmap`, `daprpr`

**Examples**

```r
library(crmReg)
data(topgear)

# get case weights from a robust estimator (covMCD function in robustbase package):
MCD <- robustbase::covMcd(topgear, alpha = 0.5)

# SPADIMO with diagnostic plots:
# Example 1:
Peugeot <- spadimo(data = topgear,
                   weights = MCD$mcd.wt,
                   obs = which(rownames(topgear) == "Peugeot 107"))

# check the plots!
# individual variable names contributing most to Peugeot 107’s outlyingness:
```
print(Peugeot$outlvars)
# sparse direction of maximal outlyingness with eta = Peugeot$eta:
print(Peugeot$a)
# default SPADIMO control parameters:
print(Peugeot$control)

# Example 2:
Bugatti <- spadimo(data = topgear,
                   weights = MCD$mcd.wt,
                   obs = which(rownames(topgear) == "Bugatti Veyron"),
                   control = list(stopearly = TRUE, trace = TRUE, plot = TRUE))
# check the plots!
# individual variable names contributing most to Bugatti Veyron's outlyingness:
print(Bugatti$outlvars)
# sparse direction of maximal outlyingness with eta = Bugatti$eta:
print(Bugatti$a)

topgear

*Top Gear car data*

**Description**

The data set contains information on cars featured on the website of the popular BBC television show *Top Gear*. The original, full data set is available in the package robustHD.

**Usage**

data(topgear)

**Format**

A data frame containing 245 observations and 11 variables.

- `log(Price)` the natural logarithm of the list price (in UK pounds).
- `log(Displacement)` the natural logarithm of the displacement of the engine (in cc).
- `log(BHP)` the natural logarithm of the power of the engine (in bhp).
- `log(Torque)` the natural logarithm of the torque of the engine (in lb/ft).
- `Acceleration` the time it takes the car to get from 0 to 62 mph (in seconds).
- `log(TopSpeed)` the natural logarithm of the car's top speed (in mph).
- `MPG` the combined fuel consumption (urban + extra urban; in miles per gallon).
- `Weight` the car's curb weight (in kg).
- `Length` the car's length (in mm).
- `Width` the car's width (in mm).
- `Height` the car's height (in mm).
Source

The original data set is available in the package robustHD. The data were scraped from http://www.topgear.com/uk/ on 2014-02-24.

Examples

data(topgear)
str(topgear)
head(topgear)
summary(topgear)
Index

* datasets
  topgear, 13
  .Random.seed, 7

cellwiseheatmap, 3, 4, 8–10, 12
crm, 3, 5, 6, 6, 9, 10, 12
crmReg (crmReg-package), 2
crmReg-package, 2
daprpr, 3, 6, 8, 8, 10, 12
par, 5
predict.crm, 3, 6, 8, 9, 9, 12
spadimo, 3, 6, 8–10, 11
topgear, 13