Package ‘hypercube’

October 13, 2022

Type Package

Title Organizing Data in Hypercubes

Version 0.2.1

Author Michael Scholz

Maintainer Michael Scholz <michael.scholz@th-deg.de>

Description Provides functions and methods for organizing data in hypercubes
(i.e., a multi-dimensional cube). Cubes are generated from molten data frames.
Each cube can be manipulated with five operations: rotation (change.dimensionOrder()),
dicing and slicing (add.selection(), remove.selection()), drilling down (add.aggregation()),
and rolling up (remove.aggregation()).

License GPL-3

Encoding UTF-8

Depends R (>= 3.3.0), stats, plotly

Imports methods, stringr, dplyr

LazyData TRUE

RoxygenNote 7.0.2

NeedsCompilation no

Repository CRAN

Date/Publication 2020-02-28 07:10:08 UTC

R topics documented:

`hypercube-package` *Provides functions and methods for organizing data in hypercubes*

Description

This package provides methods for organizing data in a hypercube Each cube can be manipulated with five operations rotation (changeDimensionOrder), dicing and slicing (add.selection, remove.selection), drilling down (add.aggregation), and rolling up (remove.aggregation).

Details

Package:	hypercube
Type:	Package
Version:	0.2.1
Date:	2020-02-27
License:	GPL-3
Depends:	R (>= 3.0), methods

Author(s)

Michael Scholz <`michael.scholz@th-deg.de`>

Examples

```
# Simple example
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube

# More sophisticated example
data("sales")
```

```
cube = generateCube(sales, columns = list(time = c("month", "year"),
       location = c("state"), product = "product"), valueColumn = "amount")
cube = add.selection(cube, criteria = list(state = c("AL", "TX")))
cube = add.aggregation(cube, dimensions = c("month", "year"), fun = "sum")
cube
df = as.data.frame(cube)
df
```

`add.aggregation` *Adds an aggregation to a hypercube*

Description

This function adds a further aggregation to a hypercube. The cube itself will not be changed. The aggregation only affect the data that will be shown when printing the cube. Note that selection criteria will be applied before aggregating the data.

Usage

```
add.aggregation(
  x,
  dimensions,
  fun = c("sum", "min", "max", "prod", "mean", "median", "sd", "count")
)
```

Arguments

`x`	Hypercube for which the selection criteria will be defined.
`dimensions`	A vector of dimensions that are used in the aggregation.
`fun`	The function that is used for aggregation. Possible functions are sum, prod, min, max, mean, median, sd, and count.

Value

Returns a `Cube` object with the added aggregation.

Author(s)

Michael Scholz <`michael.scholz@th-deg.de`>

See Also

`Cube` `remove.aggregation` `add.selection`

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube = add.aggregation(cube, dimensions = c("month", "year"), fun = "sum")
cube
```

---

add.selection *Adds selection criteria to a hypercube*

Description

This function adds further selection criteria to a hypercube. The cube itself will not be changed. The selection criteria only affect the data that will be shown when printing the cube. Note that selection criteria will be applied before aggregating the data.

Usage

```
add.selection(x, criteria)
```

Arguments

`x`	Hypercube for which the selection criteria will be defined.
`criteria`	A list of selection criteria.

Value

Returns a `Cube` object with the added selection criteria.

Author(s)

Michael Scholz <`michael.scholz@th-deg.de`>

See Also

`Cube` `remove.selection` `add.aggregation`

Examples

```
data("sales")
print(str(sales))
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube = add.selection(cube, criteria = list(state = c("CA", "FL")))
cube
```

```
cube = add.selection(cube, criteria = list(state = c("TX")))
cube
```

`as.data.frame.Cube` *Converts the actual view of a cube to a data frame*

Description

Converts the actual view of a `Cube` object to a data frame. All added selections and aggregations will be regarded. Note that selection criteria will be applied before aggregating the data.

Usage

```
## S3 method for class 'Cube'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	The `Cube` object that will be converted to a data frame.
`row.names`	A character vector giving the row names for the data frame.
`optional`	Should setting row names and converting column names be optional?
`...`	Further parameters that are passed to `as.data.frame.table`.

Value

A molten data frame

Author(s)

Michael Scholz <`michael.scholz@th-deg.de`>

See Also

`add.aggregation` `add.selection`

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube = change.dimensionOrder(cube, dimensions = c("product", "month", "year", "state"))
df = as.data.frame(cube)
df
```

`change.dimensionOrder` *Changes the order of the dimensions in a given cube*

Description

Changes the order of the dimensions in a given cube

Usage

```
change.dimensionOrder(x, dimensions)
```

Arguments

`x`	Hypercube for which the dimensions should be re-ordered.
`dimensions`	Vector of dimensions. The order of the dimensions in this vector defines the order of the dimensions in the cube.

Value

Returns a `Cube` object.

Author(s)

Michael Scholz `<michael.scholz@th-deg.de>`

See Also

`Cube`

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube = change.dimensionOrder(cube, dimensions = c("product", "month", "year", "state"))
cube
```

Cube-class *Class* "Cube"

Description

Class "Cube"

Slots

data (array) The data that are represented as hypercube.

structure (list) The structure of the dimensions of the hypercube.

view (list) Information about how to build a view for the hypercube. This information is stored in a list of Dimension-class objects.

Objects from the Class

Objects can be created by calls of the form new("Cube", ...). This S4 class describes Cube objects.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

generateCube

Examples

```
# show Cube definition
showClass("Cube")
```

Dimension-class *Class* "Cube"

Description

Class "Cube"

Slots

name (character) The name of the dimension.

values (vector) A vector of selected values for this dimension.

aggregation (vector) A vector of aggregation functions that will be applied to this dimension.

Objects from the Class

Objects can be created by calls of the form `new("Dimension", ...)`. This S4 class describes `Dimension` objects.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

Examples

```
# show Dimension definition
showClass("Dimension")
```

---

generateCube — *Generates a hypercube from a given dataframe*

Description

This function generates a hypercube from a given dataframe. The dimensions of the hypercube correspond to a set of selected columns from the dataframe.

Usage

```
generateCube(
  data,
  columns,
  valueColumn,
  fun = c("sum", "min", "max", "prod", "mean", "median", "sd", "count")
)
```

Arguments

`data`	A dataframe that is used as source for the hypercube.
`columns`	A vector of column names that will form the dimensions of the hypercube.
`valueColumn`	The name of the column that provides the values for the cells of the hypercube.
`fun`	Aggregation function for aggregating over those columns that do not correspond with any dimension of the hypercube.

Value

Returns a `Cube` object.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

Cube

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
```

---

importance — *Calculates the dimension importances of a given cube.*

Description

Calculates the importance values for all dimensions of the actual view of a Cube object. All added selections and aggregations will be regarded. Note that selection criteria will be applied before aggregating the data.

Usage

```
importance(x)
```

Arguments

x	The Cube object for which the importance values will be computed.

Value

Sparsity value

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

sparsity

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
importance(cube)
```

plot,Cube-method *Visualizes a Cube object as parallel coordinate plot*

Description

Generates a parallel coordinate plot for a given Cube object. All added selections and aggregations will be regarded.

Usage

```
## S4 method for signature 'Cube'
plot(x, color = NA, colorscale = "RdBu", ...)
```

Arguments

x	The Cube object that should be plotted.
color	The color of the lines in the parallel coordinate plot. If this parameter is NA or NULL, a colorscale rather than a unique color will be used.
colorscale	The colorscale for the lines in the parallel coordinate plot. Default is RdBu. All plotly colorscales (e.g., Blackbody, Earth, Jet) are possible.
...	Further plot_ly parameters.

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

Cube

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
plot(cube)
```

`print.Importances` *Prints an Importances object.*

Description

Prints an `Importances` object.

Usage

```
## S3 method for class 'Importances'
print(x, ...)
```

Arguments

`x`	The `Importances` object that will be printed.
`...`	Ignored parameters.

Value

Sparsity value

Author(s)

Michael Scholz <`michael.scholz@th-deg.de`>

See Also

`importance`

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
importances = importance(cube)
print(importances)
```

`remove.aggregation` *Removes aggregations from a hypercube*

Description

This function removes aggregations from a hypercube. The cube itself will not be changed. The aggregation only affect the data that will be shown when printing the cube.

Usage

```
remove.aggregation(x, dimensions = NA, last = FALSE)
```

Arguments

`x`	Hypercube from which the aggregation will be removed.
`dimensions`	A vector of dimensions for which the aggregations will be removed.
`last`	Should the last aggregation be removed? If this parameter is set TRUE, the dimension vector will be ignored.

Value

Returns a `Cube` object with the added aggregation.

Author(s)

Michael Scholz <`michael.scholz@th-deg.de`>

See Also

`Cube` `add.aggregation` `remove.selection`

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube = add.aggregation(cube, dimensions = c("month", "year"), fun = "sum")
cube
cube = add.aggregation(cube, dimensions = "year", fun = "sum")
cube
cube = remove.aggregation(cube, dimensions = "year")
cube
```

`remove.selection` *Removes selection criteria from a hypercube*

Description

This function removes all selection criteria for the given dimensions. The cube itself will not be changed. The selection criteria only affect the data that will be shown when printing the cube.

Usage

```
remove.selection(x, dimensions)
```

Arguments

`x`	Hypercube for which the selection criteria will be defined.
`dimensions`	A vector of dimension names for which all selection criteria will be removed.

Value

Returns a `Cube` object with removed selection criteria.

Author(s)

Michael Scholz <`michael.scholz@th-deg.de`>

See Also

`Cube` `add.selection` `remove.aggregation`

Examples

```
data("sales")
print(str(sales))
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube = add.selection(cube, criteria = list(state = c("CA", "FL")))
cube
cube = remove.selection(cube, dimensions = c("state"))
cube
```

`sales` *Sales of books*

Description

A dataset containing 2,500 sales of 4 books in different states and countries.

Usage

```
sales
```

Format

A data fram with 2500 rows and 7 variables:

month month as number

year year as number

state abbreviation of the state as character

country country as character

product name of the product as character

unit number of sold products

amount amount of sales

Source

Synthetic dataset

`show,Cube-method` *Shows a Cube object*

Description

Shows the actual view of a `Cube` object. All added selections and aggregations will be regarded. Note that selection criteria will be applied before aggregating the data.

Usage

```
## S4 method for signature 'Cube'
show(object)
```

Arguments

`object`	The `Cube` object

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

Cube

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube
```

---

show,Dimension-method *Shows a* Dimension *object*

Description

Shows a Dimension object

Usage

```
## S4 method for signature 'Dimension'
show(object)
```

Arguments

object	The Dimension object

Author(s)

Michael Scholz <michael.scholz@th-deg.de>

See Also

Cube

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
cube@view[[1]]
```

`sparsity` *Calculates the sparsity of a given cube.*

Description

Calculates the sparsity of the actual view of a `Cube` object. All added selections and aggregations will be regarded. Note that selection criteria will be applied before aggregating the data.

Usage

```
sparsity(x)
```

Arguments

`x`	The `Cube` object for which the sparsity will be computed.

Value

Sparsity value

Author(s)

Michael Scholz `<michael.scholz@th-deg.de>`

See Also

`importance`

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
sparsity(cube)
```

`summary`	*Shows a summary for the given cube*

Description

Shows the dimensions and the number of levels per dimension of the given cube. All added selections and aggregations will be regarded.

Usage

```
summary(x)
```

Arguments

`x`	The `Cube` object for which the summary is shown.

Author(s)

Michael Scholz <`michael.scholz@th-deg.de`>

See Also

`Cube`

Examples

```
data("sales")
cube = generateCube(sales, columns = list(time = c("month", "year"),
      location = c("state"), product = "product"), valueColumn = "amount")
summary(cube)
```

Index