

# Package ‘hypercube’

September 18, 2019

**Type** Package

**Title** Organizing Data in a Hypercube

**Version** 0.2.0

**Author** Michael Scholz

**Maintainer** Michael Scholz <michael.scholz@uni-passau.de>

**Description** Provides methods for organizing data in a hypercube (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** 6.1.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-09-18 19:20:02 UTC

## R topics documented:

hypercube-package . . . . .	2
add.aggregation . . . . .	3
add.selection . . . . .	4
as.data.frame.Cube . . . . .	5
change.dimensionOrder . . . . .	6
Cube-class . . . . .	7
Dimension-class . . . . .	7
generateCube . . . . .	8
importance . . . . .	9

plot,Cube-method . . . . .	10
print.Importances . . . . .	11
remove.aggregation . . . . .	12
remove.selection . . . . .	13
sales . . . . .	14
show,Cube-method . . . . .	14
show,Dimension-method . . . . .	15
sparsity . . . . .	16

<b>Index</b>	<b>17</b>
--------------	-----------

---

hypercube-package	<i>Provides methods for organizing data in a hypercube</i>
-------------------	--

---

## 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:	0.2.0
Date:	2019-09-18
License:	GPL-3
Depends:	R (>= 3.0), methods

## Author(s)

Michael Scholz <michael.scholz@uni-passau.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@uni-passau.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	<i>Adds selection criteria to a hypercube</i>
---------------	---

---

### 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@uni-passau.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 <a href="#">as.data.frame.table</a> .

### Value

A molten data frame

### Author(s)

Michael Scholz <michael.scholz@uni-passau.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@uni-passau.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@uni-passau.de](mailto:michael.scholz@uni-passau.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@uni-passau.de>

### Examples

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

---

generateCube	<i>Generates a hypercube from a given dataframe</i>
--------------	---

---

### 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)
```

### 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.

### Value

Returns a Cube object.

### Author(s)

Michael Scholz <michael.scholz@uni-passau.de>

### See Also

[Cube](#)



## Examples

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

---

importance	<i>Calculates the dimension importances of a given cube.</i>
------------	--

---

## 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@uni-passau.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	<i>Visualizes a Cube object as parallel coordinate plot</i>
------------------	---

---

**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@uni-passau.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	<i>Prints an Importances object.</i>
-------------------	--------------------------------------

---

### 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@uni-passau.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@uni-passau.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	<i>Removes selection criteria from a hypercube</i>
------------------	--

---

### 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@uni-passau.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	<i>Sales of books</i>
-------	-----------------------

---

**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	<i>Shows a Cube object</i>
-------------------	----------------------------

---

**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@uni-passau.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@uni-passau.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	<i>Calculates the sparsity of a given cube.</i>
----------	---

---

**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@uni-passau.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)
```



# Index

- \*Topic **classes**
  - Cube-class, [7](#)
  - Dimension-class, [7](#)
- \*Topic **datasets**
  - sales, [14](#)
- \*Topic **manip**
  - hypercube-package, [2](#)
- \*Topic **methods**
  - add.aggregation, [3](#)
  - add.selection, [4](#)
  - change.dimensionOrder, [6](#)
  - generateCube, [8](#)
  - importance, [9](#)
  - plot, Cube-method, [10](#)
  - remove.aggregation, [12](#)
  - remove.selection, [13](#)
  - sparsity, [16](#)
- add.aggregation, [3](#), [4](#), [5](#), [12](#)
- add.aggregation, Cube-method  
(add.aggregation), [3](#)
- add.selection, [3](#), [4](#), [5](#), [13](#)
- add.selection, Cube-method  
(add.selection), [4](#)
- as.data.frame.Cube, [5](#)
- as.data.frame.table, [5](#)
- change.dimensionOrder, [6](#)
- change.dimensionOrder, Cube-method  
(change.dimensionOrder), [6](#)
- Cube, [3](#), [4](#), [6](#), [8](#), [10](#), [12](#), [13](#), [15](#)
- Cube-class, [7](#)
- Dimension-class, [7](#)
- generateCube, [7](#), [8](#)
- hypercube (hypercube-package), [2](#)
- hypercube-package, [2](#)
- importance, [9](#), [11](#), [16](#)
- importance, Cube-method (importance), [9](#)
- plot, Cube-method, [10](#)
- print.Importances, [11](#)
- remove.aggregation, [3](#), [12](#), [13](#)
- remove.aggregation, Cube-method  
(remove.aggregation), [12](#)
- remove.selection, [4](#), [12](#), [13](#)
- remove.selection, Cube-method  
(remove.selection), [13](#)
- sales, [14](#)
- show, Cube-method, [14](#)
- show, Dimension-method, [15](#)
- sparsity, [9](#), [16](#)
- sparsity, Cube-method (sparsity), [16](#)