Package ‘cdparcoord’

August 5, 2019

Version  1.0.1
Author  Norm Matloff <normmatloff@gmail.com> and
       Vincent Yang <vinyang@ucdavis.edu> and Harrison Nguyen <hhnguy@ucdavis.edu>
Maintainer  Norm Matloff <normmatloff@gmail.com>
Date  2019-08-04
Title  Top Frequency-Based Parallel Coordinates
Description  Parallel coordinate plotting with resolutions for large data sets
             and missing values.
Depends  R (>= 3.2.0), data.table, plotly, freqparcoord, partools
Imports
Suggests  rmarkdown, knitr
VignetteBuilder  knitr
LazyLoad  no
License  GPL (>= 2)
NeedsCompilation  no
Date/Publication  2019-08-04 23:00:02 UTC
Repository  CRAN

R topics documented:

cdparcoord .................................................. 2
clsTupleFreqs ........................................... 2
demog ..................................................... 5
discretize ................................................. 5
hrdata .................................................... 6
makeFactor ................................................. 7
quickstart ................................................. 7
reOrder .................................................... 8
showCounts ............................................... 9
smallexample ............................................ 9
Titanic .................................................... 9
Description

A novel approach to the parallel coordinates method for visualization of multiple variables at once, focused on discrete and categorical variables.

(a) Addresses the screen-clutter problem in parallel coordinates, by only plotting the "most typical" cases. These are the tuples with the highest occurrence rates.

(b) Provides a novel approach to NA values by allowing tuples with NA values to partially contribute to complete tuples rather than eliminating missing values.

Type \texttt{?quickstart} for a quick start.

Author(s)

Norm Matloff <matloff@cs.ucdavis.edu>, Vincent Yang <vinyang@ucdavis.edu>, and Harrison Nguyen <hhnguy@ucdavis.edu>

Usage

tupleFreqs(dataset,k=5,NAexp=1.0,countNAs=FALSE,saveCounts=FALSE, minFreq=NULL,accentuate=NULL,accval=100)
clsTupleFreqs(cls=NULL, dataset, k=5, NAexp=1, countNAs=FALSE)
discparcoord(data, k=5, grpcategory=NULL, permute=FALSE, interactive = TRUE, save=FALSE, name="Parcoords", labelsOff=TRUE, NAexp=1.0,countNAs=FALSE, accentuate=NULL, accval=100, inParallel=FALSE, cls=NULL, differentiate=FALSE, saveCounts=FALSE, minFreq=NULL)
**clsTupleFreqs**

**Arguments**

- **data**: The data, in data frame or matrix form.
- **k**: The number of tuples to return. These will be the k most frequent tuples, unless k is negative, in which case the least-frequent tuples will be returned. The latter is useful for hunting for outliers.
- **grpcategory**: Grouping column/variable.
- **permute**: If TRUE, randomly permute the columns before plotting.
- **interactive**: If TRUE, use interactive plotting, allowing for interactively readjusting column order and scrubbing/brushing.
- **save**: If this is TRUE and interactive mode is on, saved plot will be available from the browser.
- **name**: The name for the plot.
- **labelsOff**: If TRUE, labels are off. This only comes into effect when interactive=FALSE.
- **NAexp**: Scale for NA counts.
- **countNAs**: If TRUE, count NA values.
- **accentuate**: Character expression specifying the property to accentuate.
- **accval**: Value to accentuate.
- **inParallel**: If TRUE, calculate tuple frequencies in parallel.
- **differentiate**: If TRUE, randomize coloring to differentiate overlapping lines.
- **saveCounts**: If TRUE, save the tuple counts to the file ‘tupleCounts’.
- **minFreq**: The smallest frequency to be displayed.
- **dataset**: The dataset to process, a data frame or data.table.
- **cls**: Cluster to be used if inParallel is TRUE. If inParallel is TRUE and cls is not supplied, it will use the sensed number of cores on the calling machine by default.

**Details**

Tuple tabulation is performed by `tupleFreqs`, or in large cases, in parallel by `clsTupleFreqs`. The display is done by `discparcoord`.

The k most- or least-frequent tuples will be reported, with the latter specified via negative k. Optionally, tuples with NA values will count less, but weigh toward everything that has existing numbers in common with it.

If continuous variables are present, then in most cases, either convert to discrete using `discretize` or use `freqparcoord`.

The data will be converted into a data.table if it is not already in that form. For this and other reasons, it is advantageous to have the data in that form to begin with, say by using `data.table:: fread` to read the data.

Optionally, tuples that partially match a full tuple pattern except for NA values will add a partial count to the frequency count for the full pattern. If for instance the data consist of 8-tuples and a row in the data matches a given 8-tuple pattern in 7 of 8 components, this row would add a count of 7/8 to the frequency for that pattern. To reduce this weight, use a value greater than 1.0 for NAexp. If that value is 2, for example, the 7/8 increment will be 7/8 squared.
The functions `tupleFreqs` and `clsTupleFreqs` return an object of class `c('pna', 'data.frame')`, with each row consisting of a tuple and its count. In addition the object will have attributes `k` and `minFreq`.

The function `discparcoord` returns an object of class `c('plotly', 'htmlwidget')`. Printing the object causes display of the graph.

Author(s)

Norm Matloff <matloff@cs.ucdavis.edu>, Vincent Yang <vinyang@ucdavis.edu>, and Harrison Nguyen <hhnguy@ucdavis.edu>

Examples

```r
## Not run:
data(Titanic)
# Find frequencies in parallel
discparcoord(Titanic, inParallel=TRUE)
## End(Not run)

## Not run:
data(hrdata)
input1 = list("name" = "average_monthly_hours",
              "partitions" = 3, "labels" = c("low", "med", "high"))
input = list(input1)
# this will discretize the data by partitioning average monthly
# hours into 3 parts called low, med, and high
hrdata = discretize(hrdata, input)
print('first few discretized tuples')
# first line should be 0.38,0.53,2,low,3,0,1,00,sales,low
head(hrdata)
predicted = predict(hrdata)
print('first few most-frequent tuples')
# first line should be 0.40,0.46,2,...,11
tupleFreqs(hrdata, saveCounts=FALSE)
# account for NA values and plot with parallel coordinates
discparcoord(hrdata)
# same as above, but with scrambled columns
discparcoord(hrdata, permute=TRUE)
# same as above, but show top k values
discparcoord(hrdata, k=8)
# same as above, but group according to profession
discparcoord(hrdata, grpCategory="sales")
## End(Not run)
```
Description

Useful for embeddings. Source is catalog.data.gov.

Discretize continuous data.

Usage

discretize(dataset, input = NULL, ndigs=2, nlevels=10, presumedFactor=FALSE)

Arguments

dataset        Dataset to discretize, data frame/table.
input          Optional specification for partitioning, giving the number of partitions and labels for each partition. List of lists, one list per column to be converted. The outermost list indicates the columns to be converted, and each inner list holds the name of the column, the number of partitions, and a list of labels for each partition.
ndigs          Number of digits to retain in forming labels/values for the discretized data, if input is not supplied. E.g. if ndigs is 2 and the original datum is 38.12, it becomes 38.
nlevels        Number of partitions to form for each variable, if input is NULL.
presumedFactor If TRUE, any variable having fewer than nlevels levels will be presumed to be an informal factor, and thus will not be discretized.

Details

If input is not specified, each numeric column in the data will be discretized, with one exception: If a column is numeric but has fewer distinct values than nlevels, and if presumedFactor is TRUE, it is presumed to be an informal R factor and will not be converted. However, it is best to use makeFactor on such variables.

Author(s)

Norm Matloff <matloff@cs.ucdavis.edu>, Vincent Yang <vinyang@ucdavis.edu>, and Harrison Nguyen <hhnguy@ucdavis.edu>
Examples

data(prgeng)
pe <- prgeng[,c(1,3,5,7:9)] # extract vars of interest
pe25 <- pe[pe$wageinc < 250000,] # delete extreme values
pe25disc <- discretize(pe25) # age, wageinc and wkswrkd discretized

data(mlb)
# extract the height, weight, age, and position of players
m <- mlb[,4:7]

inp1 <- list("name" = "Height",
              "partitions"=4,
              "labels"=c("short", "shortmid", "tallmid", "tall"))

inp2 <- list("name" = "Weight",
              "partitions"=3,
              "labels"=c("light", "med", "heavy"))

inp3 <- list("name" = "Age",
              "partitions"=3,
              "labels"=c("young", "med", "old"))

# create one list to pass everything to discretize()
discreteinput <- list(inp1, inp2, inp3)
head(discreteinput)

# at this point, all of the data has been discretized
discretizedmlb <- discretize(m, discreteinput)
head(discretizedmlb)

hrdata

A human resources simulated dataset.

Description

A small fictional dataset by Kaggle that includes satisfaction level, the result of their last evaluation, number of projects, average monthly hours, time spent at the company, whether they have had a work accident, whether they have had a promotion in the last 5 years, their department, salary and finally whether the employee has left the company. Each row represents a single employee.

Author(s)

Norm Matloff <matloff@cs.ucdavis.edu>, Vincent Yang <vinyang@ucdavis.edu>, and Harrison Nguyen <hhnguy@ucdavis.edu>
**makeFactor**

*Change numeric variables factors.*

**Description**

Change numeric variables that are specified in `varnames` to factors so that `discretize` won’t partition.

**Usage**

```r
makeFactor(df, varnames)
```

**Arguments**

- `df` Input data frame.
- `varnames` Names of variables to be converted to factors.

**Author(s)**

Norm Matloff <matloff@cs.ucdavis.edu>, Vincent Yang <vinyang@ucdavis.edu>, and Harrison Nguyen <hhnguy@ucdavis.edu>

**Examples**

```r
data(prgeng)
pe <- prgeng[,c(1,3,5,7:9)]
class(pe$educ) # integer
pe <- makeFactor(pe,c('educ','occ','sex'))
class(pe$educ) # factor
# nice to give levels names
levels(pe$sex) <- c('male','female')
head(pe$sex)
```

---

**quickstart**

*cdparcoord: Quick start*

**Description**

Quick introduction to the package.
### Examples

```r
# programmer/engineer info from 2000 Census
data(prgeng)
# select some columns of interest
pe <- prgeng[,c(1,3,5,7:9)]
# remove some extreme values
pe25 <- pe[pe$wageinc < 250000,]
# some numeric variables are really factors
pe25 <- makeFactor(pe25,c('educ','occ','sex'))
# convert the continuous variables to discrete
pe25disc <- discretize(pe25,nlevels=5)
## Not run:
# display
discparcoord(pe25disc,k=150)
# then possibly brush, etc.
## End(Not run)
```

### reOrder

**Re-order levels of a factor, according to some desired ordinal form.**

#### Description

Use to order the levels of a factor in a desired sequence.

#### Usage

```r
reOrder(dataset, colName, levelNames)
```

#### Arguments

- **dataset**: Dataset to reorder.
- **colName**: Column name.
- **levelNames**: Names of the reordered levels

#### Author(s)

Norm Matloff <matloff@cs.ucdavis.edu>, Vincent Yang <vinyang@ucdavis.edu>, and Harrison Nguyen <hhnguy@ucdavis.edu>

#### Examples

```r
sl <- c('primary','college','hs','middle','hs')
z <- data.frame(
    schlvl = factor(x=sl,
        levels=c('college','hs','middle','primary'))
)

z <- reOrder(z,'schlvl',c('primary','middle','hs','college'))
str(z) # shows the desired label order in the 'categoryorder' attribute
```
showCounts

---

**showCounts**

*Show tuple counts for the most recent saved counting operation.*

---

**Description**

Used with `saveCounts` TRUE in `tupleFreqs` etc. to recover the tuple counts.

**Usage**

```r
showCounts(nshow=NULL)
```

**Arguments**

- `nshow`: Dataset to show.

**Author(s)**

Norm Matloff <matloff@cs.ucdavis.edu>, Vincent Yang <vinyang@ucdavis.edu>, and Harrison Nguyen <hhnguy@ucdavis.edu>

---

**smallexample**

*A small dataset for showing how `tupleFreqs` works in `cdparcoord`*

---

**Description**

A small fictional dataset with different values and NA's to emphasize `tupleFreqs` and frequency based calculations with `cdparcoord`.

**Author(s)**

Norm Matloff <matloff@cs.ucdavis.edu>, Vincent Yang <vinyang@ucdavis.edu>, and Harrison Nguyen <hhnguy@ucdavis.edu>

---

**Titanic**

*Titanic passengers*

---

**Description**

Famous dataset, source various.
Index

categoricaexample (smallexample), 9
cdparcoord, 2
clsTupleFreqs, 2
demog, 5
discparcoord (clsTupleFreqs), 2
discretize, 3, 5
hrdata, 6
makeFactor, 5, 7
quickstart, 7
reOrder, 8
showCounts, 9
smallexample, 9
Titanic, 9
tupleFreqs (clsTupleFreqs), 2