Package 'HPdata'

February 7, 2014

Type Package

Title Distributed Data Package

Version 0.4.0

Date 2014-01-22

Author Arash Fard

Maintainer Arash Fard <afard@vertica.com>

Depends R (>= 3.0.1), distributedR

Description

It provides several genral functions to load distributed data structures available in distributedR. It is written based on the infrastructure created in HP-Lab for distributed computing in R.

License GPL (>= 2) + file LICENSE

R topics documented:

`HPdata-package` *Distributed Data Package*

Description

HPdata encapsulate all data related functions – data loading, data preparation etc for distributed R environment. It is written based on the infrastructure created in HP-Labs for distributed computing in R.

Details

Package:	HPdata
Type:	Package
Version:	0.4.0
Date:	2014-01-22

Main Functions:

- db2darray: It is an example for loading a set of unlabeled samples stored in a table to a darray.
- db2darrays: It is an example for loading a set of labeled samples stored in a table to two darrays.
- db2matrix: It is an example for loading a set of unlabeled samples stored in a table to a matrix.
- db2dframe: It is an example for loading a set of samples stored in a table to a dframe.
- db2dgraph: It loads an adjaceny matrix to a darray from an edgelist strored in a database.
- file2dgraph: It loads an adjaceny matrix to a darray from an edgelist strored in a set of files.

Author(s)

Arash Fard <afard@vertica.com>

References

1. Using R for Iterative and Incremental Processing. Shivaram Venkataraman, Indrajit Roy, Alvin AuYoung, Rob Schreiber. HotCloud 2012, Boston, USA.
2. http://stat.ethz.ch/R-manual/R-devel/library/stats/html/kmeans.html

---

`db2darray` *A simple loader for darray*

Description

db2darray function is an example for loading a set of unlabeled samples stored in a table to a darray. It is assumed that samples are stored in a single table, and the table contains a column called 'rowid'. It is also assumed that 'rowid' starts from 0, and there is no missed 'rowid'.

Usage

```
db2darray(tableName, features = list(...), conf, nSplits)
```

Arguments

`tableName`	it is the name of the table in the database in string format.
`features`	the list of the name of columns corresponding to the features of a sample.
`conf`	the name of configuration in ODBC.INI file for connecting to the database.
`nSplits`	this optional argument specifies the desired number of splits (partitions) in the distributed array. When it is not specified, it will become equal to the number of active instances in the distributed R environment. It should be noted that the number of splits in the returned result might not be exactly the same as nSplits. Please read the details for more information.

Details

The number of partitions at the result might not be exactly the same as desired number, but it will be close to it. The final number of partitions can be determined by the following routine. When a positive integer number is specified for nSplits, the number of samples in each partition of the distributed structure will be calculated as rowsInBlock <- ceiling(nobs/nSplits) where nobs is the number of observations (number of rows) in the table. As for building the distributed data structure, the value of rowsInBlock will be used, the final number of partitions will be ceiling(nobs/rowsInBlock) which clearly might be different from nSplits.

Value

`X`	the darray of samples

Author(s)

Arash Fard <afard@vertica.com>

Examples

```
## Not run:
   require(distributedR)
   distributedR_start()
   # Assuming that samples are stored in a table named "samples",
   # and the names of the columns are "col1", "col2", "col3", and "col4".
   loadedSamples <- db2darray ("samples", list(col1", "col2",
   "col3", "col4"), conf="RDev")

## End(Not run)
```

`db2darrays` *A Simple Data Loader which loads two aligned darrays from Database*

Description

db2darrays function is a simple function to load a labeled dataset (a set of labeled samples) from a table in Vertica database to a pair of darrays which correspond to responses and predictors of a predictive model. It is assumed that samples (including responses and predictors) are stored in a single table, and the table contains a column called 'rowid'. It is also assumed that 'rowid' starts from 0, and there is no missed 'rowid'.

Usage

```
db2darrays(tableName, resp = list(...), pred = list(...), conf, nSplits)
```

Arguments

`tableName`	it is the name of the table in the database in string format.
`resp`	the list of the name of columns corresponding to responses.
`pred`	the list of the name of columns corresponding to predictors.
`conf`	the name of configuration in ODBC.INI file for connecting to the database.

`nSplits` this optional argument specifies the desired number of splits (partitions) in the distributed arrays. When it is not specified, it will become equal to the number of active instances in the distributed R environment. It should be noted that the number of splits in the returned result might not be exactly the same as nSplits. Please read the details for more information.

Details

The number of partitions at the result might not be exactly the same as desired number, but it will be close to it. The final number of partitions can be determined by the following routine. When a positive integer number is specified for nSplits, the number of samples in each partition of the distributed structure will be calculated as rowsInBlock <- ceiling(nobs/nSplits) where nobs is the number of observations (number of rows) in the table. As for building the distributed data structure, the value of rowsInBlock will be used, the final number of partitions will be ceiling(nobs/rowsInBlock) which clearly might be different from nSplits.

Value

`Y` the darray of responses

`X` the darray of predictors

Author(s)

Arash Fard

Examples

```
## Not run:
   require(distributedR)
   distributedR_start()
   # Assuming that samples are stored in a table named mortgage.
   # The name of response column is def,
   # and the names of predictive columns are mltvspline1, mltvspline2,
   # agespline1, agespline2, hpichgspline, and ficospline.
   loadedData <- db2darrays ("mortgage", list("def"), list("mltvspline1",
   "mltvspline2", "agespline1", "agespline2", "hpichgspline",
   "ficospline"), conf="RDev")

## End(Not run)
```

`db2dframe` *A simple loader for dframe*

Description

db2dframe function is an example for loading a set of samples stored in a table to a dframe. It is assumed that samples are stored in a single table, and the table contains a column called 'rowid'. It is also assumed that 'rowid' starts from 0, and there is no missed 'rowid'.

Usage

```
db2dframe(tableName, features = list(...), conf, nSplits)
```

Arguments

tableName	it is the name of the table in the database in string format.
features	the list of the name of columns corresponding to the features of a sample.
conf	the name of configuration in ODBC.INI file for connecting to the database.
nSplits	this optional argument specifies the desired number of splits (partitions) in the distributed frame. When it is not specified, it will become equal to the number of active instances in the distributed R environment. It should be noted that the number of splits in the returned result might not be exactly the same as nSplits. Please read the details for more information.

Details

The number of partitions at the result might not be exactly the same as desired number, but it will be close to it. The final number of partitions can be determined by the following routine. When a positive integer number is specified for nSplits, the number of samples in each partition of the distributed structure will be calculated as rowsInBlock <- ceiling(nobs/nSplits) where nobs is the number of observations (number of rows) in the table. As for building the distributed data structure, the value of rowsInBlock will be used, the final number of partitions will be ceiling(nobs/rowsInBlock) which clearly might be different from nSplits.

Value

X	the dframe of samples

Author(s)

Arash Fard <afard@vertica.com>

Examples

```
## Not run:
   require(distributedR)
   distributedR_start()
   # Assuming that samples are stored in a table named "samples",
   # and the names of the columns are "col1", "col2", "col3", and "col4".
   loadedSamples <- db2dframe ("samples", list(col1", "col2",
   "col3", "col4"), conf="RDev")

## End(Not run)
```

---

db2dgraph *A simple graph loader from a database*

Description

db2dgraph function is an example for loading an adjacency matrix from a table of edge-list stored in a database. It is assumed that edge-list is stored in a single table. It is also assumed that the ID of vertices in the table starts from zero. As the returned darray will be column-wise partitioned, it would be more efficient when there is projection (index) on the 'to' column of the table.

Usage

```
db2dgraph(tableName, from, to, conf)
```

Arguments

tableName	it is the name of the table in the database in string format.
from	the name of the column in the table which stores the source vertices
to	the name of the column in the table which stores the target vertices
conf	the name of configuration in ODBC.INI file for connecting to the database

Value

X	the darray of adjacency matrix

Author(s)

Arash Fard <afard@vertica.com>

Examples

```
 ## Not run:
    require(HPdgraph)
    distributedR_start()
    # Assuming that edge-list is stored in a table named "graph",
    # and the names of the columns are "x", "y".
    dgraphDB <- db2dgraph("graph", from="x", to="y", conf="RDev")

## End(Not run)
```

---

db2matrix *A simple loader for loading a matrix from a database*

Description

db2matrix function is an example for loading a set of unlabeled samples stored in a table to a matrix. It is assumed that samples are stored in a single table. All the rows of the table will be read, and each row will be a sample.

Usage

```
db2matrix(tableName, features = list(...), conf)
```

Arguments

tableName	it is the name of the table in the database in string format.
features	the list of the name of columns corresponding to the features of a sample.
conf	the name of configuration in ODBC.INI file for connecting to the database.

Value

X	the matrix

Author(s)

Arash Fard <afard@vertica.com>

Examples

```
## Not run:
   # Assuming that centers are stored in a table named "centers",
   # and the names of the columns are "col1", "col2", "col3", and "col4".
   loadedCenters <- db2matrix ("centers", list(col1", "col2",
   "col3", "col4"), conf="RDev")

## End(Not run)
```

---

file2dgraph *A simple graph loader from a set of files*

Description

file2dgraph function is an example for loading an adjacency matrix from an edge-list which is split in a set of files. It is assumed that edge-list is properly split among nfiles. Split files should correspond to column-wise partitioned adjacency matrix; i.e., each file will be used to load one partition of such a matrix. The files should be available from the same path from all the nodes; i.e., they should be located either on NFS or on the same path in different nodes of the cluster. Moreover, the input argument should be set properly according to the graph stored in the files.

Usage

```
file2dgraph(pathPrefix, nVertices, nfiles)
```

Arguments

pathPrefix	the path and prefix to the files. It should be the same on all nodes of the cluster.
nVertices	the total number of vertices in the graph
nfiles	the number of split files which contain the graph

Value

X	the sparse darray of adjacency matrix

Author(s)

Arash Fard <afard@vertica.com>

Examples

```
## Not run:
   require(HPdgraph)
   distributedR_start()

   # Assuming that the graph has 14 vertices and its edge-list is split
   # column-wise among 7 files.It is also assumed that they are all
   # located in "~/temp/test/" path which is available from all workers,
```

```
    # and their names (before identifying number) starts with "small".
    dgraphF <- file2dgraph("~/temp/test/small", nVertices=14, nfiles=7)

## End(Not run)
```

Index