Package 'birankr'

February 6, 2020

Title Ranking Nodes in Bipartite and Weighted Networks

Version 1.0.1

Description Highly efficient functions for estimating various rank (centrality) measures of nodes in bipartite graphs (two-mode networks). Includes methods for estimating HITS, Co-HITS, BGRM, and BiRank with implementation primarily inspired by He et al. (2016) <doi:10.1109/TKDE.2016.2611584>. Also provides easy-to-use tools for efficiently estimating PageRank in one-mode graphs, incorporating or removing edge-weights during rank estimation, projecting two-mode graphs to one-mode, and for converting edgelists and matrices to sparseMatrix format. Best of all, the package's rank estimators can work directly with common formats of network data including edgelists (class data.frame, data.table, or tbl_df) and adjacency matrices (class matrix or dgCMatrix).

Depends R (>= 3.4.0), Matrix, data.table

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

NeedsCompilation no

Author Brian Aronson [aut, cre],
Kai-Cheng Yang [aut]

Maintainer Brian Aronson <bdaronson@gmail.com>

R topics documented:

`bipartite_rank` *Bipartite Ranks*

Description

Estimate bipartite ranks (centrality scores) of nodes from an edge list or adjacency matrix. Functions as a wrapper for estimating rank based on a number of normalizers (algorithms) including HITS, CoHITS, BGRM, and BiRank. Returns a vector of ranks or (optionally) a list containing a vector for each mode. If the provided data is an edge list, this function returns ranks ordered by the unique values in the supplied edge list.

Usage

```
bipartite_rank(
  data,
  sender_name = NULL,
  receiver_name = NULL,
  weight_name = NULL,
  rm_weights = FALSE,
  duplicates = c("add", "remove"),
  normalizer = c("HITS", "CoHITS", "BGRM", "BiRank"),
  return_mode = c("rows", "columns", "both"),
  return_data_frame = TRUE,
  alpha = 0.85,
  beta = 0.85,
  max_iter = 200,
  tol = 1e-04,
  verbose = FALSE
)
```

Arguments

`data`	Data to use for estimating rank. Must contain bipartite graph data, either formatted as an edge list (class data.frame, data.table, or tibble (tbl_df)) or as an adjacency matrix (class matrix or dgCMatrix).
`sender_name`	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to first column of edge list.
`receiver_name`	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to the second column of edge list.
`weight_name`	Name of edge weights. Parameter ignored if data is an adjacency matrix. Defaults to edge weights = 1.
`rm_weights`	Removes edge weights from graph object before estimating rank. Parameter ignored if data is an edge list. Defaults to FALSE.
`duplicates`	How to treat duplicate edges if any in data. Parameter ignored if data is an adjacency matrix. If option "add" is selected, duplicated edges and corresponding edge weights are collapsed via addition. Otherwise, duplicated edges are removed and only the first instance of a duplicated edge is used. Defaults to "add".
`normalizer`	Normalizer (algorithm) used for estimating node ranks (centrality scores). Options include HITS, CoHITS, BGRM, and BiRank. Defaults to HITS.

return_mode	Mode for which to return ranks. Defaults to "rows" (the first column of an edge list).
return_data_frame	Return results as a data frame with node names in the first column and ranks in the second column. If set to FALSE, the function just returns a named vector of ranks. Defaults to TRUE.
alpha	Dampening factor for first mode of data. Defaults to 0.85.
beta	Dampening factor for second mode of data. Defaults to 0.85.
max_iter	Maximum number of iterations to run before model fails to converge. Defaults to 200.
tol	Maximum tolerance of model convergence. Defaults to 1.0e-4.
verbose	Show the progress of this function. Defaults to FALSE.

Details

For information about the different normalizers available in this function, see the descriptions for the HITS, CoHITS, BGRM, and BiRank functions. However, below outlines the key differences between the normalizers, with K_d and K_p representing diagonal matrices with generalized degrees (sum of the edge weights) on the diagonal (e.g. $(K_d)_{ii} = \sum_j w_{ij}$ and $(K_p)_{jj} = \sum_i w_{ij}$).

Transition matrix	S_p	S_d
HITS	W^T	W
Co-HITS	$W^T K_d^{-1}$	$W K_p^{-1}$
BGRM	$K_p^{-1} W^T K_d^{-1}$	$K_d^{-1} W K_p^{-1}$
BiRank	$K_p^{-1/2} W^T K_d^{-1/2}$	$K_d^{-1/2} W K_p^{-1/2}$

Value

A dataframe containing each node name and node rank. If return_data_frame changed to FALSE or input data is classed as an adjacency matrix, returns a vector of node ranks. Does not return node ranks for isolates.

Examples

```
#create edge list between patients and providers
    df <- data.table(
      patient_id = sample(x = 1:10000, size = 10000, replace = TRUE),
      provider_id = sample(x = 1:5000, size = 10000, replace = TRUE)
    )

#estimate CoHITS ranks
    CoHITS <- bipartite_rank(data = df, normalizer = "CoHITS")
```

---

br_bgrm *BGRM Ranks*

Description

Estimate BGRM ranks of nodes from an edge list or adjacency matrix. Returns a vector of ranks or (optionally) a list containing a vector for each mode. If the provided data is an edge list, this function returns ranks ordered by the unique values in the selected mode.

Usage

```
br_bgrm(
  data,
  sender_name = NULL,
  receiver_name = NULL,
  weight_name = NULL,
  rm_weights = FALSE,
  duplicates = c("add", "remove"),
  return_mode = c("rows", "columns", "both"),
  return_data_frame = TRUE,
  alpha = 0.85,
  beta = 0.85,
  max_iter = 200,
  tol = 1e-04,
  verbose = FALSE
)
```

Arguments

data	Data to use for estimating BGRM. Must contain bipartite graph data, either formatted as an edge list (class data.frame, data.table, or tibble (tbl_df)) or as an adjacency matrix (class matrix or dgCMatrix).
sender_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to first column of edge list.
receiver_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to the second column of edge list.
weight_name	Name of edge weights. Parameter ignored if data is an adjacency matrix. Defaults to edge weights = 1.
rm_weights	Removes edge weights from graph object before estimating BGRM. Parameter ignored if data is an edge list. Defaults to FALSE.
duplicates	How to treat duplicate edges if any in data. Parameter ignored if data is an adjacency matrix. If option "add" is selected, duplicated edges and corresponding edge weights are collapsed via addition. Otherwise, duplicated edges are removed and only the first instance of a duplicated edge is used. Defaults to "add".
return_mode	Mode for which to return BGRM ranks. Defaults to "rows" (the first column of an edge list).
return_data_frame	Return results as a data frame with node names in the first column and ranks in the second column. If set to FALSE, the function just returns a named vector of ranks. Defaults to TRUE.
alpha	Dampening factor for first mode of data. Defaults to 0.85.
beta	Dampening factor for second mode of data. Defaults to 0.85.

max_iter	Maximum number of iterations to run before model fails to converge. Defaults to 200.
tol	Maximum tolerance of model convergence. Defaults to 1.0e-4.
verbose	Show the progress of this function. Defaults to FALSE.

Details

Created by Rui et. al (2007) doi: 10.1145/1291233.1291378, BGRM (Bipartite Graph Reinforcement Model) was developed explicitly for use in bipartite graphs. Like every bipartite ranking algorithm in this package, BGRM simultaneously estimates ranks across each mode of the input data. BGRM primarily differs from CoHITS and HITS by symmetrically normalizing the transition matrix, both by the out-degree of the source node and the indegree of the target node.

Value

A dataframe containing each node name and node rank. If return_data_frame changed to FALSE or input data is classed as an adjacency matrix, returns a vector of node ranks. Does not return node ranks for isolates.

References

Xiaoguang Rui, Mingjing Li, Zhiwei Li, Wei-Ying Ma, and Nenghai Yu. "Bipartite graph reinforcement model for web image annotation". In *Proceedings of the 15th ACM International Conference on Multimedia*, MM '07, pages 585-594, New York, NY, USA, 2007. ACM.

Examples

```
#create edge list between patients and providers
    df <- data.table(
      patient_id = sample(x = 1:10000, size = 10000, replace = TRUE),
      provider_id = sample(x = 1:5000, size = 10000, replace = TRUE)
    )

#estimate BGRM ranks
    BGRM <- br_bgrm(data = df)
```

---

br_birank *BiRanks*

Description

Estimate BiRanks of nodes from an edge list or adjacency matrix. Returns a vector of ranks or (optionally) a list containing a vector for each mode. If the provided data is an edge list, this function returns ranks ordered by the unique values in the selected mode.

Usage

```
br_birank(
  data,
  sender_name = NULL,
  receiver_name = NULL,
  weight_name = NULL,
```

```
  rm_weights = FALSE,
  duplicates = c("add", "remove"),
  return_mode = c("rows", "columns", "both"),
  return_data_frame = TRUE,
  alpha = 0.85,
  beta = 0.85,
  max_iter = 200,
  tol = 1e-04,
  verbose = FALSE
)
```

Arguments

data	Data to use for estimating BiRank. Must contain bipartite graph data, either formatted as an edge list (class data.frame, data.table, or tibble (tbl_df)) or as an adjacency matrix (class matrix or dgCMatrix).
sender_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to first column of edge list.
receiver_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to the second column of edge list.
weight_name	Name of edge weights. Parameter ignored if data is an adjacency matrix. Defaults to edge weights = 1.
rm_weights	Removes edge weights from graph object before estimating BiRank. Parameter ignored if data is an edge list. Defaults to FALSE.
duplicates	How to treat duplicate edges if any in data. Parameter ignored if data is an adjacency matrix. If option "add" is selected, duplicated edges and corresponding edge weights are collapsed via addition. Otherwise, duplicated edges are removed and only the first instance of a duplicated edge is used. Defaults to "add".
return_mode	Mode for which to return BiRank ranks. Defaults to "rows" (the first column of an edge list).
return_data_frame	Return results as a data frame with node names in first column and ranks in the second column. If set to FALSE, the function just returns a named vector of ranks. Defaults to TRUE.
alpha	Dampening factor for first mode of data. Defaults to 0.85.
beta	Dampening factor for second mode of data. Defaults to 0.85.
max_iter	Maximum number of iterations to run before model fails to converge. Defaults to 200.
tol	Maximum tolerance of model convergence. Defaults to 1.0e-4.
verbose	Show the progress of this function. Defaults to FALSE.

Details

Created by He et al. (2017) doi: 10.1109/TKDE.2016.2611584, BiRank is a highly generalizable algorithm that was developed explicitly for use in bipartite graphs. In fact, He et al.'s implementation of BiRank forms the basis of this package's implementation of all other bipartite ranking algorithms. Like every other bipartite ranking algorithm, BiRank simultaneously estimates ranks across each mode of the input data. BiRank's implementation is also highly similar to BGRM in

that it symmetrically normalizes the transition matrix. BiRank differs from BGRM only in that it normalizes the transition matrix by the square-root outdegree of the source node and the square-root indegree of the target node.

Value

A dataframe containing each node name and node rank. If return_data_frame changed to FALSE or input data is classed as an adjacency matrix, returns a vector of node ranks. Does not return node ranks for isolates.

References

Xiangnan He, Ming Gao, Min-Yen Kan, and Dingxian Wang. "Birank: Towards ranking on bipartite graphs". *IEEE Transactions on Knowledge and Data Engineering*, 29(1):57-71, 2016

Examples

```
#create edge list between patients and providers
    df <- data.table(
      patient_id = sample(x = 1:10000, size = 10000, replace = TRUE),
      provider_id = sample(x = 1:5000, size = 10000, replace = TRUE)
    )

#estimate BiRank ranks
    BiRank <- br_birank(data = df)
```

---

br_cohits *CoHITS Ranks*

Description

Estimate CoHITS ranks of nodes from an edge list or adjacency matrix. Returns a vector of ranks or (optionally) a list containing a vector for each mode. If the provided data is an edge list, this function returns ranks ordered by the unique values in the selected mode.

Usage

```
br_cohits(
  data,
  sender_name = NULL,
  receiver_name = NULL,
  weight_name = NULL,
  rm_weights = FALSE,
  duplicates = c("add", "remove"),
  return_mode = c("rows", "columns", "both"),
  return_data_frame = TRUE,
  alpha = 0.85,
  beta = 0.85,
  max_iter = 200,
  tol = 1e-04,
  verbose = FALSE
)
```

Arguments

data	Data to use for estimating CoHITS. Must contain bipartite graph data, either formatted as an edge list (class data.frame, data.table, or tibble (tbl_df)) or as an adjacency matrix (class matrix or dgCMatrix).
sender_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to first column of edge list.
receiver_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to the second column of edge list.
weight_name	Name of edge weights. Parameter ignored if data is an adjacency matrix. Defaults to edge weights = 1.
rm_weights	Removes edge weights from graph object before estimating CoHITS. Parameter ignored if data is an edge list. Defaults to FALSE.
duplicates	How to treat duplicate edges if any in data. Parameter ignored if data is an adjacency matrix. If option "add" is selected, duplicated edges and corresponding edge weights are collapsed via addition. Otherwise, duplicated edges are removed and only the first instance of a duplicated edge is used. Defaults to "add".
return_mode	Mode for which to return CoHITS ranks. Defaults to "rows" (the first column of an edge list).
return_data_frame	Return results as a data frame with node names in the first column and ranks in the second column. If set to FALSE, the function just returns a named vector of ranks. Defaults to TRUE.
alpha	Dampening factor for first mode of data. Defaults to 0.85.
beta	Dampening factor for second mode of data. Defaults to 0.85.
max_iter	Maximum number of iterations to run before model fails to converge. Defaults to 200.
tol	Maximum tolerance of model convergence. Defaults to 1.0e-4.
verbose	Show the progress of this function. Defaults to FALSE.

Details

Created by Deng, Lyo, and Kind (2009) doi: 10.1145/1557019.1557051, CoHITS was developed explicitly for use in bipartite graphs as a way to better-incorporate content information (the "Co" in CoHITS) in HITS ranks. Like HITS, CoHITS is based on a markov process for simultaneously estimating ranks across each mode of the input data. CoHITS primarily differs from HITS in that it normalizes the transition matrix by the out-degree of the source nodes, leading to an interpretation more similar to that of a random walk.

Value

A dataframe containing each node name and node rank. If return_data_frame changed to FALSE or input data is classed as an adjacency matrix, returns a vector of node ranks. Does not return node ranks for isolates.

References

Hongbo Deng, Michael R. Lyu, and Irwin King. "A generalized co-hits algorithm and its application to bipartite graphs". In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 239-248, New York, NY, USA, 2009. ACM.

Examples

```
#create edge list between patients and providers
    df <- data.table(
      patient_id = sample(x = 1:10000, size = 10000, replace = TRUE),
      provider_id = sample(x = 1:5000, size = 10000, replace = TRUE)
    )

#estimate CoHITS ranks
    CoHITS <- br_cohits(data = df)
```

---

br_hits *HITS Ranks*

Description

Estimate HITS ranks of nodes from an edge list or adjacency matrix. Returns a vector of ranks or (optionally) a list containing a vector for each mode. If the provided data is an edge list, this function returns ranks ordered by the unique values in the selected mode.

Usage

```
br_hits(
  data,
  sender_name = NULL,
  receiver_name = NULL,
  weight_name = NULL,
  rm_weights = FALSE,
  duplicates = c("add", "remove"),
  return_mode = c("rows", "columns", "both"),
  return_data_frame = TRUE,
  alpha = 0.85,
  beta = 0.85,
  max_iter = 200,
  tol = 1e-04,
  verbose = FALSE
)
```

Arguments

data	Data to use for estimating HITS. Must contain bipartite graph data, either formatted as an edge list (class data.frame, data.table, or tibble (tbl_df)) or as an adjacency matrix (class matrix or dgCMatrix).
sender_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to first column of edge list.
receiver_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to the second column of edge list.
weight_name	Name of edge weights. Parameter ignored if data is an adjacency matrix. Defaults to edge weights = 1.
rm_weights	Removes edge weights from graph object before estimating HITS. Parameter ignored if data is an edge list. Defaults to FALSE.

duplicates	How to treat duplicate edges if any in data. Parameter ignored if data is an adjacency matrix. If option "add" is selected, duplicated edges and corresponding edge weights are collapsed via addition. Otherwise, duplicated edges are removed and only the first instance of a duplicated edge is used. Defaults to "add".
return_mode	Mode for which to return HITS ranks. Defaults to "rows" (the first column of an edge list).
return_data_frame	Return results as a data frame with node names in the first column and ranks in the second column. If set to FALSE, the function just returns a named vector of ranks. Defaults to TRUE.
alpha	Dampening factor for first mode of data. Defaults to 0.85.
beta	Dampening factor for second mode of data. Defaults to 0.85.
max_iter	Maximum number of iterations to run before model fails to converge. Defaults to 200.
tol	Maximum tolerance of model convergence. Defaults to 1.0e-4.
verbose	Show the progress of this function. Defaults to FALSE.

Details

Although originally designed for estimating ranks in unipartite graphs, HITS (Hyperlink-Induced Topic Search) is also one of the earliest bipartite ranking algorithms. Created by Jon Kleinberg (2009) doi: 10.1145/324133.324140 as an alternative to PageRank, HITS takes better account of the topology of bipartite networks by iteratively ranking nodes according to their role as an "Authority" and as a "Hub". Nodes with authority have high indegree from high ranking hubs; high ranking hubs have high outdegree to nodes with high authority. This function provides a slightly expanded version of HITS that only interfaces with bipartite networks and that allows for weighted edges. In general, HITS ranks tend to be more sensitive to user query than PageRanks, but HITS is substantially less efficient in ranking large graphs. HITS is likely less preferable than the other bipartite ranking algorithms in most applications. There are a number of contexts where HITS performs poorly, such as in graphs with extreme outliers.

Value

A dataframe containing each node name and node rank. If return_data_frame changed to FALSE or input data is classed as an adjacency matrix, returns a vector of node ranks. Does not return node ranks for isolates.

References

Jon M. Kleinberg. "Authoritative sources in a hyperlinked environment". *J. ACM*, 46(5):604-632, September 1999.

Examples

```
#create edge list between patients and providers
    df <- data.table(
      patient_id = sample(x = 1:10000, size = 10000, replace = TRUE),
      provider_id = sample(x = 1:5000, size = 10000, replace = TRUE)
    )

#estimate HITS ranks
    HITS <- br_hits(data = df)
```

pagerank *Estimate PageRank*

Description

Estimate PageRank (centrality scores) of nodes from an edge list or adjacency matrix. If data is a bipartite graph, estimates PageRank based on a one-mode projection of the input. If the data is an edge list, returns ranks ordered by the unique values in the supplied edge list (first by unique senders, then by unique receivers). Does not support edge-weights; all edges are treated as equal to one.

Usage

```
pagerank(
  data,
  is_bipartite = TRUE,
  project_mode = c("rows", "columns"),
  sender_name = NULL,
  receiver_name = NULL,
  return_data_frame = TRUE,
  alpha = 0.85,
  max_iter = 200,
  tol = 1e-04,
  verbose = FALSE
)
```

Arguments

data	Data to use for estimating PageRank. Can contain unipartite or bipartite graph data, either formatted as an edge list (class data.frame, data.table, or tibble (tbl_df)) or as an adjacency matrix (class matrix or dgCMatrix).
is_bipartite	Indicate whether input data is bipartite (rather than unipartite/one-mode). Defaults to TRUE.
project_mode	Mode for which to return PageRank estimates. Parameter ignored if is_bipartite = FALSE. Defaults to "rows" (the first column of an edge list).
sender_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to first column of edge list.
receiver_name	Name of sender column. Parameter ignored if data is an adjacency matrix. Defaults to the second column of edge list.
return_data_frame	Return results as a data frame with node names in the first column and ranks in the second column. If set to FALSE, the function just returns a named vector of ranks. Defaults to TRUE.
alpha	Dampening factor. Defaults to 0.85.
max_iter	Maximum number of iterations to run before model fails to converge. Defaults to 200.
tol	Maximum tolerance of model convergence. Defaults to 1.0e-4.
verbose	Show the progress of this function. Defaults to FALSE.

Details

The default optional arguments are likely well-suited for most users. However, it is critical to change the is.bipartite function to FALSE when working with one mode data. In addition, when estimating PageRank in unipartite edge lists that contain nodes with outdegrees or indegrees equal to 0, it is recommended that users append self-ties to the edge list to ensure that the returned PageRank estimates are ordered intuitively.

Value

A dataframe containing each node name and node rank. If return_data_frame changed to FALSE or input data is classed as an adjacency matrix, returns a vector of node ranks. Does not return node ranks for isolates.

References

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. "The pagerank citation ranking: Bringing order to the web". Technical report, Stanford InfoLab, 1999

Examples

```
#Prepare one-mode data
    df_one_mode <- data.frame(
      sender = sample(x = 1:10000, size = 10000, replace = TRUE),
      receiver = sample(x = 1:10000, size = 10000, replace = TRUE)
    )

#Add self-loops for all nodes
    unique_ids <- unique(c(df_one_mode$sender, df_one_mode$receiver))
    df_one_mode <- rbind(df_one_mode, data.frame(sender = unique_ids,
    receiver = unique_ids))

#Estimate PageRank in one-mode data
    PageRank <- pagerank(data = df_one_mode, is_bipartite = FALSE)

#Estimate PageRank in two-mode data
    df_two_mode <- data.frame(
      patient_id = sample(x = 1:10000, size = 10000, replace = TRUE),
      provider_id = sample(x = 1:5000, size = 10000, replace = TRUE)
    )
    PageRank <- pagerank(data = df_two_mode)
```

---

project_to_one_mode *Create a one-mode projection of a two mode graph*

Description

Create a one-mode projection of a two mode graph. Converts a rectangular matrix to a square one by taking the cross product of the input matrix. The edge weights in the resulting matrix are equal to the number of transitive ties of each node in the input matrix.

Usage

```
project_to_one_mode(adj_mat, mode = c("rows", "columns"))
```

Arguments

adj_mat	Sparse matrix of class dgCMatrix
mode	Mode to return. Defaults to projecting by rows.

Value

A double or complex matrix, with appropriate dimnames taken from x and y.

Examples

```
#make matrix
   my_matrix <- sparseMatrix(i = c(1, 1, 2, 3, 4, 4, 5, 6, 7, 7),
       j = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10), x = 1
   )
#project to one mode
   project_to_one_mode(adj_mat = my_matrix, mode = "rows")
```

---

sparsematrix_from_edgelist

Convert edge list to sparse matrix

Description

Converts edge lists (class data.frame) to sparse matrices (class "dgCMatrix"). For unipartite edge lists that contain any nodes with outdegrees or indegrees equal to 0, it is recommended that users append self-ties to the edge list to ensure that the IDs of the rows and columns are ordered intuitively to the user.

Usage

```
sparsematrix_from_edgelist(
  data,
  sender_name = NULL,
  receiver_name = NULL,
  weight_name = NULL,
  duplicates = c("add", "remove"),
  is_bipartite = T
)
```

Arguments

data	Edge list to convert to sparse matrix. Must be in edge list format and of class data.frame, data.table, or tbl_df.
sender_name	Name of sender column. Defaults to the first column of an edge list.
receiver_name	Name of sender column. Defaults to the second column of an edge list.
weight_name	Name of edge weights. Defaults to edge weight = 1.
duplicates	How to treat duplicate edges from edge list. If option "add" is selected, duplicated edges and corresponding edge weights are collapsed via addition. Otherwise, duplicated edges or removed and only the first instance of a duplicated edge is used. Defaults to "add".
is_bipartite	Indicate whether input data is bipartite (rather than unipartite/one-mode). Defaults to TRUE.

Value

A sparse matrix of class dgCMatrix.

Examples

```
#make edge.list
   df <- data.frame(
     id1 = sample(x = 1:20, size = 100, replace = TRUE),
     id2 = sample(x = 1:10, size = 100, replace = TRUE),
     weight = sample(x = 1:10, size = 100, replace = TRUE)
   )
#convert to sparsematrix
   sparsematrix_from_edgelist(data = df)
```

---

sparsematrix_from_matrix

Convert matrix to sparse matrix

Description

Converts adjacency matrices (class "matrix") to a sparse matrices (class "dgCMatrix").

Usage

```
sparsematrix_from_matrix(adj_mat)
```

Arguments

adj_mat	Adjacency matrix.

Value

A sparse matrix of class dgCMatrix.

Examples

```
#make matrix
   my_matrix <- rep(0, 100)
   my_matrix[c(1, 11, 22, 33, 44, 54, 65, 76, 87, 97)] <- 1
   my_matrix <- matrix(data = my_matrix, nrow = 10, ncol = 10)
#convert to sparsematrix
   sparsematrix_from_matrix(adj_mat = my_matrix)
```

sparsematrix_rm_weights *Remove sparse matrix edge weights*

Description

Removes edge weights from sparse matrices.

Usage

```
sparsematrix_rm_weights(adj_mat)
```

Arguments

adj_mat	Sparse matrix of class dgCMatrix

Value

A sparse matrix of class dgCMatrix.

Examples

```
#make matrix
   my_matrix <- sparseMatrix(
       i = c(1, 1, 2, 3, 4, 4, 5, 6, 7, 7),
       j = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10),
       x = c(1, 1, 3, 1, 2, 1, 1, 1, 2, 1)
   )
#remove weights
   sparsematrix_rm_weights(my_matrix)
```

Index