

# Package ‘DCEM’

January 20, 2025

**Type** Package

**Title** Clustering Big Data using Expectation Maximization Star (EM\*)  
Algorithm

**Version** 2.0.5

**Maintainer** Sharma Parichit <parishar@iu.edu>

**Description** Implements the Improved Expectation Maximisation EM\* and the traditional EM algorithm for clustering big data (gaussian mixture models for both multivariate and univariate datasets). This version implements the faster alternative-EM\* that expedites convergence via structure based data segregation. The implementation supports both random and K-means++ based initialization. Reference: Parichit Sharma, Hasan Kurban, Mehmet Dalkilic (2022) <[doi:10.1016/j.softx.2021.100944](https://doi.org/10.1016/j.softx.2021.100944)>. Hasan Kurban, Mark Jenne, Mehmet Dalkilic (2016) <[doi:10.1007/s41060-017-0062-1](https://doi.org/10.1007/s41060-017-0062-1)>.

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**Imports** mvtnorm (>= 1.0.7), matrixcalc (>= 1.0.3), MASS (>= 7.3.49),  
Rcpp (>= 1.0.2)

**LinkingTo** Rcpp

**RoxygenNote** 7.1.2

**Depends** R(>= 3.2.0)

**URL** <https://github.com/parichit/DCEM>

**BugReports** <https://github.com/parichit/DCEM/issues>

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** Sharma Parichit [aut, cre, ctb],  
Kurban Hasan [aut, ctb],  
Dalkilic Mehmet [aut]

**Repository** CRAN

**Date/Publication** 2022-01-16 00:02:52 UTC

## Contents

build_heap . . . . .	2
DCEM . . . . .	3
dcem_cluster_mv . . . . .	4
dcem_cluster_uv . . . . .	6
dcem_predict . . . . .	7
dcem_star_cluster_mv . . . . .	8
dcem_star_cluster_uv . . . . .	9
dcem_star_train . . . . .	10
dcem_test . . . . .	11
dcem_train . . . . .	12
expectation_mv . . . . .	14
expectation_uv . . . . .	15
get_priors . . . . .	15
insert_nodes . . . . .	16
ionosphere_data . . . . .	16
maximisation_mv . . . . .	17
maximisation_uv . . . . .	18
max_heapify . . . . .	18
meu_mv . . . . .	19
meu_mv_impr . . . . .	19
meu_uv . . . . .	20
meu_uv_impr . . . . .	20
separate_data . . . . .	21
sigma_mv . . . . .	22
sigma_uv . . . . .	22
trim_data . . . . .	23
update_weights . . . . .	23
validate_data . . . . .	24
<b>Index</b>	<b>25</b>

---

build_heap	<i>build_heap: Part of DCEM package.</i>
------------	--

---

## Description

Implements the creation of heap. Internally called by the `dcem_star_train`.

## Usage

```
build_heap(data)
```

**Arguments**

data (NumericMatrix): The dataset provided by the user.

**Value**

A NumericMatrix with the max heap property.

**Author(s)**

Parichit Sharma <parishar@iu.edu>, Hasan Kurban, Mehmet Dalkilic

---

DCEM

*DCEM: Clustering Big Data using Expectation Maximization Star (EM\*) Algorithm.*

---

**Description**

Implements the EM\* and EM algorithm for clustering the (univariate and multivariate) Gaussian mixture data.

**Demonstration and Testing**

**Cleaning the data:** The data should be cleaned (redundant columns should be removed). For example columns containing the labels or redundant entries (such as a column of all 0's or 1's). See [trim\\_data](#) for details on cleaning the data. Refer: [dcem\\_test](#) for more details.

**Understanding the output of [dcem\\_test](#)**

The function `dcem_test()` returns a list of objects. This list contains the parameters associated with the Gaussian(s), posterior probabilities (`prob`), mean (`meu`), co-variance/standard-deviation(`sigma`), priors (`prior`) and cluster membership for data (`membership`).

**Note:** The routine `dcem_test()` is only for demonstration purpose. The function [dcem\\_test](#) calls the main routine [dcem\\_train](#). See [dcem\\_train](#) for further details.

**How to run on your dataset**

See [dcem\\_train](#) and [dcem\\_star\\_train](#) for examples.

**Package organization**

The package is organized as a set of preprocessing functions and the core clustering modules. These functions are briefly described below.

1. [trim\\_data](#): This is used to remove the columns from the dataset. The user should clean the dataset before calling the `dcem_train` routine. **User can also clean the dataset themselves (without using `trim_data`) and then pass it to the `dcem_train` function**

2. `dcm_star_train` and `dcm_train`: These are the primary interface to the EM\* and EM algorithms respectively. These functions accept the cleaned dataset and other parameters (number of iterations, convergence threshold etc.) and run the algorithm until:
  - (a) The number of iterations is reached.
  - (b) The convergence is achieved.

### DCEM supports following initialization schemes

1. **Random Initialization:** Initializes the mean randomly. Refer `meu_uv` and `meu_mv` for initialization on univariate and multivariate data respectively.
2. **Improved Initialization:** Based on the Kmeans++ idea published in, K-means++: The Advantages of Careful Seeding, David Arthur and Sergei Vassilvitskii. URL <http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf>. See `meu_uv_impr` and `meu_mv_impr` for details.
3. Choice of initialization scheme can be specified as the **seeding** parameter during the training. See `dcm_train` for further details.

### References

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

**External Packages:** DCEM requires R packages 'mvtnorm'[1], 'matrixcalc'[2] 'Rcpp'[3] and 'MASS'[4] for multivariate density calculation, checking matrix singularity, compiling routines written in C and simulating mixture of gaussians, respectively.

[1] Alan Genz, Frank Bretz, Tetsuhisa Miwa, Xuefei Mi, Friedrich Leisch, Fabian Scheipl, Torsten Hothorn (2019). mvtnorm: Multivariate Normal and t Distributions. R package version 1.0-7. URL <http://CRAN.R-project.org/package=mvtnorm>

[2] Frederick Novomestky (2012). matrixcalc: Collection of functions for matrix calculations. R package version 1.0-3. <https://CRAN.R-project.org/package=matrixcalc>

[3] Dirk Eddelbuettel and Romain Francois (2011). Rcpp: Seamless R and C++ Integration. *Journal of Statistical Software*, 40(8), 1-18. URL <http://www.jstatsoft.org/v40/i08/>.

[4] Venables, W. N. & Ripley, B. D. (2002) *Modern Applied Statistics with S*. Fourth Edition. Springer, New York. ISBN 0-387-95457-0

[5] K-Means++: The Advantages of Careful Seeding, David Arthur and Sergei Vassilvitskii. URL <http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf>

---

dcm\_cluster\_mv

*dcm\_cluster (multivariate data): Part of DCEM package.*

---

### Description

Implements the Expectation Maximization algorithm for multivariate data. This function is called by the `dcm_train` routine.

**Usage**

```
dcm_cluster_mv(data, meu, sigma, prior, num_clusters, iteration_count,
               threshold, num_data)
```

**Arguments**

data	A matrix: The dataset provided by the user.
meu	(matrix): The matrix containing the initial meu(s).
sigma	(list): A list containing the initial covariance matrices.
prior	(vector): A vector containing the initial prior.
num_clusters	(numeric): The number of clusters specified by the user. Default value is 2.
iteration_count	(numeric): The number of iterations for which the algorithm should run, if the convergence is not achieved then the algorithm stops. Default: 200.
threshold	(numeric): A small value to check for convergence (if the estimated meu are within this specified threshold then the algorithm stops and exit). <b>Note: Choosing a very small value (0.0000001) for threshold can increase the runtime substantially and the algorithm may not converge. On the other hand, choosing a larger value (0.1) can lead to sub-optimal clustering. Default: 0.00001.</b>
num_data	(numeric): The total number of observations in the data.

**Value**

A list of objects. This list contains parameters associated with the Gaussian(s) (posterior probabilities, meu, co-variance and prior)

1. (1) Posterior Probabilities: **prob** :A matrix of posterior-probabilities.
2. (2) Meu: **meu**: It is a matrix of meu(s). Each row in the matrix corresponds to one meu.
3. (3) Sigma: Co-variance matrices: **sigma**
4. (4) prior: **prior**: A vector of prior.
5. (5) Membership: **membership**: A vector of cluster membership for data.

**References**

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

---

dcem\_cluster\_uv      *dcem\_cluster\_uv (univariate data): Part of DCEM package.*

---

### Description

Implements the Expectation Maximization algorithm for the univariate data. This function is internally called by the `dcem_train` routine.

### Usage

```
dcem_cluster_uv(data, meu, sigma, prior, num_clusters, iteration_count,
                threshold, num_data, numcols)
```

### Arguments

<code>data</code>	(matrix): The dataset provided by the user (converted to matrix format).
<code>meu</code>	(vector): The vector containing the initial meu.
<code>sigma</code>	(vector): The vector containing the initial standard deviation.
<code>prior</code>	(vector): The vector containing the initial prior.
<code>num_clusters</code>	(numeric): The number of clusters specified by the user. Default is 2.
<code>iteration_count</code>	(numeric): The number of iterations for which the algorithm should run. If the convergence is not achieved then the algorithm stops. Default: 200.
<code>threshold</code>	(numeric): A small value to check for convergence (if the estimated meu(s) are within the threshold then the algorithm stops). <b>Note: Choosing a very small value (0.0000001) for threshold can increase the runtime substantially and the algorithm may not converge. On the other hand, choosing a larger value (0.1) can lead to sub-optimal clustering. Default: 0.00001.</b>
<code>num_data</code>	(numeric): The total number of observations in the data.
<code>numcols</code>	(numeric): Number of columns in the dataset (After processing the missing values).

### Value

A list of objects. This list contains parameters associated with the Gaussian(s) (posterior probabilities, meu, standard-deviation and prior)

- (1) Posterior Probabilities: **prob**: A matrix of posterior-probabilities.
- (2) Meu(s): **meu**: It is a vector of meu. Each element of the vector corresponds to one meu.
- (3) Sigma: Standard-deviation(s): **sigma**: A vector of standard deviation.
- (4) prior: **prior**: A vector of prior.
- (5) Membership: **membership**: A vector of cluster membership for data.

## References

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

---

dcm_predict	<i>dcm_predict: Part of DCEM package.</i>
-------------	---

---

## Description

Predict the cluster membership of test data based on the learned parameters i.e, output from `dcm_train` or `dcm_star_train`.

## Usage

```
dcm_predict(param_list, data)
```

## Arguments

param_list	(list): List of distribution parameters. The list contains the learned parameteres of the distribution.
data	(vector or dataframe): A vector of data for univariate data. A dataframe (rows represent the data and columns represent the features) for multivariate data.

## Value

A list containing the cluster membership for the test data.

## References

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

## Examples

```
# Simulating a mixture of univariate samples from three distributions
# with meu as 20, 70 and 100 and standard deviation as 10, 100 and 40 respectively.
sample_uv_data = as.data.frame(c(rnorm(100, 20, 5), rnorm(70, 70, 1), rnorm(50, 100, 2)))

# Select first few points from each distribution as test data
test_data = as.vector(sample_uv_data[c(1:5, 101:105, 171:175),])

# Remove the test data from the training set
sample_uv_data = as.data.frame(sample_uv_data[-c(1:5, 101:105, 171:175), ])

# Randomly shuffle the samples.
sample_uv_data = as.data.frame(sample_uv_data[sample(nrow(sample_uv_data)),])

# Calling the dcm_train() function on the simulated data with threshold of
```

```
# 0.000001, iteration count of 1000 and random seeding respectively.
sample_uv_out = dcm_train(sample_uv_data, num_clusters = 3, iteration_count = 100,
threshold = 0.001)

# Predict the membership for test data
test_data_membership <- dcm_predict(sample_uv_out, test_data)

# Access the output
print(test_data_membership)
```

---

dcm\_star\_cluster\_mv *dcm\_star\_cluster\_mv (multivariate data): Part of DCEM package.*

---

## Description

Implements the EM\* algorithm for multivariate data. This function is called by the `dcm_star_train` routine.

## Usage

```
dcm_star_cluster_mv(data, meu, sigma, prior, num_clusters, iteration_count, num_data)
```

## Arguments

<code>data</code>	(matrix): The dataset provided by the user.
<code>meu</code>	(matrix): The matrix containing the initial <code>meu(s)</code> .
<code>sigma</code>	(list): A list containing the initial covariance matrices.
<code>prior</code>	(vector): A vector containing the initial priors.
<code>num_clusters</code>	(numeric): The number of clusters specified by the user. Default value is 2.
<code>iteration_count</code>	(numeric): The number of iterations for which the algorithm should run, if the convergence is not achieved then the algorithm stops and exits. Default: 200.
<code>num_data</code>	(numeric): Number of rows in the dataset.

## Value

A list of objects. This list contains parameters associated with the Gaussian(s) (posterior probabilities, `meu`, co-variance and priors)

- (1) Posterior Probabilities: **prob** A matrix of posterior-probabilities for the points in the dataset.
- (2) `Meu`: **meu**: A matrix of `meu(s)`. Each row in the matrix corresponds to one `meu`.
- (3) `Sigma`: Co-variance matrices: **sigma**: List of co-variance matrices.
- (4) `Priors`: **prior**: A vector of prior.
- (5) `Membership`: **membership**: A vector of cluster membership for data.



## References

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

---

dcm\_star\_cluster\_uv *dcm\_star\_cluster\_uv (univariate data): Part of DCEM package.*

---

## Description

Implements the EM\* algorithm for the univariate data. This function is called by the `dcm_star_train` routine.

## Usage

```
dcm_star_cluster_uv(data, meu, sigma, prior, num_clusters, num_data,
  iteration_count)
```

## Arguments

<code>data</code>	(matrix): The dataset provided by the user.
<code>meu</code>	(vector): The vector containing the initial meu.
<code>sigma</code>	(vector): The vector containing the initial standard deviation.
<code>prior</code>	(vector): The vector containing the initial priors.
<code>num_clusters</code>	(numeric): The number of clusters specified by the user. Default is 2.
<code>num_data</code>	(numeric): number of rows in the dataset (After processing the missing values).
<code>iteration_count</code>	(numeric): The number of iterations for which the algorithm should run. If the convergence is not achieved then the algorithm stops. Default is 100.

## Value

A list of objects. This list contains parameters associated with the Gaussian(s) (posterior probabilities, meu, standard-deviation and priors)

- (1) Posterior Probabilities: **prob** A matrix of posterior-probabilities
- (2) Meu: **meu**: It is a vector of meu. Each element of the vector corresponds to one meu.
- (3) Sigma: Standard-deviation(s): **sigma**  
For univariate data: Vector of standard deviation.
- (4) Priors: **prior**: A vector of priors.
- (5) Membership: **membership**: A vector of cluster membership for data.

## References

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

---

dcem\_star\_train      *dcem\_star\_train: Part of DCEM package.*

---

## Description

Implements the improved EM\* ([1], [2]) algorithm. EM\* avoids revisiting all but high expressive data via structure based data segregation thus resulting in significant speed gain. It calls the [dcm\\_star\\_cluster\\_uv](#) routine internally (univariate data) and [dcm\\_star\\_cluster\\_mv](#) for (multivariate data).

## Usage

```
dcm_star_train(data, iteration_count, num_clusters, seed_meu, seeding)
```

## Arguments

data	(dataframe): The dataframe containing the data. See <a href="#">trim_data</a> for cleaning the data.
iteration_count	(numeric): The number of iterations for which the algorithm should run, if the convergence is not achieved then the algorithm stops and exit. <b>Default: 200.</b>
num_clusters	(numeric): The number of clusters. <b>Default: 2</b>
seed_meu	(matrix): The user specified set of meu to use as initial centroids. <b>Default: None</b>
seeding	(string): The initialization scheme ('rand', 'improved'). <b>Default: rand</b>

## Value

A list of objects. This list contains parameters associated with the Gaussian(s) (posterior probabilities, meu, sigma and priors). The parameters can be accessed as follows where `sample_out` is the list containing the output:

- (1) Posterior Probabilities: **sample\_out\$prob** A matrix of posterior-probabilities.
- (2) Meu(s): **sample\_out\$meu**  
 For multivariate data: It is a matrix of meu(s). Each row in the matrix corresponds to one mean.  
 For univariate data: It is a vector of meu(s). Each element of the vector corresponds to one meu.
- (3) Co-variance matrices: **sample\_out\$sigma**  
 For multivariate data: List of co-variance matrices.  
 Standard-deviation: **sample\_out\$sigma**  
 For univariate data: Vector of standard deviation.
- (4) Priors: **sample\_out\$prior** A vector of priors.
- (5) Membership: **sample\_out\$membership**: A dataframe of cluster membership for data. Columns numbers are data indices and values are the assigned clusters.

## References

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

## Examples

```
# Simulating a mixture of univariate samples from three distributions
# with mean as 20, 70 and 100 and standard deviation as 10, 100 and 40 respectively.
sample_uv_data = as.data.frame(c(rnorm(100, 20, 5), rnorm(70, 70, 1), rnorm(50, 100, 2)))

# Randomly shuffle the samples.
sample_uv_data = as.data.frame(sample_uv_data[sample(nrow(sample_uv_data)),])

# Calling the dcm_star_train() function on the simulated data with iteration count of 1000
# and random seeding respectively.
sample_uv_out = dcm_star_train(sample_uv_data, num_clusters = 3, iteration_count = 100)

# Simulating a mixture of multivariate samples from 2 gaussian distributions.
sample_mv_data = as.data.frame(rbind(MASS::mvrnorm(n=2, rep(2,5), Sigma = diag(5)),
MASS::mvrnorm(n=5, rep(14,5), Sigma = diag(5))))

# Calling the dcm_star_train() function on the simulated data with iteration count of 100 and
# random seeding method respectively.
sample_mv_out = dcm_star_train(sample_mv_data, iteration_count = 100, num_clusters=2)

# Access the output
sample_mv_out$mu
sample_mv_out$sigma
sample_mv_out$prior
sample_mv_out$prob
print(sample_mv_out$membership)
```

---

dcm\_test

*dcm\_test: Part of DCEM package.*

---

## Description

For demonstrating the execution on the bundled dataset.

## Usage

```
dcm_test()
```

## Details

The dcm\_test performs the following steps in order:

1. Read the data from the disk (from the file data/ionosphere\_data.csv). The data folder is under the package installation folder.

2. The dataset details can be see by typing `ionosphere_data` in R-console or at <http://archive.ics.uci.edu/ml/datasets/Ionosphere>.
3. Clean the data (by removing the columns). **The data should be cleaned before use.** Refer `trim_data` to see what columns should be removed and how. The package provides the basic interface for removing columns.
4. Call the `dcm_star_train` on the cleaned data.

### Accessing the output parameters

The function `dcm_test()` calls the `dcm_star_train`. It returns a list of objects as output. This list contains estimated parameters of the Gaussian (posterior probabilities, `meu`, `sigma` and `prior`). The parameters can be accessed as follows where `sample_out` is the list containing the output:

1. (1) Posterior Probabilities: `sample_out$prob` A matrix of posterior-probabilities
2. (2) Meu: `meu`  
For multivariate data: It is a matrix of `meu(s)`. Each row in the matrix corresponds to one `meu`.
3. (3) Co-variance matrices: `sample_out$sigma`  
For multivariate data: List of co-variance matrices for the Gaussian(s).  
Standard-deviation: `sample_out$sigma`  
For univariate data: Vector of standard deviation for the Gaussian(s)
4. (4) Priors: `sample_out$prior` A vector of prior.
5. (5) Membership: `sample_out$membership`: A dataframe of cluster membership for data. Columns numbers are data indices and values are the assigned clusters.

### References

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

---

dcm\_train

*dcm\_train: Part of DCEM package.*

---

### Description

Implements the EM algorithm. It calls the relevant clustering routine internally `dcm_cluster_uv` (univariate data) and `dcm_cluster_mv` (multivariate data).

### Usage

```
dcm_train(data, threshold, iteration_count, num_clusters, seed_meu, seeding)
```

## Arguments

data	(dataframe): The dataframe containing the data. See <a href="#">trim_data</a> for cleaning the data.
threshold	(decimal): A value to check for convergence (if the meu are within this value then the algorithm stops and exit). <b>Default: 0.00001.</b>
iteration_count	(numeric): The number of iterations for which the algorithm should run, if the convergence is not achieved within the specified count then the algorithm stops and exit. <b>Default: 200.</b>
num_clusters	(numeric): The number of clusters. Default: <b>2</b>
seed_meu	(matrix): The user specified set of meu to use as initial centroids. Default: <b>None</b>
seeding	(string): The initialization scheme ('rand', 'improved'). Default: <b>rand</b>

## Value

A list of objects. This list contains parameters associated with the Gaussian(s) (posterior probabilities, meu, sigma and priors). The parameters can be accessed as follows where `sample_out` is the list containing the output:

1. (1) Posterior Probabilities: **sample\_out\$prob**: A matrix of posterior-probabilities
2. (2) Meu: **sample\_out\$meu**  
For multivariate data: It is a matrix of meu(s). Each row in the matrix corresponds to one meu.  
For univariate data: It is a vector of meu(s). Each element of the vector corresponds to one meu.
3. (3) Sigma: **sample\_out\$sigma**  
For multivariate data: List of co-variance matrices for the Gaussian(s).  
For univariate data: Vector of standard deviation for the Gaussian(s).
4. (4) Priors: **sample\_out\$prior**: A vector of priors.
5. (5) Membership: **sample\_out\$membership**: A dataframe of cluster membership for data. Columns numbers are data indices and values are the assigned clusters.

## References

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

## Examples

```
# Simulating a mixture of univariate samples from three distributions
# with meu as 20, 70 and 100 and standard deviation as 10, 100 and 40 respectively.
sample_uv_data = as.data.frame(c(rnorm(100, 20, 5), rnorm(70, 70, 1), rnorm(50, 100, 2)))

# Randomly shuffle the samples.
sample_uv_data = as.data.frame(sample_uv_data[sample(nrow(sample_uv_data)),])

# Calling the dcm_train() function on the simulated data with threshold of
```

```

# 0.000001, iteration count of 1000 and random seeding respectively.
sample_uv_out = dcem_train(sample_uv_data, num_clusters = 3, iteration_count = 100,
threshold = 0.001)

# Simulating a mixture of multivariate samples from 2 gaussian distributions.
sample_mv_data = as.data.frame(rbind(MASS::mvrnorm(n=100, rep(2,5), Sigma = diag(5)),
MASS::mvrnorm(n=50, rep(14,5), Sigma = diag(5))))

# Calling the dcem_train() function on the simulated data with threshold of
# 0.00001, iteration count of 100 and random seeding method respectively.
sample_mv_out = dcem_train(sample_mv_data, threshold = 0.001, iteration_count = 100)

# Access the output
print(sample_mv_out$meu)
print(sample_mv_out$sigma)
print(sample_mv_out$prior)
print(sample_mv_out$prob)
print(sample_mv_out$membership)

```

---

expectation\_mv

*expectation\_mv: Part of DCEM package.*

---

## Description

Calculates the probabilistic weights for the multivariate data.

## Usage

```
expectation_mv(data, weights, meu, sigma, prior, num_clusters, tolerance)
```

## Arguments

data	(matrix): The input data.
weights	(matrix): The probability weight matrix.
meu	(matrix): The matrix of meu.
sigma	(list): The list of sigma (co-variance matrices).
prior	(vector): The vector of priors.
num_clusters	(numeric): The number of clusters.
tolerance	(numeric): The system epsilon value.

## Value

Updated probability weight matrix.

---

expectation_uv	<i>expectation_uv: Part of DCEM package.</i>
----------------	--

---

**Description**

Calculates the probabilistic weights for the univariate data.

**Usage**

```
expectation_uv(data, weights, meu, sigma, prior, num_clusters, tolerance)
```

**Arguments**

data	(matrix): The input data.
weights	(matrix): The probability weight matrix.
meu	(vector): The vector of meu.
sigma	(vector): The vector of sigma (standard-deviations).
prior	(vector): The vector of priors.
num_clusters	(numeric): The number of clusters.
tolerance	(numeric): The system epsilon value.

**Value**

Updated probability weight matrix.

---

get_priors	<i>get_priors: Part of DCEM package.</i>
------------	--

---

**Description**

Initialize the priors.

**Usage**

```
get_priors(num_priors)
```

**Arguments**

num_priors	(numeric): Number of priors one corresponding to each cluster.
------------	--

**Details**

For example, if the user specify 2 priors then the vector will have 2 entries (one for each cluster) where each will be 1/2 or 0.5.

**Value**

A vector of uniformly initialized prior values (numeric).

---

insert_nodes	<i>insert_nodes: Part of DCEM package.</i>
--------------	--

---

**Description**

Implements the node insertion into the heaps.

**Usage**

```
insert_nodes(heap_list, heap_assn, data_probs, leaves_ind, num_clusters)
```

**Arguments**

heap_list	(list): The nested list containing the heaps. Each entry in the list is a list maintained in max-heap structure.
heap_assn	(numeric): The vector representing the heap assignments.
data_probs	(string): A vector containing the probability for data.
leaves_ind	(numeric): A vector containing the indices of leaves in heap.
num_clusters	(numeric): The number of clusters. Default: <b>2</b>

**Value**

A nested list. Each entry in the list is a list maintained in the max-heap structure.

**References**

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

---

ionosphere_data	<i>Ionosphere data: A dataset of 351 radar readings</i>
-----------------	---

---

**Description**

This dataset contains 351 entries (radar readings from a system in goose bay laboratory) and 35 columns. The 35th column is the label column identifying the entry as either good or bad. Additionally, the 2nd column only contains 0's.

**Usage**

```
ionosphere_data
```



**Format**

A file with 351 rows and 35 columns of multivariate data in a csv file. All values are numeric.

**Source**

Space Physics Group Applied Physics Laboratory Johns Hopkins University Johns Hopkins Road Laurel, MD 20723 Web URL: <http://archive.ics.uci.edu/ml/datasets/Ionosphere>

**References:** Sigillito, V. G., Wing, S. P., Hutton, L. V., & Baker, K. B. (1989). Classification of radar returns from the ionosphere using neural networks. Johns Hopkins APL Technical Digest, 10, 262-266.

---

maximisation\_mv

*maximisation\_mv: Part of DCEM package.*

---

**Description**

Calculates meu, sigma and prior based on the updated probability weight matrix.

**Usage**

```
maximisation_mv(data, weights, meu, sigma, prior, num_clusters, num_data)
```

**Arguments**

data	(matrix): The input data.
weights	(matrix): The probability weight matrix.
meu	(matrix): The matrix of meu.
sigma	(list): The list of sigma (co-variance matrices).
prior	(vector): The vector of priors.
num_clusters	(numeric): The number of clusters.
num_data	(numeric): The total number of observations in the data.

**Value**

Updated values for meu, sigma and prior.

---

maximisation_uv	<i>maximisation_uv: Part of DCEM package.</i>
-----------------	---

---

### Description

Calculates meu, sigma and prior based on the updated probability weight matrix.

### Usage

```
maximisation_uv(data, weights, meu, sigma, prior, num_clusters, num_data)
```

### Arguments

data	(matrix): The input data.
weights	(matrix): The probability weight matrix.
meu	(vector): The vector of meu.
sigma	(vector): The vector of sigma (standard-deviations).
prior	(vector): The vector of priors.
num_clusters	(numeric): The number of clusters.
num_data	(numeric): The total number of observations in the data.

### Value

Updated values for meu, sigma and prior.

---

max_heapify	<i>max_heapify: Part of DCEM package.</i>
-------------	---

---

### Description

Implements the creation of max heap. Internally called by the dcem\_star\_train.

### Usage

```
max_heapify(data, index, num_data)
```

### Arguments

data	(NumericMatrix): The dataset provided by the user.
index	(int): The index of the data point.
num_data	(numeric): The total number of observations in the data.

**Value**

A NumericMatrix with the max heap property.

**Author(s)**

Parichit Sharma <parishar@iu.edu>, Hasan Kurban, Mehmet Dalkilic

---

meu\_mv

*meu\_mv: Part of DCEM package.*

---

**Description**

Initialize the meu(s) by randomly selecting the samples from the dataset. This is the **default** method for initializing the meu(s).

**Usage**

```
# Randomly seeding the mean(s).
meu_mv(data, num_meu)
```

**Arguments**

data (matrix): The dataset provided by the user.  
num\_meu (numeric): The number of meu.

**Value**

A matrix containing the selected samples from the dataset.

---

meu\_mv\_impr

*meu\_mv\_impr: Part of DCEM package.*

---

**Description**

Initialize the meu(s) by randomly selecting the samples from the dataset. It uses the proposed implementation from K-means++: The Advantages of Careful Seeding, David Arthur and Sergei Vassilvitskii. URL <http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf>.

**Usage**

```
# Randomly seeding the meu.
meu_mv_impr(data, num_meu)
```

**Arguments**

data (matrix): The dataset provided by the user.  
 num\_meu (numeric): The number of meu.

**Value**

A matrix containing the selected samples from the dataset.

---

meu_uv	<i>meu_uv: Part of DCEM package.</i>
--------	--------------------------------------

---

**Description**

This function is internally called by the dcem\_train to initialize the meu(s). It randomly selects the meu(s) from the range min(data):max(data).

**Usage**

```
# Randomly seeding the meu.
meu_uv(data, num_meu)
```

**Arguments**

data (matrix): The dataset provided by the user.  
 num\_meu (number): The number of meu.

**Value**

A vector containing the selected samples from the dataset.

---

meu_uv_impr	<i>meu_uv_impr: Part of DCEM package.</i>
-------------	---

---

**Description**

This function is internally called by the dcem\_train to initialize the meu(s). It uses the proposed implementation from K-means++: The Advantages of Careful Seeding, David Arthur and Sergei Vassilvitskii. URL <http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf>.

**Usage**

```
# Seeding the meu using the K-means++ implementation.
meu_uv_impr(data, num_meu)
```

**Arguments**

data (matrix): The dataset provided by the user.  
num\_meu (number): The number of meu.

**Value**

A vector containing the selected samples from the dataset.

---

separate\_data *separate\_data: Part of DCEM package.*

---

**Description**

Separate leaf nodes from the heaps.

**Usage**

```
separate_data(heap_list, num_clusters)
```

**Arguments**

heap\_list (list): The nested list containing the heaps. Each entry in the list is a list maintained in max-heap structure.  
num\_clusters (numeric): The number of clusters. Default: **2**

**Value**

A nested list where,  
First entry is the list of heaps with leaves removed.  
Second entry is the list of leaves.

**References**

Parichit Sharma, Hasan Kurban, Mehmet Dalkilic DCEM: An R package for clustering big data via data-centric modification of Expectation Maximization, *SoftwareX*, 17, 100944 URL <https://doi.org/10.1016/j.softx.2021.100944>

---

sigma_mv	<i>sigma_mv: Part of DCEM package.</i>
----------	--

---

**Description**

Initializes the co-variance matrices as the identity matrices.

**Usage**

```
sigma_mv(num_sigma, numcol)
```

**Arguments**

num_sigma	(numeric): Number of covariance matrices.
numcol	(numeric): The number of columns in the dataset.

**Value**

A list of identity matrices. The number of entries in the list is equal to the input parameter (num\_cov).

---

sigma_uv	<i>sigma_uv: Part of DCEM package.</i>
----------	--

---

**Description**

Initializes the standard deviation for the Gaussian(s).

**Usage**

```
sigma_uv(data, num_sigma)
```

**Arguments**

data	(matrix): The dataset provided by the user.
num_sigma	(number): Number of sigma (standard_deviations).

**Value**

A vector of standard deviation value(s).

---

trim_data	<i>trim_data: Part of DCEM package. Used internally in the package.</i>
-----------	---

---

**Description**

Removes the specified column(s) from the dataset.

**Usage**

```
trim_data(columns, data)
```

**Arguments**

columns	(string): A comma separated list of column(s) that needs to be removed from the dataset. Default: ""
data	(dataframe): Dataframe containing the input data.

**Value**

A dataframe with the specified column(s) removed from it.

---

update_weights	<i>update_weights: Part of DCEM package.</i>
----------------	--

---

**Description**

Update the probability values for specific data points that change between the heaps.

**Usage**

```
update_weights(temp_weights, weights, index_list, num_clusters)
```

**Arguments**

temp_weights	(matrix): A matrix of probabilistic weights for leaf data.
weights	(matrix): A matrix of probabilistic weights for all data.
index_list	(vector): A vector of indices.
num_clusters	(numeric): The number of clusters.

**Value**

Updated probabilistic weights matrix.

---

validate_data	<i>validate_data: Part of DCEM package. Used internally in the package.</i>
---------------	---

---

### Description

Implements sanity check for the input data. This function is for internal use and is called by the [dcem\\_train](#).

### Usage

```
validate_data(columns, numcols)
```

### Arguments

columns	(string): A comma separated list of columns that needs to be removed from the dataset. Default: ""
numcols	(numeric): Number of columns in the dataset.

### Details

An example would be to check if the column to be removed exist or not? [trim\\_data](#) internally calls this function before removing the column(s).

### Value

boolean: TRUE if the columns exists otherwise FALSE.



# Index

## \* datasets

ionosphere\_data, 16

build\_heap, 2

DCEM, 3

dcm\_cluster\_mv, 4, 12

dcm\_cluster\_uv, 6, 12

dcm\_predict, 7

dcm\_star\_cluster\_mv, 8, 10

dcm\_star\_cluster\_uv, 9, 10

dcm\_star\_train, 3, 4, 7, 10, 12

dcm\_test, 3, 11

dcm\_train, 3, 4, 7, 12, 24

expectation\_mv, 14

expectation\_uv, 15

get\_priors, 15

insert\_nodes, 16

ionosphere\_data, 12, 16

max\_heapify, 18

maximisation\_mv, 17

maximisation\_uv, 18

meu\_mv, 4, 19

meu\_mv\_impr, 4, 19

meu\_uv, 4, 20

meu\_uv\_impr, 4, 20

separate\_data, 21

sigma\_mv, 22

sigma\_uv, 22

trim\_data, 3, 10, 12, 13, 23, 24

update\_weights, 23

validate\_data, 24